



Christian Franz Kantner, Bakk.rer.soc.oec.

**Konzeption und Entwicklung  
eines  
Online - Fotoausarbeitungstools**

**MASTERARBEIT**

zur Erlangung des akademischen Grades

Master of Science

Masterstudium Softwareentwicklung – Wirtschaft

eingereicht an der

**Technischen Universität Graz**

Betreuer

Univ.-Prof. Dipl.-Ing. Dr.techn. Frank Kappe

Institut für Informationssysteme und Computer Medien

Graz, Jänner 2014



Christian Franz Kantner, Bakk.rer.soc.oec.

**Conception and Implementation  
of an  
Online Photo Print Service**

**MASTER'S THESIS**

to achieve the university degree of

Master of Science

Master's degree programme: Software Development and Business Management

submitted to

**Graz University of Technology**

Supervisor

Univ.-Prof. Dipl.-Ing. Dr.techn. Frank Kappe

Institute for Information Systems and Computer Media

Graz, January 2015

## **Kurzfassung**

Das Thema Fotografie bewegt die Geschichte der Menschheit schon seit über 2000 Jahren. Der Wunsch, Momente der Realität festzuhalten wird getrieben durch verschiedene Motivationen. Fotos werden zur Erinnerung, Informationsübertragung, Dokumentation oder zur Kommunikation gemacht. Grundlegende Idee hinter Fotos ist das Festhalten dieser. Neben digitaler Speicherung werden nach wie vor Fotos gerne gedruckt und aufbewahrt.

In dieser Arbeit wird die Möglichkeit der Ausarbeitung von digitalen Fotos über das Internet untersucht. Dafür werden auf dem zentraleuropäischen Markt befindliche Plattformen analysiert, ausgewertet und in einer Matrix gegenübergestellt.

Es werden Technologien und Methoden zur Entwicklung moderner Web-Anwendungen behandelt und deren Einsatzmöglichkeiten, anhand von Beispielen, demonstriert. Den Abschluss bildet die Vorstellung eines Prototyps der nicht nur eine Ausarbeitungsplattform, sondern auch die Möglichkeit der digitalen Speicherung auf Basis der vorgestellten Technologien implementiert.

## **Abstract**

The topic of photography impacts the history of mankind since more than 2000 years. To capture moments of reality is a desire, motivated through different kinds of reasons. Memories, transmission of information, documentation or communication are major fueling factors. Keeping moments is the base idea behind a photograph. Besides storing pictures in a digital way, people still like to print them.

In this thesis the opportunity to print digital pictures over the internet is investigated. Therefore several established platforms of central Europe are analyzed, evaluated and compared to each other.

Technologies and methods to develop modern web applications are discussed and their application is showed in practical examples. Finally a prototype based on these technologies is demonstrated, which not only permits pictures to be printed, but also to store pictures in a cloud space.

### **Eidesstattliche Erklärung**

Ich erkläre an Eides statt, dass ich die vorliegende Arbeit selbstständig verfasst, andere als die angegebenen Quellen/Hilfsmittel nicht benutzt, und die den benutzten Quellen wörtlich und inhaltlich entnommene Stellen als solche kenntlich gemacht habe.

---

Datum, Ort, Unterschrift

### **Statutory Declaration**

I declare that I have authored this thesis independently, that I have not used other than the declared sources / resources, and that I have explicitly marked all material which has been quoted either literally or by content from the used sources.

---

Date, Place, Signature

# Danksagung

An dieser Stelle möchte ich mich bei allen Personen bedanken die mir geholfen und mich unterstützt haben.

An erster Stelle voran danke ich meinen Eltern, die mir meine Ausbildung ermöglicht und während meines Studiums immer ausnahmslos unterstützt haben.

Des Weiteren bedanke ich mich bei meinem Betreuer Univ.-Prof. Dipl.-Ing. Dr.techn. Frank Kappe für die Unterstützung, Tipps und Geduld die er mir während dieser Arbeit entgegengebracht hat.

Großer Dank gilt auch meinen Freunden und Bekannten die mich immer wieder motiviert haben und für mich eine Quelle der Inspiration während dieser Arbeit waren. Dank gilt Ph.D. MSc. Cristina Schitco, für die konstante Motivation. Danke an Markus Grandel, Thomas Vötsch und Mag. Igor Jurcevic für die Inspiration und Ideen. Danke an Bettina Kropf, für die Korrektur meiner Rechtschreibfehler und Grammatikalischen Verbrechen. Weitere Personen bei denen ich mich bedanken möchte sind Julia und Petra Kaltenbeck, Rene Hartmann, Thomas Maule und die Mitarbeiter des Pro – Photo Teams.

Danke an alle Lehrer aus meiner Hauptschulzeit, die meinen Plan eine Matura zu absolvieren nur belächelt haben. Danke an alle Lehrer meiner HTL, die meinen Plan ein Studium zu absolvieren nur belächelt haben. Euer Zweifel war pure Energie und eure Ignoranz wie Zucker für meinen Erfolg. Danke an alle Lehrer dieser Zeit die Zielstrebigkeit erkannten und wussten zu fördern.

## Inhaltsverzeichnis

1. Einführung .....	1
1.1 Motivation .....	1
1.2 Struktur der Arbeit.....	2
2. Fotografie .....	3
2.1 Eine kurze Geschichte der Fotografie .....	3
2.2 Digitale Bilder .....	5
2.2.1 Vektorgrafik .....	5
2.2.2 Rastergrafik .....	6
2.2.3 Qualitätsmerkmale von digitalen Bildern.....	7
2.3 Aktuelle Trends und Entwicklungen in der Digitalfotografie.....	8
3. Bild Ausarbeitungsplattformen im Internet.....	9
3.1 Snapfish .....	10
3.2 Pixum.....	13
3.3 Foto Charly .....	16
3.4 Happy Foto .....	19
3.5 Fotocom.....	22
3.6 BIPA Foto Shop .....	25
3.7 Plattformen Vergleichsmatrix .....	28
4. Ausgewählte Technologien und Methoden moderner Webanwendungen .....	34
4.1 HTML5.....	35
4.1.1 Das HTML5 Grundgerüst.....	37
4.1.2 Das HTML5 Multi File Select Element .....	38
4.1.3 Die HTML5 File Reader Schnittstelle.....	39
4.1.4 Drag & Drop mit HTML5 .....	40
4.1.5 HTML5 Canvas .....	41

4.1.6 HTML5 Lokaler Speicher .....	44
4.1.7 AJAX 2.0 .....	46
4.1.8 Server Sent Events.....	50
4.2 Client und Server Entwicklung mit Java .....	52
4.2.1 Anonyme Funktionen .....	53
4.2.2 Methoden Referenzen.....	55
4.2.3 Collections und die Java Stream Processing API.....	56
4.2.4 Web Applikationen auf Basis von Spring und Hibernate .....	62
5. Konzeption und Umsetzung der Ausarbeitungsplattform .....	81
5.1 Anforderungsanalyse .....	81
5.2 Systemarchitektur .....	88
5.3 Verwendete Werkzeuge und Technologien.....	92
5.3.1 IntelliJ IDEA .....	92
5.3.2 Apache Chainsaw .....	92
5.3.3 Telerik Fiddler .....	93
5.3.4 Oracle JavaFX Scene Builder.....	93
5.3.5 Regex Buddy .....	93
5.3.6 Adobe Photoshop.....	93
5.3.7 SQL Lite Database Browser.....	94
5.3.8 Java 8 und JavaFX.....	94
5.3.9 Apache Tomcat 8 Embedded, JSP, JSTL und EL .....	94
5.3.10 Spring .....	94
5.3.11 Hibernate .....	95
5.3.12 Hypersonic SQL .....	95
5.3.13 Hikari DBCP .....	95
5.3.14 SQLite.....	95
5.3.15 Simple Logging Facade und Log4j .....	95



5.3.16 JUnit .....	96
5.3.17 Apache Maven.....	96
5.3.18 JQuery.....	96
5.4 Entwicklung grafischer Komponenten mit JavaScript .....	97
5.5 Der Web Client.....	105
5.6 Der Desktop Client .....	115
5.7 Die Implementierung der Server API.....	125
5.8 Quellcodestatistik .....	129
5.9 Kritik und Erweiterungsmöglichkeiten .....	130
6. Zusammenfassung .....	131
7. Quellenverzeichnis .....	133

# Abbildungsverzeichnis

Abbildung 1: Das Snapfish Bestellformular.....	12
Abbildung 2: Das Pixum Bestellformular .....	15
Abbildung 3: Das Foto Charly Bestellformular .....	17
Abbildung 4: Das Happy Foto Bestellformular.....	20
Abbildung 5: Das Fotocom Bestellformular .....	23
Abbildung 6: Das BIPA Bestellformular .....	26
Abbildung 7: TIOBE Index der populärsten Programmiersprachen (Quelle [Tiobe14])	52
Abbildung 8: Deklarative Verarbeitung von Daten durch Streams und Lambda Ausdrücken. Quelle [Urma14] .....	56
Abbildung 9: Spring MVC Architektur. Quelle [Schaefer14] .....	68
Abbildung 10: Der Aufruf des Demo Beispiels. ....	78
Abbildung 11: Skizze der Grafischen Benutzeroberfläche .....	87
Abbildung 12: Server-Architektur.....	88
Abbildung 13: Schnittstelle für das Picture Domain Objekt.....	89
Abbildung 14: Schnittstelle für das Album Domain Objekt .....	90
Abbildung 15: Schnittstelle für das User Domain Objekt.....	90
Abbildung 16: Schnittstelle für das UserAccount Domain Objekt .....	91
Abbildung 17: Schnittstelle für das PictureBlob Domain Objekt .....	91
Abbildung 18: Pager Java Script Komponente .....	97
Abbildung 19: Bild Browser Komponente.....	99
Abbildung 20: Foto Mappe Java Script Komponente .....	101
Abbildung 21: Fotomappe mit Bilder in unterschiedlichen Formaten .....	102
Abbildung 22: Anmeldung und Registrierung am System.....	105
Abbildung 23: Benutzeroberfläche des Web Client .....	106
Abbildung 24: Benutzeroberfläche für neues Album anlegen .....	107
Abbildung 25: Album Attribute editieren .....	107
Abbildung 26: Kontext Menü für Bilder .....	108
Abbildung 27: Bearbeiten der Metadaten eines Bildes .....	108
Abbildung 28: Ausarbeitungsoptionen.....	109
Abbildung 29: Verfügbare Formate .....	109
Abbildung 30: Gegenüberstellung Bild mit und Bild ohne Rahmen.....	110
Abbildung 31: Kontext Menü für Album.....	111

Abbildung 32: Maske zum Hochladen von Bildern .....	112
Abbildung 33: Bildvorschau für das Hochladen .....	113
Abbildung 34: Fortschritt Statusbox .....	113
Abbildung 35: Anmeldung Cloud Client .....	115
Abbildung 36: Cloud Client Hauptmaske .....	116
Abbildung 37: Bearbeiten der lokalen Bildverzeichnisse .....	117
Abbildung 38: Bildverzeichnisse durchsuchen .....	117
Abbildung 39: Album Kontext Menü.....	119
Abbildung 40: Neues Album anlegen .....	119
Abbildung 41: Album entfernen.....	120
Abbildung 42: Album umbenennen .....	120
Abbildung 43: Upload Fortschritt .....	121
Abbildung 44: Speicherplatz Anzeige.....	122
Abbildung 45: Einstellungen des Cloud Client .....	123
Abbildung 46: Cloud Client mit alternativem Thema.....	124
Abbildung 47: Server API Schnittstelle .....	126
Abbildung 48: JUnit Server Test Suite.....	128
Abbildung 49: Quellcodestatistik von Server und Web Client .....	129
Abbildung 50: : Quellcodestatistik des Cloud Client .....	129

# 1. Einführung

## 1.1 Motivation

Das Thema Fotografie bewegt die Geschichte der Menschheit schon seit über 2000 Jahren. Der Wunsch Momente der Realität festzuhalten wird getrieben durch verschiedene Motivationen. Besondere Erlebnisse verursachen eine subjektive emotionale Reaktion im Menschen. Positive Erlebnisse möchten abermals erlebt werden. Negative Erlebnisse möchten möglichst vermieden werden. Um Erinnerungen an solche Erlebnisse und emotionalen Momente festzuhalten, bedienen sich viele Menschen der Fotografie. Fotos dienen dabei als Schlüssel, der die gespeicherte emotionale Erfahrung abermals ins Gedächtnis rufen kann.

Ein Bild sagt mehr als 1000 Worte. Im englischen Original von 1921 „One Look is Worth A Thousand Words“ [Wikipedia1408]. Dies ist wohl eine der gebräuchlichsten Metaphern im Kontext Informationstransfer. Die visuelle Informationsverarbeitung des Menschen ist viel mächtiger als jenes des Hörens und des Lesens. Visuelle Information wird quasi parallel aufgenommen und Eindrücke rasch vom Gehirn verarbeitet. Bei Hören und Lesen erfolgt die Informationsaufnahme nur sequentiell. Menschen bedienen sich diesem Konzept und nutzen Fotografie und Fotos um Informationen schnellstmöglich zu übertragen.

Im Kontext Bildung erfüllt das Bild eine weitere wichtige Funktion, die der Dokumentation. Der Durst nach Wissen und Aufklärung motivierte das Thema Fotografie seit Jahrhunderten. Mit Bildern lassen sich rasche komplexe oder flüchtige Sachverhalte festhalten wodurch diese analysiert werden können.

Neben den erwähnten traditionellen Motivationen kommen selbst heute, 2400 Jahre nach den ersten Experimenten mit Fotografie, neue Motivationen hinzu. Durch Miniaturisierung und Kosteneffizienzsteigerungen ist es möglich geworden das jeder Mensch eine Kamera im täglichen Leben, fast permanent, bei sich hat. Dies resultierte darin das Menschen mittlerweile Fotos nutzen um zu kommunizieren. Anstelle von Ereignissen zu erzählen, wird der erlebte Eindruck anhand eines Fotos über ein Medium, zum Beispiel durch ein Mobiltelefon, übermittelt.

Wie sich erkennen lässt, werden Fotos aus unterschiedlichen Motivationen aufgenommen. Die meisten davon basieren auf der Kernidee des Festhaltens und des Speicherns für die Ewigkeit. Neben der digitalen Speicherung, welche das Problem der Datenerhaltung beinhaltet, ist das beliebteste Medium zur Materialisierung von Fotos immer noch der Druck.

Diese Arbeit beschäftigt sich mit dem Thema der Fotoausarbeitung von digitalen Bildern unter Verwendung moderner Kommunikationstechnologien, um die erwähnten Bedürfnisse zum Zwecke eines kommerziellen Nutzens zu befriedigen.

## **1.2 Struktur der Arbeit**

Diese Arbeit ist in 6 Kapitel gegliedert. Kapitel 1 stellt die Motivation vor.

In Kapitel 2 dieser Arbeit wird zunächst die geschichtliche Entwicklung der Fotografie beleuchtet. Es wird eine Zeitreise über 2400 Jahre unternommen, zurück zu Aristoteles bis hin zu Halbleitern und der digitalen Revolution. Abschließend werden aktuelle Trends der digitalen Fotografie behandelt.

Kapitel 3 untersucht Online-Ausarbeitungsanbieter am zentraleuropäischen Markt. Es werden 6 Plattformen hinsichtlich deren Funktionsumfang und Drucksorten von etablierten Unternehmen analysiert.

Kapitel 4 stellt moderne Technologien und Methoden vor mit deren Hilfe Web-Plattformen entwickelt werden können. Behandelt wird im Speziellen die HTML5 Technologie, welche erhebliche Verbesserungen der Benutzerfreundlichkeit ermöglicht, sowie die Java 8 Technologie, die ein Zugeständnis an das funktionale Programmierparadigma ist. Des Weiteren werden die Frameworks Spring und Hibernate vorgestellt und welche Vorteile durch Dependency Injection erreicht werden.

Kapitel 5 stellt den entwickelten Prototypen vor der aus einer Server-Applikation, einem Web-Client sowie einem Desktop-Client besteht. Neben der Anforderungsanalyse werden auch die Architektur sowie die Entwicklung grafischer Komponenten beleuchtet. Abschließend werden Kritik und Erweiterungsmöglichkeiten besprochen.

In Kapitel 6 werden die Ergebnisse der Arbeit zusammengefasst.

## 2. Fotografie

### 2.1 Eine kurze Geschichte der Fotografie

Die Anfänge der Fotografie reichen bis in das 4. Jahrhundert vor Christus zurück. Aristoteles hielt in seiner Schrift „Problemata physica“ zum ersten Mal das Prinzip der Camera obscura, lateinisch für Dunkelkammer, fest. Den physikalischen Sachverhalt beleuchtete später Leonardo Da Vinci. [Wikipedia1401] Erste Versionen dieser Kamera bestanden alle aus einem Raum der durch ein winziges Loch in einer Seitenwand ein punktgespiegeltes Abbild auf die gegenüberliegende Wand projizierte. Ermöglicht wird dieser Vorgang durch den Diffraktionseffekt. [Hoy06]

Ein Vorläufer der Kamera wurde zunächst zu Beginn des 19. Jahrhunderts entwickelt. Dabei handelte es sich um ein optisches Element und einer mechanischen Gerätschaft die die Umrisse einer Szene auf einen Zeichenblock projizierten. Hobbykünstler konnten so mittels eines Stiftes eine Zeichnung anfertigen indem sie die Konturen der Abbildung nachzeichneten. [Hoy06]

Im Jahr 1819 wurde von John Herschel die chemische Substanz Natriumthiosulfat entdeckt. Durch dessen chemische Eigenschaften wurden die Entwicklungen, die später zur Fotografie führten, überhaupt erst ermöglicht. [Frizot98]

Erste Experimente, die von der Vision des Festhaltens der Realität auf Papier motiviert wurden, unternahm der Franzose Nicephore Niepce. Er verfolgte zum einen das Ziel der Reproduktion von bereits existierenden Bildern, und zum Anderen die Erzeugung von direkten und vollständigen Bildern der Natur. Ab 1822 kam Niepce mit seinen Forschungsarbeiten immer zügiger voran. 1826 gelang ihm nach achtstündiger Belichtungszeit die erste beständige Aufnahme eines Bildes. Es zeigte einen Bauernhof seines Anwesens. [Hoy06]

Später im Jahre 1827 arbeitete Niepce zusammen mit Louis-Jaques-Mande Daguerre zusammen. Daguerre entdeckte Methoden um die Belichtungszeit wesentlich zu verkürzen und die Qualität der Bilder drastisch zu erhöhen. Das neu entwickelte Verfahren nannte er Daguerreotypie welches er patentierte und der französischen

Regierung überließ, welche es 1839 veröffentlichte und der ganzen Welt zugänglich machte. [Hoy06]

Der Engländer William Henry Fox Talbot entwickelte 1840 das Negativ-Positiv-Verfahren. Im Gegensatz zur Daguerreotypie, welches nur die Erzeugung von Unikaten erlaubte, konnten mit diesem neuen Verfahren nun Bilder vervielfältigt werden. Dies war das von Niepce ursprünglich verfolgte Ziel. [Frizot98]

Lois Desire Blanquart-Evrard entdeckte um 1850 Chemikalien die den Kontrast der Bilder erheblich verbesserten und somit auch die Bildqualität erheblich positiv beeinflussten. 1851 wurde vom Engländer Frederick Scott Archer ein neues Verfahren entwickelt, das die benötigte Belichtungszeit auf zwei bis drei Sekunden reduziert. Ab 1856 wurde Blitzlicht verwendet welches fotografieren ohne Tageslicht ermöglichte. In Schottland gelang es James Clerk Maxwell im Jahr 1861 die erste Farbfotografie herzustellen. [Hoy06] Die erste wissenschaftliche Verwendung von Fotografie erfolgte 1871 in einer Arbeit zur Dokumentation der Bewegungsabläufe von Tieren. [Frizot98] Durch die Entdeckungen von Richard L. Maddox und Charles Harper Bennet um 1880 wurde die Belichtungszeit drastisch auf 0,2 Sekunden verkürzt. Das revolutionierte die Fotografie da ab diesen Zeitpunkt keine Stative mehr notwendig waren. Die Fotografie wurde dadurch für wirtschaftliche Interessen bedeutend. George Eastman erfand den Film auf Rolle und gründete 1888 die Firma Kodak. Kodak brachte die erste kommerzielle Kamera, die Kodak Nr.1, auf den Markt. [Frizot98]

Die Brüder Lumiere entwickelten um 1907 das erste praktikable Verfahren für Farbfotografie. Leopold Mannes und Leopold Godowsky entwickelten um 1935 einen Farbfilm der in jeder beliebigen Kamera verwendet werden konnte. [Newhall98]

Nach Ende des zweiten Weltkrieges begann mit der Sofortbildfotografie ein neuer Abschnitt. Bilder konnten nun innerhalb von Minuten ohne Fotolabor aufgenommen werden. [Hoy06]

Durch das Aufkommen der Digitalfotografie begann in der Fotoindustrie eine Revolution. 1981 stellte Sony die erste Digitalkamera vor. Intensive Forschungsarbeit verbesserte die Qualität von Digitalkameras enorm sodass diese den analogen Spiegelreflexkameras heute um nichts mehr nachstehen. Digitalkameras speichern Bilder nicht mehr auf einem beschichteten Film sondern auf digitalen Medien. [Hoy06]

## 2.2 Digitale Bilder

Um visuelle Informationen mittels eines Computers maschinell zu verarbeiten stehen prinzipiell zwei voneinander verschiedene Datenstrukturen zur Verfügung. Jede dieser beiden Datenstrukturen verfügt über bestimmte Vor- und Nachteile. Welche Datenstruktur für eine Anwendung am geeignetsten ist hängt von deren Anforderungen sowie den verwendeten Algorithmen ab.

### 2.2.1 Vektorgrafik

Bei der Vektorgrafik handelt es sich um eine Datenstruktur die es erlaubt visuelle Informationen textuell zu beschreiben. Ein Interpreter liest diese Beschreibung von einer Quelle, zum Beispiel einer Datei, interpretiert die Anweisungen, und stellt diese auf einem Anzeigegerät dar. Die textuellen Anweisungen definieren dabei unterschiedliche grafische Formen wie zum Beispiel einen Kreis, ein Rechteck und desgleichen. Auch Angaben zur Position und Farbe sind durch Anweisungen festgelegt. Die möglichen Instruktionen die zur Erstellung einer Vektorgrafik verwendet werden können, werden vom verwendeten Standard vorgegeben. Einer der bekanntesten Standards ist dabei der Scalable Vector Graphics Standard, oder kurz SVG. [Eisenberg02]

Der SVG Standard in der Version 1.1 wird aktuell bereits von den Marktführenden Webbrowsern, wenn auch nicht immer uneingeschränkt, unterstützt. [Caniuse14]

Verwendet werden kann zur Erstellung von SVG Grafiken jeder Texteditor. Komplexe Vektorgrafiken werden jedoch vorzugsweise mit einem geeigneten Editor erstellt. Dies hat den Vorteil, dass Benutzer sich nicht mit der Grammatik und Syntax des Formats auseinandersetzen müssen. Der Vorteil von Vektorgrafik liegt klar in der offenen Struktur des Formats, sowie dass innerhalb der Definition von grafischen Informationen, auch maschinell gesucht werden kann, ohne komplexe Algorithmen zu verwenden. Als Nachteil erweist sich das Format allerdings dann, wenn komplexe farbliche Informationen, pixelgetreu gespeichert werden sollen. Vektorgrafik findet vielfach Anwendung in CAD Programmen, Desktop-Publishing sowie Layout-Gestaltung für den Druck. [Eisenberg02]



## **2.2.2 Rastergrafik**

Bei der Rastergrafik wird die visuelle Information in einem rechteckigen Datenfeld bzw. einem Array gespeichert. Der Index eines jeden Eintrags des Arrays bestimmt dabei die Position der Information am Anzeigegerät. Der Inhalt des Eintrags referenziert dabei einen Farbwert der zur Darstellung verwendet werden soll. Jeder Eintrag des Datenfeldes wird dabei auch als Pixel bezeichnet. Die Rastergrafik ist die ideale Datenstruktur zum Speichern von digitalen Bildern. [Eisenberg02]

Im Laufe der Jahre wurde eine Vielzahl von unterschiedlichen Rastergrafik Formaten entwickelt die sich jeweils durch die konkrete Implementierung voneinander unterscheiden. Die meisten dieser Formate beschäftigen sich damit die Bildinformation komprimiert zu speichern ohne einen wesentlichen qualitativen Verlust zu verursachen. Im Folgenden werden die wichtigsten Bildformate von Internet-Anwendungen kurz vorgestellt.

### **2.2.2.1 Das Graphics Interchange Format – GIF**

Das GIF-Format wurde 1987 von der Firma CompuServe entwickelt. Es unterstützt die Speicherung der Farbinformation mit einer Größe bis zu 8 Bit. Das ermöglicht die Verwendung von 256 verschiedenen Farben aus dem 24 Bit RGB Farbraum. Eine Besonderheit des GIF Formats ist die Unterstützung von Animationen. Dabei wird jedes Bild der Animation als Einzelbild mit eigener 256 Farben Palette in die Datei eingebettet. Um Fotos von Digitalkameras zu speichern eignet sich das Format nicht. Jedoch können einfache Bilder, wie zum Beispiel Comics, sehr effizient gespeichert werden. GIF verwendet den Lempel-Ziv-Welch Algorithmus zur Kompression der Bildinformation. [Miano99]

### **2.2.2.2 Das Joint Photographic Experts Group Format – JPEG**

Der JPEG-Standard zur Speicherung von visuellen Informationen wurde 1986 von der Joint Photographic Expert Group entwickelt und 1994 als ISO Standard verabschiedet. Der Standard beschreibt dabei den Algorithmus wie Bildinformationen in einen binären Datenstrom komprimiert werden können, als auch wie dieser Prozess wieder rückgängig gemacht werden kann. Wie diese Binärdaten in eine Datei, die als Container fungiert, gespeichert werden kann wird im JPEG File Interchange Format (JFIF) definiert.

Die Kompression erfolgt bei JPEG nicht verlustfrei. Anwender können anhand eines

Qualitätsfaktors entscheiden wie gut oder wie schlecht das Endergebnis aussehen soll. Linear dazu verhält es sich dann mit der resultierenden Dateigröße. Die Farbmodelle die von JPEG unterstützt werden ermöglichen den Einsatz für Digitalfotos.

### **2.2.2.3 Das Portable Network Graphics Format – PNG**

Das PNG-Format erlaubt die Speicherung von Bildinformation unter der Verwendung einer verlustlosen Komprimierung. Es stellt eine Weiterentwicklung des GIF Formats dar. Der Standard wird von der PNG Development Group des W3C entwickelt. PNG wurde speziell für die Verwendung im Internet verwendet und unterstützt daher nur RGB Farbräume mit Farbpaletten bis zu 24 Bit. 2004 wurde der PNG Standard auch als ISO Standard verabschiedet. [Miano99]

Obwohl PNG als Weiterentwicklung des GIF-Formats betrachtet werden kann unterstützt es keine Animationen. Mit dem APNG-Standard versucht man diesen Sachverhalt nun zu korrigieren. [Wikipedia1402]

### **2.2.3 Qualitätsmerkmale von digitalen Bildern**

Durch die digitale Bearbeitung, Komprimierung und Übertragung von Bildinformationen können Qualitätsverluste auftreten. Je nach Anwendungsfall muss entschieden werden welcher Qualitätsverlust des Bildes akzeptabel und vertretbar ist. Die Qualität eines Bildes kann prinzipiell auf zwei Arten gemessen werden. Bei der subjektiven Qualitätsbestimmung trifft eine Reihe von Experten anhand von zuvor festgelegten Kriterien eine Bewertung des Bildes. Dieses Verfahren ist aufwendig sowie kostspielig und wird vorwiegend bei TV- und Videoproduktionen eingesetzt. Objektive quantitative Methoden bedienen sich von Algorithmen. Dabei wird ein digitales Bild mit einem Referenzbild verglichen und der Unterschied von Kennwerten, wie zum Beispiel der durchschnittliche quadratische Unterschied, festgestellt. Methoden dieser Art erlauben die Automatisierung zum Finden von Optimierungsparametern die den Anwender bei der Wahl des Komprimierungsfaktors unterstützen können. Weitere Merkmale digitaler Bilder hängen stark von der menschlichen Wahrnehmung ab. Das sind zum Beispiel Kontrast, Qualität der Kanten und Form, Texturen und Farbwerte etc. Anhand mathematischer Methoden kann für jedes Bild ein Histogramm berechnet werden welches die Verteilung von Farben visualisiert. Anhand des Histogramms können so automatisiert Bildoperationen

vorgenommen werden die eine subjektive Verbesserung der Bildwahrnehmung zur Folge haben. [Sonka08]

## **2.3 Aktuelle Trends und Entwicklungen in der Digitalfotografie**

Die Ursprüngliche Idee der Fotografie war das Festhalten von Momenten und Situationen. Eine große Rolle spielte dabei der Bedarf nach Erinnerung. Neben der Dokumentationsfunktion erfüllte Fotografie auch oft eine Beweisfunktion. Letztere leidet stark unter der modernen digitalen Fotografie und den Möglichkeiten der digitalen Nachbearbeitung. Dies ist wohl einer der großen Nachteile der digitalen Fotografie.

Durch den Prozess die Miniaturisierung von Technik und er Verbesserung der Energieeffizienz von mobilen Geräten erscheinen aber auch noch heute, knapp 194 Jahre nach der Erfindung der Fotografie, neue Anwendungsbereiche und Möglichkeiten. Zu Beginn des neuen Jahrtausends wurden vermehrt Digitalkameras in Mobiltelefonen verbaut. Dies ermöglicht seither einen absolut neuen Weg der Kommunikation. Wurde früher vermehrt telefoniert oder textuelle Nachrichten ausgetauscht, so werden nunmehr digitale Bildinhalte getauscht um Inhalte mitzuteilen. Die Kombination von funkgesteuerten Flugzeugmodellen und Digitalkameras erlaubt das Fotografieren aus großer Höhe oder an Orten die sonst nicht oder nur schwer zugänglich sind. Der Einsatz von Bodykameras wird zunehmend, im privaten Bereich, sehr gerne im Sport verwendet um Leistungen und Erlebnisse zu dokumentieren. Aber auch der Einsatz solcher Kameras bei Ordnungsorganen und zivilem Schutzpersonal findet immer mehr Anwendung. Nicht zuletzt werden immer mehr digitale Fotos von automatisierten Robotern im zivilen Bereich erstellt, etwa zum Zweck der Überwachung. [Westphalen13]

Die Vermutung liegt nahe, dass auch in Zukunft noch weitere spannende Einsatzgebiete der digitalen Fotografie entstehen.

# 3. Bild Ausarbeitungsplattformen im Internet

Im folgenden Abschnitt sollen bereits existierende Plattformen zur digitalen Fotoausarbeitung genauer untersucht werden. Fokussiert werden Plattformen die ihren Service in Europa bzw. im speziellen in Österreich anbieten. Wie sich zeigen wird, gibt es bereits eine Vielzahl von Anbietern mit einer sehr breit gefächerten Auswahl an Leistungen. Eine Google Suche ergab auf die Suchanfrage „Online Fotoausarbeitung Österreich“ insgesamt 21.700 Treffer.

Die Österreichische Zeitschrift Konsument führte im Jahr 2007 eine Untersuchung der österreichischen Plattformen durch. Betrachtet wurde dabei eine Vielzahl von Anbietern und deren Leistungen. Das kostenpflichtige Ergebnis der Untersuchung kann auf der Webseite abgerufen werden. [Konsument14] Für die Erhebung im Zuge dieser Diplomarbeit wurde eine Teilmenge der untersuchten Plattformen ausgewählt. Sie stellt den Ausgangspunkt der Analyse dar. Jede der ausgewählten Plattformen wurde daraufhin analysiert. Im Mittelpunkt steht dabei:

- Leistungsumfang der Plattform. Welche Leistungen und Produkte werden auf der Plattform angeboten?
- Funktionsumfang der Services. Welche Optionen bzw. Funktionen werden von der Plattform bereitgestellt um die gewünschten Produkte zu konfigurieren.
- Eingesetzte Technologie. Wie wird die Plattform technisch umgesetzt?
- Bestellvorgang und Kundenbetreuung. Wird auf einen kundenfreundlichen Bestellvorgang Rücksicht genommen? Gibt es die Möglichkeit mit Kundenbetreuern in Kontakt zu treten?

Einige Plattformen die von Österreichischen Großunternehmen betrieben werden wurden in dieser Arbeit absichtlich nicht untersucht. Dazu gehören zum Beispiel die Plattformen von Interspar oder Hartlauer. Der Grund hierfür ist, dass diese nur eine thematisch angepasste Version der CEWE Plattform verwenden. CEWE ist eine Marke der CEWE Stiftung & Co. KGaA. Diese bietet Unternehmen Partnerschaften für den Vertrieb von Digitalen Druckleistungen an. Eine Plattform, die auf CEWE basiert, wird von der BIPA GmbH betrieben und auch in dieser Arbeit betrachtet. Die Plattformen

von Interspar und Hartlauer sind analog zu dieser. Die folgende Liste zeigt welche Anbieter analysiert wurden.

Snapfish	<a href="http://www.snapfish.at/">http://www.snapfish.at/</a>
Pixum	<a href="http://www.pixum.at/">http://www.pixum.at/</a>
Foto Charly	<a href="http://www.fotocharly.at/">http://www.fotocharly.at/</a>
Happy Foto	<a href="http://www.happyfoto.at/">http://www.happyfoto.at/</a>
Fotocom	<a href="http://at.foto.com/">http://at.foto.com/</a>
BIPA Foto Shop	<a href="http://fotoshop.bipa.at/">http://fotoshop.bipa.at/</a>

### 3.1 Snapfish

Die Plattform Snapfish wurde 1999 in den USA von Rajil Kapoor, Bala Parthasarathy, Suneet Wadhwa und Shripati Acharya gegründet und 2005 von Hewlett Packard übernommen. [Wikipedia1403] Der Hauptsitz des Unternehmens ist in San Francisco, USA.

Nach eigenen Angaben verfügt das Portal über 100 Millionen Kunden und verwaltet über 1 Milliarde Fotos in ihrem System. [Snapfish14]

Snapfish bietet das Drucken von Digitalfotos auf bzw. für folgende Produkte an:

Fotodruck auf Papier	Spiele
Fotobücher	Kleidung
Passfotos	Taschen
Kalender	
Grußkarten, Briefkarten, Klappkarten	
Tassen und Krüge	
Poster	
Leinwände	

Für das Hochladen der Bilddateien bietet Snapfish auf der Webseite ein Formular mit Wizzard an. Verfolgt wird dabei auch die Idee des Albums, das als Container für Fotos dienen soll. Es können mehrere Dateien gleichzeitig ausgewählt werden, und dann der Reihe nach auf den Server geladen werden. Snapfish fragt vor dem Vorgang noch nach der Dateigröße des Fotos und unterscheidet dabei zwischen „normale“ und „große“ Fotos geht aber nicht weiter auf den Unterschied von normal und groß ein. Direkt beim

Hochladen der Fotos kann noch eine Option zum Entfernen von roten Augen und zur Bildoptimierung gewählt werden. Dies erfolgt vermutlich automatisiert. Nachdem Fotos auf den Server geladen wurden, muss gewählt werden, welche Fotos aus welchen Alben ausgearbeitet werden sollen. Im dritten Schritt erfolgt die Auswahl der gewünschten Größen und Anzahl der Fotos. Abbildung 1 zeigt das Snapfish Bestellformular.

Die folgende Tabelle listet ermittelte Merkmale und Funktionen der Bildbestellung:

Anzahl der Formate	5
Anzahl der Papiersorten	1
Rahmen verfügbar	Ja
Bildoptimierungsoption	Ja
Collagenoption	Ja
Geschenksoption	Nein
Import von Facebook	Ja
Import von Flickr	Ja
Import von Instagram	Nein
Kostenvorschau	Ja
Bedienungsassistent	Ja
Besonderheiten	Mehrere Designer für unterschiedliche Produkte

Die folgende Tabelle listet ermittelte Merkmale und Funktionen der Fotobuchbestellung:

Online Designer	Ja
Fotobuch Software	Ja
Unterstützte Systeme	Windows, Mac, Linux;
Anzahl der Formate	9
Anzahl der Papiersorten	1
Seitenzahl minimal	Keine Angabe
Seitenzahl maximal	Keine Angabe
Einband	Softcover, Hardcover, Ledereinband, Leineneinband
Collagenoption	Ja
Bedienungsassistent	Nein
Besonderheiten	Foto kann in Einband eingearbeitet werden.


[KLICKEN SIE HIER FÜR UNSERE BESTELLFRISTEN ZUM WEIHNACHTSFEST](#)


[Warenkorb und Gutscheine](#)
[2](#)
[Lieferung](#)
[3](#)
[Zahlung](#)
[4](#)
[Überprüfen](#)
[5](#)
[Bestellbestätigung](#)

### Ihr Warenkorb

Bitte überprüfen Sie Ihre Bestellung, und klicken Sie dann auf "Kasse", um fortzufahren.

[Hier finden Sie Hilfe](#)

Fotos
Auswahl minimieren

**Schnellauswahl** Die hier gewählte Menge pro Größe wird für alle Abzüge verwendet.

Anzahl:  10x15 Premium [Anwenden](#)

**Druckoptionen**

Papiersorte: Hochglanz

Automatische Optimierung:  Fotos automatisch optimieren

Rahmen hinzufügen [Beispiele anzeigen](#)

Kein Rahmen

Preise

[Alle Preise für Abzüge anzeigen](#)

[Versandkosten](#)

Weitere...



[Weitere Fotos hinzufügen](#)

[Weitere Fotocollage-Abzüge hinzufügen](#)

Die Markierung ▲ bedeutet, dass Abzüge in der ausgewählten Größe nicht empfohlen werden.

**Ihr Guthaben**

20 Kostenlose(s) Foto(s) im Format 10x15

	Preise	Anzahl	Gesamt
 9x13 Premium	0,09 €	<input type="text" value="0"/>	0,00 €
10x15 Premium	0,09 €	<input type="text" value="1"/>	0,09 €
11x15 Premium	0,14 €	<input type="text" value="0"/>	0,00 €
13x18 Premium	0,18 €	<input type="text" value="0"/>	0,00 €
20x30 Premium ▲	0,89 €	<input type="text" value="0"/>	0,00 €
Brieftaschen-Drucke ▲	1,99 €	<input type="text" value="0"/>	0,00 €
Passfoto-Drucke	1,99 €	<input type="text" value="7"/>	13,93 €
Fotoset ▲	3,99 €	<input type="text" value="0"/>	0,00 €
<a href="#">Fotoposter erstellen</a>			
 9x13 Premium	0,09 €	<input type="text" value="0"/>	0,00 €
10x15 Premium	0,09 €	<input type="text" value="1"/>	0,09 €
11x15 Premium ▲	0,14 €	<input type="text" value="7"/>	0,98 €
13x18 Premium ▲	0,18 €	<input type="text" value="0"/>	0,00 €
20x30 Premium ▲	0,89 €	<input type="text" value="8"/>	7,12 €
Brieftaschen-Drucke ▲	1,99 €	<input type="text" value="0"/>	0,00 €
Passfoto-Drucke ▲	1,99 €	<input type="text" value="8"/>	15,92 €
Fotoset ▲	3,99 €	<input type="text" value="0"/>	0,00 €
<a href="#">Fotoposter erstellen</a>			
2 Guthabepunkte für 10x15 Premium Abzug/Abzüge wurden angewandt. Sie haben 18 Verbleibende.			-0,18 €
Abzüge:	2 (10x15 Premium), 7 (11x15 Premium), 8 (20x30 Premium), 15 (Passfoto-Drucke)	Abzüge insgesamt:	<b>37,95 €</b>



### Gutscheine & Geschenkgutscheine einlösen

Zwischensumme: 37,95 €

[Einlösen](#)

Preise gelten für Postversand  
Preis inkl. Mehrwertsteuer  
Im Preis sind keine Versandkosten enthalten.

Pro Bestellung kann nur ein Gutschein eingelöst werden.

**Gesamtsumme: 37,95 €**

[Weiter einkaufen](#)

[Kasse](#)

[Warenkorb leeren](#)

**Abbildung 1: Das Snapfish Bestellformular**

Die folgende Tabelle listet ermittelte Merkmale und Funktionen der Plattform im Allgemeinen:

Registrierung erforderlich	Ja
Willkommensbonus	Ja, 20 Gratis Fotos
Versandkostenfreie Lieferung	Ja, ab 50 Euro
Versandkostenbasis	Bestellwert
Abholen im Shop – Funktion	Nein
Online bestellbar	Ja
Offline bestellbar	Nein
Transparenter Bestellvorgang	Ja
Bezahlmethoden	Kreditkarte, auf Rechnung, Bankeinzug, Paypal
Kundenservice	Via E-Mail
Newsletter System	Ja
Partnerprogramm (bzw. Franchise)	Nein
Empfehlungen	Nein
Eingesetzte Technologien	Flash und Desktop Softwarepaket

### 3.2 Pixum

Die Plattform Pixum wird von der Firma Pixum - Dignet GmbH & Co. KG betrieben. Sie wurde im Jahr 2000 von Daniel Attallah gegründet. Hauptsitz des Unternehmens ist Köln, Deutschland.

Pixum bietet das Drucken von Digitalfotos auf bzw. für folgende Produkte an:

Fotodruck auf Papier	Spiele (Spielkarten, Plüschtiere, diverse Spiele)
Fotobücher	Kleidung
Passfotos	Taschen
Kalender	Notizblöcke und diverse Büroartikel
Postkarten, Klappkarten	Mobiltelefonhüllen
Tassen, Krüge und Flaschen	Klebefolien und Sticker
Poster	Glasbilder
Leinwände	Dekorartikel



Pixum bietet für die Bestellung der Fotos eine Maske an die das Hochladen mittels Drag & Drop erlaubt, oder aber auch die altbekannte Methode des Ladeknopfes. Die Maske erlaubt das selektieren der Bilder und die anschließende Wahl des Formates, sowie Zusatzoptionen wie Rahmen und Stückzahl. Nach erfolgter Wahl können weitere Fotos hochgeladen und dem Warenkorb hinzugefügt werden. Abhängig von der Anzahl der bestellten Fotos, wird sofort ein möglicher Rabatt angezeigt und in der Preisvorschau vom finalen Preis abgezogen. Fotos von minderer Qualität werden in der Warenkorbübersicht sowie in der Bestellmaske markiert. Positiv zu erwähnen sei, dass keine Anmeldung erforderlich ist um den Bestellvorgang einzuleiten. Benutzer können auch ihren Facebook Zugang verwenden und müssen keinen neuen Zugang auf der Seite registrieren. Abbildung 2 zeigt das Pixum Bestellformular. [Pixum14]

Die Folgende Tabelle listet ermittelte Merkmale und Funktionen der Bildbestellung:

Anzahl der Formate	6
Anzahl der Papiersorten	1
Rahmen verfügbar	Ja
Bildoptimierungsoption	Nein
Collagenoption	Nein
Geschenksoption	Nein
Import von Facebook	Nein
Import von Flickr	Nein
Import von Instagram	Nein
Kostenvorschau	Ja
Bedienungsassistent	Nein
Besonderheiten	Mehrere Designer für unterschiedliche Produkte

Die Folgende Tabelle listet ermittelte Merkmale und Funktionen der Fotobuchbestellung:

Online Designer	Ja
Fotobuch Software	Ja
Unterstützte Systeme	Windows, Mac, Linux, iOS, Android;
Anzahl der Formate	8
Anzahl der Papiersorten	5
Seitenzahl minimal	Keine Angabe

Seitenzahl maximal	Keine Angabe
Einband	Heft, Softcover, Hardcover, Leinencover, Premiumleinen, Kunstleder
Collagenoption	Nein
Bedienungsassistent	Nein
Besonderheiten	Online Fotobuch Designer nur mit limitiertem Funktionsumfang und Seitenbeschränkung.

Abbildung 2: Das Pixum Bestellformular

Die Folgende Tabelle listet ermittelte Merkmale und Funktionen der Plattform im Allgemeinen:

Registrierung erforderlich	Nein
Willkommensbonus	Ja, Vergünstigungen auf ausgewählte Produkte.
Versandkostenfreie Lieferung	Nein
Versandkosten Basis	Anzahl der Produkte sowie physikalische Eigenschaften
Abholen im Shop – Funktion	Nein
Online bestellbar	Ja
Offline bestellbar	Nein
Transparenter Bestellvorgang	Ja
Bezahlmethoden	Kreditkarte, Auf Rechnung, Bankeinzug, Paypal
Kundenservice	Telefonisch und via E-Mail
Newsletter System	Ja
Partnerprogramm (bzw. Franchise)	Ja, mit Provisionssystem
Empfehlungen	Ja, von Deutschen Zeitschriften
Eingesetzte Technologien	HTML5, Flash und Desktop Softwarepaket

### 3.3 Foto Charly

Die Plattform Foto Charly wird von der Firma Color Drack GmbH & Co KG mit Hauptsitz in Schwarzach im Pongau, Österreich, betrieben. Neben der österreichischen Plattform existieren auch noch Versionen für den Schweizer sowie für den deutschen Markt. Das 2011 gegründete Unternehmen erwirtschaftet mit rund 87 Mitarbeitern einen Umsatz von ca. 9,5 Millionen Euro. [FirmenABC1401]

Die Plattform bietet dem Benutzer ein einfaches, aber nutzdienliches, Bestellformular. Das Hochladen der Bilder erfolgt mittels einer klassischen Upload-Schaltfläche. Dabei können auch mehrere Fotos in einem Schritt hochgeladen werden. Jeder Ladevorgang wird dabei gesondert behandelt. So wird zunächst Format, Stückzahl und Ausarbeitungsoptionen gewählt. Danach werden die Fotos hochgeladen die für diese Einstellung ausgearbeitet werden sollen. Im Anschluss daran können für die bereits hochgeladenen Fotos noch einmal andere Ausarbeitungsoptionen gewählt und dem

Warenkorb hinzugefügt werden. Abbildung 3 zeigt das Foto Charly Bestellformular. [FotoCharly14]

Foto Charly bietet das Drucken von Digitalfotos auf bzw. für folgende Produkte an:

Fotodruck auf Papier	Spiele (Memory und Puzzle)
Fotobücher	Kleidung
Bilder mit Rahmung	Taschen
Kalender	Holzbilder
Grußkarten	Dekorartikel
Tassen, Krüge und Flaschen	Fotokristall
Poster	Lebensmittel (Schokolade)
Leinwände	

Preisliste / Artikel für Online-Bestellung wählen


- 6x9 cm**  
1 Stück € 0,13  
ab 26 Stück € 0,11  
ab 51 Stück € 0,09
- 9x13/11 cm**  
1 Stück € 0,10
- 10x15/13 cm**  
1 Stück € 0,11  
ab 40 Stück € 0,10  
ab 300 Stück € 0,09
- 11x17 cm**  
1 Stück € 0,16
- 13x18/16 cm**  
1 Stück € 0,16

**Menge:**  
 Stk.

**Artikel-Optionen:**  
Oberfläche:

Autom. 2:3 Umwandlung:

**Diese Bilder wurden bereits übertragen:** ▶ alle Bilder auswählen



← Neue Bilder übertragen In den Warenkorb legen

**Hinweis zu den Papiergrößen von Fotos**

Für die Bestellung von Fotos werden Produktbezeichnungen wie 9x13 oder 10x15 verwendet, die **genaue Größe** der Ausarbeitung hängt immer vom Ursprungsformat Ihres Bildes ab (Seitenverhältnis 2:3 oder 3:4), beachten Sie daher die genaue Ausarbeitungsgröße (siehe Tabelle)!

Format	Größe 2:3-Format	Größe 3:4-Format
Foto 6x9cm	5,9x8,9 cm	5,9x7,9 cm

Abbildung 3: Das Foto Charly Bestellformular

Die folgende Tabelle listet ermittelte Merkmale und Funktionen der Bildbestellung:

Anzahl der Formate	5
Anzahl der Papiersorten	1
Rahmen verfügbar	Nein
Bildoptimierungsoption	Ja
Collagenoption	Nein
Geschenksoption	Nein
Import von Facebook	Nein
Import von Flickr	Nein
Import von Instagram	Nein
Kostenvorschau	Nein
Bedienungsassistent	Nein
Besonderheiten	Derselbe Designer für alle Drucksorten

Die folgende Tabelle listet ermittelte Merkmale und Funktionen der Fotobuchbestellung:

Online Designer	Nein
Fotobuch Software	Ja
Unterstützte Systeme	Windows, Mac, Linux
Anzahl der Formate	5
Anzahl der Papiersorten	1
Seitenzahl minimal	16
Seitenzahl maximal	100
Einband	Softcover, Hardcover, Fotoheft, Taschenbuch
Collagenoption	Nein
Bedienungsassistent	Nein

Die folgende Tabelle zeigt ermittelte Merkmale und Funktionen der Plattform im Allgemeinen:

Registrierung erforderlich	Nein
Willkommensbonus	Nein
Versandkostenfreie Lieferung	Nein
Versandkosten Basis	Abmessungen der Produkte
Abholen im Shop – Funktion	Nein
Online bestellbar	Ja
Offline bestellbar	Nein
Transparenter Bestellvorgang	Ja
Bezahlmethoden	Kreditkarte, Auf Rechnung;
Kundenservice	Telefonisch und via E-Mail
Newsletter System	Ja
Partnerprogramm (bzw. Franchise)	Nein
Empfehlungen	Ja, von Schweizer Onlineportal
Eingesetzte Technologien	HTML5 und Desktop Softwarepaket

### 3.4 Happy Foto

Die Plattform Happy Foto wird von der Firma HappyFoto GmbH mit Hauptsitz in Freistadt, Österreich betrieben. Das Unternehmen wurde 1978 von Bernhard Kittel als Einmannbetrieb gegründet. Firmengegenstand zur Zeit der Gründung war die Fotoausarbeitung. Heute erwirtschaften 70 Mitarbeiter einen Jahresumsatz von 25 Millionen Euro. [FirmenABC1402]

Die Plattform bietet den Kunden zwei Möglichkeiten Fotos hochzuladen: Eine Methode wurde mittels Java Applet Technologie umgesetzt die nur mehr eingeschränkt von modernen Browsern unterstützt wird. Das Applet erlaubt das Auswählen beliebig vieler Fotos. Für jedes Foto kann das Format und die Anzahl der Fotos gewählt werden. Zusätzlich besteht die Option das Foto automatisch auf die Proportionen des gewählten Formats anzupassen.

Die zweite Möglichkeit ist ein HTML Formular zu verwenden. Dies bedient sich auch dem Prinzip des Albums als Container für Bilder. Die verfügbaren Optionen sind ident mit jenen der Java Applet Methode. Auffällig ist, dass sobald die Stückzahl für ein Foto verändert wird, der Benutzer explizit durch eine Nachrichtenbox nochmals dazu

aufgefordert wird dies zu bestätigen. Abbildung 4 zeigt das Happy Foto Bestellformular. [HappyFoto14]

Happy Foto bietet das Drucken von Digitalfotos auf bzw. für folgende Produkte an:

Fotodruck auf Papier	Spiele (Memory und Puzzle)
Fotobücher	Kleidung
Kalender	Taschen
Grußkarten	Etiketten
Tassen, Krüge und Flaschen	Dekorartikel
Poster	
Schmuck (Freundschaftsband)	
Fliesen	

The screenshot shows the HappyFoto website interface. At the top, there is a blue header with the HappyFoto logo, the slogan "Klick, click & happy.", and a search bar. Below the header is a navigation menu with categories like Fotobücher, Fotos, Fotogeschenke, Poster, Grußkarten, and Aktionen. A yellow banner indicates "Lieferzeiten zu Weihnachten". The main content area shows "Meine Alben" with a sub-section for "My First Album (1 Fotos)". On the left, there are links for "Online-Fotoservice", "Formate und Preise", "Foto-Bestellassistent", "HTML-Upload", "Ideal- oder Klassikformat?", "Auflösung und Dateiformat", "Fotos vom Datenträger", "Fotos vom Analog-Film", "Foto", "Sonderausarbeitungen", and "Premium-Qualität". On the right, there are buttons for "Fotos hinzufügen", "In den Warenkorb", and a section for "Format ändern:", "Anzahl ändern:", and "Zuschnitt ändern:". At the bottom, there are buttons for "Album umbenennen", "Album löschen", and "zur Albenübersicht". The footer contains contact information, services, and social media links.

Abbildung 4: Das Happy Foto Bestellformular

Die Folgende Tabelle zeigt ermittelte Merkmale und Funktionen der Bildbestellung:

Anzahl der Formate	4
Anzahl der Papiersorten	1
Rahmen verfügbar	Nein
Bildoptimierungsoption	Ja
Collagenoption	Nein
Geschenksoption	Nein
Import von Facebook	Nein
Import von Flickr	Nein
Import von Instagram	Nein
Kostenvorschau	Nein
Bedienungsassistent	Nein
Besonderheiten	Bildproportionen werden automatisch auf das gewählte Format angepasst.

Die Folgende Tabelle zeigt ermittelte Merkmale und Funktionen der Fotobuchbestellung:

Online Designer	Ja
Fotobuch Software	Ja
Unterstützte Systeme	Windows, iOS
Anzahl der Formate	23
Anzahl der Papiersorten	1
Seitenzahl minimal	16
Seitenzahl maximal	240
Einband	Softcover, Hardcover, Ringbindung
Collagenoption	Nein
Bedienungsassistent	Nein

Die Folgende Tabelle zeigt ermittelte Merkmale und Funktionen der Plattform im Allgemeinen:

Registrierung erforderlich	Ja
Willkommensbonus	Nein
Versandkostenfreie Lieferung	Nein
Versandkosten Basis	Versandkosten pauschal



Abholen im Shop – Funktion	Nein
Online bestellbar	Ja
Offline bestellbar	Ja, Datenaufbereitungsgebühr wird verrechnet
Transparenter Bestellvorgang	Ja
Bezahlmethoden	Kreditkarte, auf Rechnung
Kundenservice	Telefonisch und via E-Mail
Newsletter System	Ja
Partnerprogramm (bzw. Franchise)	Nein
Empfehlungen	Ja, von der Republik Österreich und diversen Instituten sowie Zeitschriften
Eingesetzte Technologien	HTML5, Java Applets, und Desktop Softwarepaket





### 3.5 Fotocom

Die Plattform Fotocom wird von der Firma FOTOCOM SA mit Hauptsitz in Louvain-la-Neuve, Belgien, betrieben. Das Unternehmen betreibt 15 Standorte weltweit und versendet in 186 Länder. Auf 8400 Quadratmeter verfügt Fotocom über eine Produktionskapazität von 3 Millionen Fotos pro Tag. [Foto14]

Das Bestellformular von Fotocom wurde auf zwei separate Masken verteilt: Die erste Maske erlaubt dem Benutzer das hinzufügen und hochladen von Bildern. Dazu steht eine Reihe von Möglichkeiten zur Auswahl. Ein sogenannter Komfort-Datentransfer verwendet ein Java Applet für die Übertragung. Ein weiterer Transfermodus verwendet ein Adobe Flash Formular. Die Flash-Methode ist dabei jene die als Standard-Methode verwendet wird. Fotocom bietet des Weiteren den Import von Fotos aus Facebook, Flickr, Picasa und Instagram. Die zweite Maske erlaubt dem Kunden für jedes hochgeladene Foto die Ausarbeitungsoptionen zu treffen. Zur Verfügung stehen verschieden Papiersorten und Formate, sowie Bildoptimierungs-Optionen. Abbildung 5 zeigt die Maske mit Ausarbeitungsoptionen von Fotocom. [ Foto14]

Fotoabzüge

Fotos von Online-Alben übertragen :

bilder hinzufügen    

**Grundeinstellungen für alle Fotos :**

Bezeichnung der Bestellung

Wird auf die Rückseite der Fotos gedruckt.  
[Beispiel](#)

Automatische Bildkorrektur

Ja  
 (dringend empfohlen)

Nein  
 (nur für Profis & individuell bearbeitete Digitalfotos)

ANWENDEN

**Diese Einstellungen auf alle ausgewählten Fotos anwenden :**

Anzahl:  Format: 10 x 15 Autofill: Ausfüllen

Rand:

Papiersorte

Premium

Matt

Eco



Optionen auf alle Bilder anwenden



Auf ausgewählte Bilder angewendet werden sollen.

ANWENDEN



**WEITER**

2 Foto In Ihrem Warenkorb  
 Gesamt Foto-Anzahl : 2

Bild 1 bis 2  Grenze des bedruckten Bereichs  Alle Fotos dieser Seite löschen [Bestellung löschen](#)

2014-07-08.jpg	Anzahl	Format	Seitenverhältnis	Rand	Autofill	Papiersorte	Löschen
	<input type="text" value="1"/>	10 x 15	3/2 720/405	<input type="checkbox"/>	Ausfüllen	<input checked="" type="radio"/> Premium <input type="radio"/> Matt <input type="radio"/> Eco	
<a href="#">Voransicht prüfen</a> <a href="#">Format hinzufügen</a>							
Verwendete Datentransfermethode : aurigmaflash Qualität : Grenzwertig							

2-pig*.jpg	Anzahl	Format	Seitenverhältnis	Rand	Autofill	Papiersorte	Löschen
	<input type="text" value="1"/>	10 x 15	4/3 640/512	<input type="checkbox"/>	Ausfüllen	<input checked="" type="radio"/> Premium <input type="radio"/> Matt <input type="radio"/> Eco	
<a href="#">Voransicht prüfen</a> <a href="#">Format hinzufügen</a>							
Verwendete Datentransfermethode : aurigmaflash Qualität : Grenzwertig							

**WEITER**

Abbildung 5: Das Fotocom Bestellformular

Fotocom bietet das Drucken von Digitalfotos auf bzw. für folgende Produkte an:

Fotodruck auf Papier	Kleidung
Fotobücher	Taschen
Kalender	Dekorartikel
Grußkarten	
Tassen, Krüge und Flaschen	
Poster	
Leinwand	
Spiele (Plüschtiere und Puzzle)	

Die Folgende Tabelle zeigt ermittelte Merkmale und Funktionen der Bildbestellung.

Anzahl der Formate	16
Anzahl der Papiersorten	3
Rahmen verfügbar	Ja
Bildoptimierungsoption	Ja
Collagenoption	Ja
Geschenksoption	Nein
Import von Facebook	Ja
Import von Flickr	Ja
Import von Instagram	Ja
Kostenvorschau	Nein
Bedienungsassistent	Nein
Besonderheiten	Import von Picasa

Die Folgende Tabelle zeigt ermittelte Merkmale und Funktionen der Fotobuchbestellung:

Online Designer	Ja
Fotobuch Software	Nein
Unterstützte Systeme	Keine Angabe
Anzahl der Formate	24
Anzahl der Papiersorten	1
Seitenzahl minimal	1
Seitenzahl maximal	70
Einband	Softcover, Hardcover, Ringbindung

Collagenoption	Ja
Bedienungsassistent	Nein

Die folgende Tabelle zeigt ermittelte Merkmale und Funktionen der Plattform im Allgemeinen:

Registrierung erforderlich	Nein
Willkommensbonus	Ja, ausgewählte Produkte verbilligt
Versandkostenfreie Lieferung	Nein
Versandkosten Basis	Anzahl der bestellten Produkte
Abholen im Shop – Funktion	Nein
Online bestellbar	Ja
Offline bestellbar	Nein
Transparenter Bestellvorgang	Ja
Bezahlmethoden	Kreditkarte, Sofortüberweisung, weitere
Kundenservice	E-Mail bzw. Web-Formular
Newsletter System	Ja
Partnerprogramm (bzw. Franchise)	Nein
Empfehlungen	Nein
Eingesetzte Technologien	Flash und Java Applets

### 3.6 BIPA Foto Shop

Das Unternehmen BIPA ist ein 1980 gegründetes österreichisches Einzelhandelsunternehmen mit ca. 3.400 Mitarbeitern und einem Jahresumsatz von 544 Millionen Euro. Gegründet wurde BIPA von Karl Wlaschek. Die Abkürzung BIPA entstand in dem Wlaschek jeweils die ersten beiden Buchstaben seines Geschäftes „Billige Parfümerie“ in einem Wort zusammenfasste. Der BIPA Foto Shop ist eine Online Plattform auf der BIPA die Ausarbeitung von digitalen Fotos sowie Fotobücher anbietet. Die Plattform ist dabei keine individual Entwicklung sondern eine themenbasierte Anpassung der von CEWE angebotenen lizensierbaren CEWE Fotoplattform. [Bipa14] Diese Plattform wird auch aktuell von den österreichischen Unternehmen Interspar und Hartlauer genutzt.

Das CEWE Bestellformular ist einfach gehalten und kommt ganz ohne proprietäre Software aus. Es erlaubt das Hochladen von mehreren Fotos in einem Batch. Die

Ausarbeitungsoptionen sind äquivalent jener von anderen Anbietern. So kann Papierqualität, Bildformat und sogar eine Bildoptimierung sowie ein Rand für jedes auszuarbeitende Foto gewählt werden. Das Bestellformular verfügt über eine Preisvorschau und gibt Auskunft über die Qualität des zu druckenden Bildes.

Abbildung 6: Das BIPA Bestellformular

BIPA bietet das Drucken von Digitalfotos auf bzw. für folgende Produkte an:

Fotodruck auf Papier	Kleidung und Textilien
Fotobücher	Taschen
Kalender	Sticker
Grußkarten, Briefkarten, Klappkarten	Glas-Foto
Tassen und Krüge	Büro- und Schreibwaren
Poster	Hüllen für mobile Geräte
Leinwände	Dekorartikel
Spiele (Puzzle und Plüschtiere)	

Die Folgende Tabelle zeigt ermittelte Merkmale und Funktionen der Bildbestellung:

Anzahl der Formate	8
Anzahl der Papiersorten	2
Rahmen verfügbar	Ja
Bildoptimierungsoption	Ja
Collagenoption	Nein
Geschenksoption	Nein
Import von Facebook	Nein
Import von Flickr	Nein
Import von Instagram	Nein
Kostenvorschau	Ja
Bedienungsassistent	Nein

Die Folgende Tabelle zeigt ermittelte Merkmale und Funktionen der Fotobuchbestellung:

Online Designer	Nein
Fotobuch Software	Ja
Unterstützte Systeme	Windows, Mac, Linux
Anzahl der Formate	9
Anzahl der Papiersorten	5
Seitenzahl minimal	Keine Angabe
Seitenzahl maximal	Keine Angabe
Einband	Heft, Softcover, Hardcover, Leinenoptik, Premiumleinen, Leder

Collagenoption	Ja
Bedienungsassistent	Nein
Besonderheiten	Videos im Fotobuch durch QR-Code

Die folgende Tabelle zeigt ermittelte Merkmale und Funktionen der Plattform im Allgemeinen:

Registrierung erforderlich	Nein
Willkommensbonus	Nein
Versandkostenfreie Lieferung	Nein
Versandkosten Basis	Pauschal je nach Produkt
Abholen im Shop – Funktion	Nein
Online bestellbar	Ja
Offline bestellbar	Nein
Transparenter Bestellvorgang	Ja
Bezahlmethoden	Kreditkarte, auf Rechnung, Sofortüberweisung, Paypal
Kundenservice	Telefonisch
Newsletter System	Ja
Partnerprogramm (bzw. Franchise)	Nein
Empfehlungen	Nein
Eingesetzte Technologien	HTML5 und Desktop Softwarepaket

### 3.7 Plattformen Vergleichsmatrix

Abschließend sollen die Plattformen mit ihren Funktionen und Merkmalen in einer Matrix gegenübergestellt werden. Obwohl alle Plattformen größtenteils dasselbe Angebot an Drucksorten aufweisen, so gibt es doch einige Unterschiede und Alleinstellungsmerkmale. Die Besonderheiten der Plattformen werden folgend beschrieben.

Die Bedienung des Snapfish Bestellformulars überrascht mit einem Assistenten der den Kunden beim Bestellvorgang unterstützen soll. Keine der anderen Plattformen bietet diese Funktion. Snapfish ist auch die einzige der untersuchten Plattformen die auch die Ausarbeitung von Passfotos anbietet. Positiv zu erwähnen ist auch die Methode der Versandkostenermittlung. Diese wird auf Basis des Bestellwertes ermittelt, und bietet

auch die Möglichkeit der versandkostenfreien Bestellung, ab einem bestimmten Bestellwert.

Pixum erlaubt, im Vergleich zu den anderen Plattformen, auch eine Gestaltung von Fotobüchern auf Android Geräten. Neben dem BIPA Foto Shop werden für den Fotobuchdruck fünf verschiedene Papiersorten angeboten. Des Weiteren bietet die Plattform ein Partnerprogramm mit Provisionssystem.

Foto Charly überrascht mit der Möglichkeit, Pralinen Schachteln zu bedrucken, die auch mit Inhalt von der Firma Milka geliefert werden. Weiters differenziert sich die Plattform durch die Druckmöglichkeiten auf Holz sowie durch Anfertigung von Fotokristallen.

Happy Foto erlaubt Fotobücher von der Größe bis zu 240 Seiten, was die höchste Seitenzahl der untersuchten Plattformen ist. Neben Pixum ermöglicht diese Plattform auch die Erstellung von Fotobüchern auf iOS Systemen. Als einziger Anbieter ermöglicht Happy Foto das Bedrucken von Fliesen, Schmuck sowie Etiketten.

Auf Fotocom können Bilder in 16 verschiedenen Formaten auf drei unterschiedlichen Papiersorten bestellt werden. Keine der anderen Plattformen bietet diesen Umfang an. Als einziger Anbieter verfügt Fotocom über keine Desktop Fotobuchsoftware und bietet nur einen Online-Designer, der jedoch 24 verschiedene Fotobuch Formate bietet, so viel wie keiner der anderen Anbieter.

Der BIPA Foto Shop erlaubt seinen Kunden die Auswahl zwischen fünf verschiedenen Einbänden für Fotobücher. Die höchste ermittelte Anzahl. BIPA bzw. das CEWE System bietet überdies den Druck auf 15 verschiedenen Drucksorten an. Verglichen mit Pixum und Foto Charly mit jeweils 14, Happy Foto mit 13, Snapfish mit 12 und Fotocom mit 10 Drucksorten, die größte Auswahl.

Während der Analyse sind einige Schwierigkeiten mit der Bedienung der Bestellformulare aufgefallen. Diese Problematik wurde allerdings nicht bewertet oder in den Matrizen erwähnt. Hier soll jedoch darauf hingewiesen werden, dass neben der Beleuchtung der Merkmale und Funktionen, die einen rein statischen Eindruck liefern, auch ein Usability Test, zum Beispiel „Thinking Aloud Test“, der Plattformen von Interesse wäre.



Funktion / Merkmal	Snapfish	Pixum	Foto Charly	Happy Foto	Fotocom	BIPA Foto Shop
<b>Drucksorten</b>						
Fotodruck auf Papier	Ja	Ja	Ja	Ja	Ja	Ja
Fotobücher	Ja	Ja	Ja	Ja	Ja	Ja
Passfotos	Ja	Nein	Nein	Nein	Nein	Nein
Kalender	Ja	Ja	Ja	Ja	Ja	Ja
Grußkarten, Briefkarten, Klappkarten etc.	Ja	Ja	Ja	Ja	Ja	Ja
Tassen, Krüge, Flaschen;	Ja	Nein	Ja	Ja	Nein	Ja
Poster	Ja	Ja	Ja	Ja	Ja	Ja
Leinwände (mit/ohne Rahmen)	Ja	Ja	Ja	Nein	Ja	Ja
Spiele	Ja	Ja	Ja	Ja	Ja	Ja
Kleidung / Textilien	Ja	Ja	Ja	Ja	Ja	Ja
Taschen	Ja	Ja	Ja	Ja	Ja	Ja
Büroartikel (Notizblöcke, Briefpapier etc.)	Nein	Ja	Nein	Nein	Nein	Ja
Hüllen/Taschen für mobile Geräte	Nein	Ja	Nein	Nein	Nein	Ja
Klebefolien und Sticker	Nein	Ja	Nein	Nein	Nein	Ja
Glasbilder	Nein	Ja	Nein	Nein	Nein	Ja
Holzbilder	Nein	Nein	Ja	Nein	Nein	Nein
Fotokristalle	Nein	Nein	Ja	Nein	Nein	Nein
Dekorartikel	Ja	Ja	Ja	Ja	Ja	Ja
Lebensmittel	Nein	Nein	Ja	Nein	Nein	Nein
Schmuck	Nein	Nein	Nein	Ja	Nein	Nein
Fliesen	Nein	Nein	Nein	Ja	Nein	Nein
Etiketten	Nein	Nein	Nein	Ja	Nein	Nein

**Tabelle 1: Drucksorten Matrix**

Funktion / Merkmal	Snapfish	Pixum	Foto Charly	Happy Foto	Fotocom	BIPA Foto Shop
<b>Bildausarbeitung</b>						
Anzahl der Formate	5	6	5	4	16	8
Anzahl der Papiersorten	1	1	1	1	3	2
Rahmen verfügbar	Ja	Ja	Nein	Nein	Ja	Ja
Bildoptimierungsoption	Ja	Nein	Ja	Ja	Ja	Ja
Collagenoption	Ja	Nein	Nein	Nein	Ja	Nein
Geschenksoption	Nein	Nein	Nein	Nein	Nein	Nein
Import von Facebook	Ja	Nein	Nein	Nein	Ja	Nein
Import von Flickr	Ja	Nein	Nein	Nein	Ja	Nein
Import von Instagram	Nein	Nein	Nein	Nein	Ja	Nein
Kostenvorschau	Ja	Ja	Nein	Nein	Nein	Ja
Bedienungsassistent	Ja	Nein	Nein	Nein	Nein	Nein
<b>Fotobuchausarbeitung</b>						
Online Designer	Ja	Ja	Nein	Ja	Ja	Nein
Fotobuch Software	Ja	Ja	Ja	Ja	Nein	Ja
Unterstützte Systeme	Win, Mac, Linux;	Win, Mac, Linux, iOS, Droid;	Win, Mac, Linux	Win, iOS;	K.A.	Win, Mac, Linux;
Anzahl der Formate	9	8	5	23	24	9
Anzahl der Papiersorten	1	5	1	1	1	5
Seitenzahl minimal	K.A.	K.A.	16	16	1	K.A.
Seitenzahl maximal	K.A.	K.A.	100	240	70	K.A.
Einband	Soft, Hard, Leder, Leinen;	Soft, Hard, Leinen, Leder;	Soft, Hard, Heft;	Soft, Hard, Ring;	Soft, Hard, Ring;	Soft, Hard, Leinen, Leder, Heft;
Collagenoption	Ja	Nein	Nein	Nein	Ja	Ja
Bedienungsassistent	Nein	Nein	Nein	Nein	Nein	Nein

**Tabelle 2: Ausarbeitungs Matrix**

Funktion / Merkmal	Snapfish	Pixum	Foto Charly	Happy Foto	Fotocom	BIPA Foto Shop
<b>Plattform</b>						
Registrierung erforderlich	Ja	Nein	Nein	Ja	Nein	Nein
Willkommensbonus	Ja	Ja	Nein	Nein	Ja	Nein
Versandkostenfreie Lieferung	Ja	Nein	Nein	Nein	Nein	Nein
Versandkosten Basis	Bestellwert	Produkte, Attribute;	Abmessung d. Produkte	pauschal	Anz. Produkte	pauschal
Abholen im Shop – Funktion	Nein	Nein	Nein	Nein	Nein	Nein
Online bestellbar	Ja	Ja	Ja	Ja	Ja	Ja
Offline bestellbar	Nein	Nein	Nein	Ja	Nein	Nein
Transparenter Bestellvorgang	Ja	Ja	Ja	Ja	Ja	Ja
Bezahlmethoden	K,R,B,P;	K, R, B, P;	K, R;	K, R;	K, S;	K, R, S, P;
Kundenservice	E-Mail	E-Mail, Telefon;	E-Mail, Telefon;	E-Mail, Telefon;	E-Mail, Webform	Telefon
Newsletter - System	Ja	Ja	Ja	Ja	Ja	Ja
Partnerprogram	Nein	Ja	Nein	Nein	Nein	Nein
Empfehlungen	Nein	Ja	Ja	Ja	Nein	Nein
Eingesetzte Technologien	Flash, Desktop;	HTML5, Flash, Desktop;	HTML5, Desktop;	HTML5, Java, Desktop;	Flash, Java;	HTML5, Desktop;

Tabelle 3: Plattform Matrix

<b>Legende für Bezahlmethoden</b>	
K	Kreditkarte
R	auf Rechnung
B	Bankeinzug
P	Paypal
S	Sofortüberweisung

**Tabelle 4: Legende für Plattform Matrix**

## 4. Ausgewählte Technologien und Methoden moderner Webanwendungen

Die Analyse existierender Ausarbeitungsplattformen hat in Kapitel 2 gezeigt, dass Anbieter aus einer Vielzahl von vorhandenen Technologien schöpfen um den Kunden einen möglichst anwenderfreundlichen Umgang zu erlauben. In diesem Kapitel wird zunächst der wichtige Begriff der Technologie definiert und anschließend die Motivation hinter der Verwendung moderner Technologien erklärt. Es werden problematische Werkzeuge kurz erwähnt und anschließend ausgewählte moderne Ansätze gezeigt.

Zur Definition des Wortes Technologie sei hier erwähnt, dass es sich von dem zusammengesetzten griechischen Wort *technologia* ableitet. Darin enthalten ist das Wort *techne*, die Kunst bzw. das Handwerk, und *logos*, die Lehre oder die Wissenschaft. Die genauen Ursprünge des Wortes sind jedoch unbestimmt. Eine allgemein anerkannte Definition des Wortes existiert nicht. [Wikipedia1404] In dieser Arbeit wird der Begriff „Technologie“ als Synonym für ein technisches Werkzeug der Informationsverarbeitung verwendet.

Einige der, von den in Kapitel 2 untersuchten Anbietern, verwenden Technologien die nicht mehr zeitgemäß sind, oder deren zukünftige Entwicklung ungewiss ist.

Als Beispiel für eine Technologie die nicht mehr in Webanwendungen verwendet werden sollte, seien hier die Java Applets genannt. Diese bauen auf die Unterstützung des Browsers der NPAPI. NPAPI ist eine Abkürzung für Netscape Plugin Application Programming Interface eine Plugin Architektur die von Netscape in den 1990er Jahren entworfen wurde. Google ist bereits damit beschäftigt, die Unterstützung dieser API bis 2015 vollkommen aus dem Google Chrome Browser zu entfernen. [Wikipedia1405]

Eine kritische Technologie, die noch immer sehr weite Unterstützung findet, deren Zukunft aber ungewiss scheint, ist Adobe Flash. Flash ist eine Multimedia-Software Plattform zur Erstellung von reichhaltigen Internet-Anwendungen (Rich Internet Applications, RIAs). Ursprünglich von Macromedia entwickelt, erfolgt die Weiterentwicklung von der Firma Adobe. Durch die Standardisierung und die weite

Verbreitung von HTML5 wird die Funktionalität von Flash weitestgehend ersetzt. Aktuell bietet die Entwicklung von Flash Applikationen noch den Vorteil, dass viele Entwicklerwerkzeuge dafür existieren. HTML5 im Vergleich dazu, basiert auf Java Script und HTML, sowie CSS, also auf Technologien, die Basiswissen für Webentwickler darstellen. Das hat den Vorteil, dass Entwickler keine neuen Konzepte lernen müssen und mit vorhandenem Wissen weiter arbeiten können.

Von den 6 analysierten Plattformen in Kapitel 2, verwendet die Hälfte immer noch, zumindest zur Unterstützung, Adobe Flash. Zwei weitere Anbieter verwenden die nicht mehr vollständig unterstützte Java Applet Technologie.

Die Nachteile die dadurch entstehen sind vielfältig. Eine Gefahr die darin besteht ist, dass moderne Browser nur schlechte bis gar keine Unterstützung für veraltete Technologien bieten. Dies bedeutet das früher oder später der Betreiber der Plattform eine Veränderung bewirken muss und die Software entsprechend zu ändern hat. Einher geht damit sehr oft auch eine Veränderung in der Bedienung der Plattform. Funktional betrachtet ist dies nichts Schlimmes, jedoch bestrafen Benutzer diese Veränderungen oft indem nach alternativen Anbietern gesucht wird. Entscheidet sich der Betreiber aufgrund dieser Tatsache, sein aktuelles System nicht zu verändern, so wird es früher oder später passieren, dass die Plattform nicht mehr für jeden Benutzer zugänglich sein wird. Dies trifft im Besonderen auf neue Kunden, die mit der Zeit gehen zu. Ein Dilemma für Anbieter. Sowohl die Veränderung, als auch das Ausbleiben führt zu Schwierigkeiten.

Durch die Entscheidung für moderne Technologien und deren Einsatz die einen langen Lebenszyklus versprechen, kann zumindest wenn schon nicht vollkommen, dieses Problem hinausgezögert werden.

## **4.1 HTML5**

Der häufigste Beweggrund hinter der Entscheidung eine der erwähnten problematischen Technologien zu verwenden lag darin, dass schlicht bestimmte Funktionen von Browsern nicht unterstützt werden. Dies liegt in der geschichtlichen Entwicklung. HTML wurde 1993 von Time Berners-Lee, einem Engländer, am CERN in Genf, Schweiz, erfunden. Ursprünglich war es als Dokumenten-Auszeichnungssprache gedacht. Kurz darauf wurde eine grafische Benutzeroberfläche, der Browser, entwickelt

um HTML Dokumente zu visualisieren und die Navigation darin zu erleichtern. Zu diesem Zeitpunkt war der Hauptanwendungsfall die Präsentation statischer Information. Erst später fand die Dynamik Einzug durch das hinzufügen von Java Script Funktionalität in den Browsern und CGI Skripte am Server. Später kamen Server basierte Erweiterungen auf den Markt die es erlaubten, mittels eigener Sprachen, HTML Dokumente individuell für jeden Abruf zu erzeugen. [Wikipedia1406] Die frühen Jahre der Webentwicklung, bis ca. 2003, zeichneten sich dadurch aus, dass die meiste Aufmerksamkeit der Serverseite zukam. Durch die vermehrte Verwendung von asynchronen Funktionen der Browser, die bereits seit 1996 zur Verfügung standen, entwickelte sich um 2004 vermehrt der Trend, klassische Aufgaben vom Server weg und hin zum Client in den Browser zu verlagern. Dies revolutionierte die Webentwicklung. Die Technologie Ajax und das Konzept des Web 2.0 bildeten das Fundament dieser Revolution und formten zugleich die Grundlage für die spätere Entwicklung der zweiten und wohl größten Revolution im Internet, HTML5. [Wikipedia1407]

In den letzten 10 Jahren, seit 2004, fand eine Bewegung statt, die immer mehr klassische Desktop Applikation in Web Applikationen verlagerte. Das Web wurde vom Dokumenten-Browser zu einer Anwendungsplattform. Die rasante Entwicklung am Mobilfunkmarkt ermöglichte neue Geräte die über permanentes Internet verfügen und diesen Trend noch weiter verstärken. Das klassische Web transformierte sich in einen Daten Provider. Der Browser wurde in den letzten Jahren immer mehr zum Ersatz des Betriebssystems. [Reelsen11]

Um technologisch diese Entwicklung zu unterstützen wurde der HTML5 Standard entwickelt. Der Standard wird, wie schon der HTML4 Standard, vom World Wide Web Consortium (W3C) verwaltet und entwickelt. [W3C14]

Durch HTML5 werden nun Funktionen angeboten, die die Verwendung älterer Technologien, wie Flash oder Java Applets, überflüssig machen. Einige der wichtigsten Funktionen werden in den folgenden Abschnitten erwähnt und erklärt.

Eine sehr hilfreiche Webseite bei der Entwicklung von HTML5 Applikation ist canisue.com. Diese Seite bietet Informationen zur aktuellen unterstützten Funktionen von HTML5 in Browsern, und zeigt weitere Informationsquellen für jede Neuerung von HTML5 an. [Caniuse1402]

### 4.1.1 Das HTML5 Grundgerüst

Wie bereits erwähnt ist HTML eine Auszeichnungssprache. Durch sie lässt sich die Struktur eines Dokumentes beschreiben. Zum Einsatz kommen dabei sogenannte Tags. Diese können zusätzlich mit Attributen versehen werden um deren Semantik noch weiter zu spezifizieren. Ein Browser liest ein HTML Dokument aus einer externen Quelle, zum Beispiel einer Datei, oder einem entfernten Server, interpretiert die darin enthaltenen Anweisungen, ladet eventuelle externe Ressourcen die noch benötigt werden und stellt das Ergebnis grafisch am Bildschirm dar. Zur Erzeugung einer HTML5 Datei wird nur ein Texteditor benötigt. Das folgende Beispiel zeigt ein minimales Beispiel einer HTML5 Seite.

```
01: <!DOCTYPE html>
02: <meta charset=utf-8>
03: <title>Hallo HTML5!</title>
04: <p>Hallo Welt!
```

HTML5 erlaubt weiterhin die Auszeichnung durch das XHTML Format, welches strengere Markup Regeln dem Dokument zu Grunde legt, um die XML Konformität nicht zu verletzen. Das folgende Beispiel illustriert dies. [Förster11]

```
01: <?xml version="1.0" encoding="UTF-8"?>
02: <html xmlns="http://www.w3.org/1999/xhtml">
03: <head>
04:   <meta charset="utf-8"/>
05:   <title>Hallo HTML5!</title>
06: </head>
07: <body>
08:   <p>Hello Welt!</p>
09: </body>
10: </html>
```

Der Unterschied zwischen den beiden gezeigten Varianten liegt darin, dass die XHTML Variante ein explizites schließen der Tags erfordert, sowie die Verwendung von Anführungszeichen bei Attributwerten. Des Weiteren muss das Dokument vor der Übertragung von Servern mit einem speziellen Header versehen werden. [Förster11] Für Details zu XHTML5 siehe [W3C1402].



## 4.1.2 Das HTML5 Multi File Select Element

In herkömmlichen HTML Formularen, die das Hochladen von Dateien erlauben, war es bis HTML4 nur möglich, pro Upload-Formular Element eine einzige Datei hochzuladen. Diese Limitierung stellte einer der Hauptgründe für die Verwendung von proprietären Technologien dar. Will der Benutzer 20 oder 30 Dateien gleichzeitig hochladen, so muss jede einzelne ausgewählt werden. In HTML5 gibt es nun ein neues Element das diesen Umstand beseitigt. Das folgende Beispiel zeigt die Verwendung dieses neuen Elementes.

```
01: <!DOCTYPE html>
02: <html>
03: <head>
04:   <meta charset="UTF-8">
05:   <title>HTML5 multiple file upload</title>
06: </head>
07: <body>
08:   <form action="#" method="POST" accept-charset="UTF-8"
09:     enctype="multipart/form-data" autocomplete="off"
10:     novalidate>
11:     <label>File:<input type="file" name="myfiles"
12:       multiple="multiple"></label>
13:   </form>
14: </body>
15: </html>
```

Eine weitere Neuerung von HTML5 ist die automatische Validierung von Formularen. Durch hinzufügen des required Attributs bei einem input Element, stellt der Browser sicher, dass das Element vor dem Absenden mit Daten befüllt wurde. Ist dieses Verhalten nicht erwünscht, so kann es durch Angabe des novalidate Attributs im Form Element abgeschaltet werden. Im gezeigten Beispiel passiert dies in Zeile 8. Zeile 9 zeigt wie ein herkömmliches Datei Input Element in ein HTML5 Element, durch Angabe des multiple Attributs verwandelt werden kann. Wird dieses Attribut hinzugefügt, so kann der Benutzer mehrere Dateien zum Hochladen auswählen. Nachteilig ist allerdings dass keine Verzeichnisse gewählt werden können.

Das Multi File Select Element wird aktuell von den Browsern Internet Explorer, Firefox, Chrome, Opera, Safari und iOS Safari unterstützt. [Caniuse1403]

### 4.1.3 Die HTML5 File Reader Schnittstelle

Durch die neue File Reader Schnittstelle ist es nun erstmals möglich, Daten direkt von der Festplatte zu lesen, ohne dass ein Server notwendig ist. In herkömmlichen Applikationen musste zuerst vom Benutzer eine Datei bestimmt werden, diese anschließend zu einem entfernten Server übertragen und danach von diesem wieder runtergeladen werden. Nur so konnte eine Webseite auf lokale Daten zugreifen. Durch die File Reader Schnittstelle entfällt zwar nicht der Schritt, dass der Benutzer Dateien wählen muss, die gelesen werden dürfen, jedoch ist dafür kein Server mehr notwendig. Das folgende Beispiel zeigt das Einlesen lokaler Dateien durch Verwendung dieser neuen Schnittstelle.

```
01: <!DOCTYPE html>
02: <html>
03: <head>
04: <meta charset="UTF-8">
05: <title>HTML5 File Reader API</title>
06: </head>
07: <body>
08: <form action="#" method="POST" accept-charset="UTF-8"
    enctype="multipart/form-data" autocomplete="off"
    novalidate>
09:     <label>File:<input type="file" id="myfile"
    accept="text/plain"></label>
10: </form>
11: <div id="container"></div>
12: <script type="text/javascript">
13: document.getElementById('myfile').addEventListener('change',
    read, false);
14: function read(file)
15: {
16:     var reader = new FileReader();
17:     reader.onload = function (event)
18:     {
19:         document.getElementById('container').innerHTML =
            event.target.result;
20:     };
21:     reader.readAsText(file.target.files[0]);
22: }
23: </script>
24: </body>
25: </html>
```

Das Beispiel verfügt über ein Formular, das ein File Upload Element beinhaltet. In den Zeilen 13 bis 22 wird in JavaScript das Element mit einem Event Handler versehen. Der Handler ist eine JavaScript Funktion, welche zunächst ein File Reader Objekt erzeugt, Zeile 16, und das Einlesen der im Event referenzierten Datei beginnt, Zeile 21. Nach Abschluss des Lesevorgangs erfolgt ein callback des File Readers an die in Zeile 17 definierte anonyme Funktion. Diese stellt nach Abschluss des Lesevorgangs den Inhalt im Container Div Element dar. Da es für dieses Beispiel nur Sinnhaftig ist wenn Textdaten gelesen werden, wurde in Zeile 9 dem Upload Element eine zusätzliche Restriktion auferlegt durch das Attribut `accept`. Dieses erlaubt die Beschränkung der Auswahl auf Dateien mit textuellem Inhalt. Das Attribut ist jedoch mit Vorsicht einzusetzen, da nicht garantiert wird, dass die ausgewählte Datei wirklich nur Text beinhaltet.

In komplizierten Applikationen kann diese Schnittstelle eine enorme Erleichterung sein. Zum Beispiel in Situationen wo Daten noch lokal nachbearbeitet werden müssen bevor sie zum Server gesendet werden. Dadurch kann sich ein Plattformbetreiber wertvolle Serverressourcen ersparen, was den Betrieb kosteneffizienter gestaltet.

#### 4.1.4 Drag & Drop mit HTML5

Eine weitere Möglichkeit Dateien für eine Web-Applikation verfügbar zu machen ist die in HTML5 eingeführte Drag & Drop Funktion. Mittels dieser Funktionalität kann ein Benutzer nun Dateien auf seinem Computer auswählen und diese durch klicken und ziehen in das Applikationsfenster der Anwendung übergeben. Vereinfacht gesprochen, passiert dabei nichts anderes als beim Auswählen von Dateien durch das File-Upload Element. Einzig der Mechanismus der Auswahl ist ein anderer. Das folgende Beispiel zeigt wie Drag & Drop in HTML5 umgesetzt werden kann.

```
01: <!DOCTYPE html>
02: <html>
03: <head>
04:   <meta charset="UTF-8">
05:   <title>HTML5 Drag and Drop</title>
06: </head>
07: <body>
08:   <div id="dropZone" style="width:200px; height:200px;
09:     background-color: red;"></div>
10:   <div id="container" style="width:200; height:"></div>
11: <script type="text/javascript">
12: document.getElementById( 'dropZone' ).ondragover = function ( )
```

```

012: {
013:   this.style.backgroundColor = 'green';
014:   return false;
015: };
016: document.getElementById('dropZone').ondrop = function (e)
017: {
018:   e.preventDefault();
019:   this.style.backgroundColor = 'red';
020:   read(e.dataTransfer.files[0]);
021: }
022: function read(file)
023: {
024:   var reader = new FileReader();
025:   reader.onload = function (event)
026:   {
027:     document.getElementById('container').innerHTML =
       event.target.result;
028:   };
029:   reader.readAsText(file);
030: }
031: </script>
032: </body>
033: </html>

```

Das Beispiel zeigt in Zeile 8 und 9 die Erstellung zweier Div-Container. Der erste dient dabei als Ziel für Drag & Drop Aktionen des Benutzers. Der zweite dient der Darstellung des Inhalts der Datei die der Benutzer wählt. Zeile 11 und 16 implementiert die benötigten Event-Handler. Damit Drag & Drop für ein Element erst überhaupt ermöglicht wird, ist der Handler in Zeile 11 notwendig. Der Handler in Zeile 16 wird ausgelöst sobald der Benutzer eine Datei über den Ziel-Container fallen lässt. Dieser bezieht eine Referenz auf die erste Datei die fallen gelassen wurde, und startet die Funktion „read“, die daraufhin den Inhalt in bekannter Weise liest und diesen wieder ausgibt.

#### 4.1.5 HTML5 Canvas

Das HTML5 Canvas Element ist eines der interessantesten und mächtigsten Neuerungen die der HTML5 Standard hervorgebracht hat. Ursprünglich wurde es von Apple 2004 entwickelt um Desktop-Widgets zu implementieren und um die Funktionalität des Safari Browsers, ein Produkt von Apple, zu erweitern. Durch das Element kann eine Fläche in eine Webseite eingebettet werden, auf welche anschließend, durch eine Java Script API, gezeichnet werden kann. Unterschieden

werden dabei der 2D Kontext und der 3D Kontext. Letzter erlaubt die Realisierung komplexer 3D Applikation wie Spiele, CAD oder 3D Modellierungs Applikationen. Der 2D Kontext erlaubt die Erstellung von Rastergrafiken mithilfe von Java Script Funktionsaufrufen. [Rowell11]

Im Vergleich zu Flash, Silverlight oder SVG funktioniert das HTML5 Canvas Element auf Basis des immediate render mode. Das bedeutet, dass der gesamte Buffer des Elements, bei jedem Animation Frame, auf die Canvas Fläche gezeichnet wird. Es wird intern keine Liste von grafischen Objekten geführt. Für diese ist der Anwender selbst verantwortlich. [Fulton11]

Durch die Natur des 2D Kontext und die verfügbare Java Script API eignet sich das Element auch ideal zur Bildbearbeitung. Das folgende Beispiel zeigt die Verwendung eines HTML5 Canvas.

```
01: <!DOCTYPE html>
02: <html>
03: <head>
04:   <meta charset="UTF-8">
05:   <title>HTML5 Canvas</title>
06: </head>
07: <body>
08:   <div id="dropZone" style="width:200px; height:200px;
09:     background-color: red;"></div>
10:   <canvas id="myCanvas" width="400" height="400"
11:     style="border:1px solid black;"></canvas>
12: <script type="text/javascript">
13: document.getElementById('dropZone').ondragover = function ()
14: {
15:   this.style.backgroundColor = 'green';
16:   return false;
17: };
18: document.getElementById('dropZone').ondrop = function (e)
19: {
20:   e.preventDefault();
21:   this.style.backgroundColor = 'red';
22:   read(e.dataTransfer.files[0]);
23: }
24: function read(file)
25: {
26:   var reader = new FileReader();
27:   reader.onload = function (event)
28:   {
29:     var image = document.createElement('img');
30:     image.onload = function () {
```

```

29:     var context =
30:         document.getElementById('myCanvas').getContext('2d');
31:         context.drawImage(image, 0,0, image.width, image.height);
32:     };
33:     image.src = event.target.result;
34:     reader.readAsDataURL(file);
35: }
36: </script>
37: </body>
38: </html>

```

Das gezeigte Beispiel erweitert die bisherigen Beispiele um das Canvas Element. Nach dem Einlesevorgang des Bildes erfolgt im Event-Handler der Zeile 25 die Erzeugung eines image Elements. Als Bildquelle URL wird dem Element der erzeugte Daten URL des Events gegeben, Zeile 32. Nachdem die Daten des Bildes vom Browser vollständig in das image Element eingelesen wurden, wird das Bild anschließend auf der Canvas Fläche dargestellt, Zeile 30.

Eine der häufigsten Aufgaben eines Web Servers ist die Erzeugung von Thumbnails von großen Bilddateien. Dies ist ein Rechenintensiver Vorgang der viele Ressourcen des Servers in Anspruch nimmt. Durch das Canvas Element ist es nun möglich, solche Skalierungsoperationen vollständig am Client durchzuführen und nur mehr das fertige Ergebnis an den Server zu senden. Das folgende Beispiel zeigt wie eine solche Skalierung durchgeführt werden kann.

```

01: function read(file)
02: {
03:     var reader = new FileReader();
04:     reader.onload = function (event)
05:     {
06:         var image = document.createElement('img');
07:         image.onload = function (){
08:             var context =
09:                 document.getElementById('myCanvas').getContext('2d');
10:                 context.drawImage(image, 0,0, image.width, image.height,
11:                 0, 0, image.width * .5, image.height * .5);
12:             };
13:             image.src = event.target.result;
14:         };
15:         reader.readAsDataURL(file);
16:     }
17: }

```

Im Beispiel wird eine alternative Methode der read Funktion des vorangegangenen Beispiels verwendet. Hier wird nach dem Laden der Bilddaten eine Funktion des 2D Kontext aufgerufen die es erlaubt, einen bestimmten Bildausschnitt an eine bestimmte Position der Zeichenfläche zu kopieren und dabei die Zielgröße festzulegen, Zeile 9. Wird als Ausschnitt das gesamte Quellbild verwendet und als Zielgröße die ursprüngliche Dimension mit einem konstanten Faktor multipliziert und der Funktion übergeben, so erfolgt eine Skalierung. Das Beispiel führt eine gleichmäßige Skalierung durch. Soll das Bild auf eine Zieldimension skaliert werden, die vom ursprünglichen Bildverhältnis abweicht, so müssen natürlich weitere Berechnungen durchgeführt werden.

Um schließlich die Bilddaten der Zeichenfläche weiter verarbeiten zu können, um sie zum Beispiel an den Server zu senden, bietet das Canvas-Element eine entsprechende Funktion. Das folgende Beispiel zeigt wie der Zustand des internen Bild Buffers vom Kontext bezogen werden kann.

```
01: var dataURL =  
    document.getElementById( 'myCanvas' ).toDataURL( 'image/jpeg' ,1.0 );
```

Durch die Angabe eines MIME Typs kann der Funktion mitgeteilt werden, in welchem Format die Bildinformation kodiert sein soll. Zusätzlich kann ein Qualitätsparameter übergeben werden, der bestimmt wie gut das Bild aussehen soll. Die Angabe von 1 entspricht dabei dem höchsten Wert. Beachtet werden muss, dass je höher die Qualität, desto länger dauert die Skalierung und desto größer ist der erhaltene Data URL.

Die letzten drei Beispiele haben gezeigt wie eine Skalierungsoperation mit Hilfe des HTML5 Canvas Elements durchgeführt werden kann. Der im letzten Beispiel erhaltenen Daten URL kann an einen Server gesendet werden, der nur mehr eine Dekodierung durchführen muss, um das skalierte Bild zu erhalten.

#### **4.1.6 HTML5 Lokaler Speicher**

Die hauptsächliche Kommunikation von Web Anwendungen erfolgt auf Basis des HTTP Protokolls. Da HTTP ein Verbindungsorientiertes zustandsloses Protokoll ist, wurde der Cookie Mechanismus entwickelt, um Benutzer über mehrere Anfragen hinweg, ein Token zuzuweisen und so eine Identifizierung und Zuordnung der

eingehend Verbindungen durchführen zu können. Cookies stellten lange Zeit den einzigen Mechanismus dar um im Browser benutzerspezifische Daten zu speichern. Die Nachteile von Cookies sind deren stark begrenzte Kapazität und dass sie bei jeder Anfrage die vom Browser an den Server gesendet wird, im Anfragekopf mit zum Server übertragen werden. Das erhöht die Transferzeit und resultiert in einer höheren Latenz für den Benutzer, sowie einen erhöhten Aufwand für den Server. Durch HTML5 wurde es nun möglich Anwender spezifische Daten direkt im Browser zu speichern. Ermöglicht wird dies durch die Local Storage API. Diese Schnittstelle erlaubt das beziehen eines Storage-Objekts in welches Daten für eine bestimmte Lebensdauer gespeichert werden können. Das folgende Beispiel zeigt die Verwendung der Storage API.

```
01: <!DOCTYPE html>
02: <html>
03: <head>
04:   <meta charset="UTF-8">
05:   <title>HTML5 Local Storage</title>
06: </head>
07: <body>
08:   <textarea id="textData" style="width:200px;
    height:200px;"></textarea>
09:   <button id="save" type="button">Save</button>
10:   <button id="load" type="button">Load</button>
11:   <script type="text/javascript">
12:     function save()
13:     {
14:       var storage = window['localStorage'];
15:       var content = document.querySelector('#textData').value;
16:       storage.setItem('data', content);
17:     }
18:     function load()
19:     {
20:       var storage = window['localStorage'];
21:       var content = storage.getItem('data');
22:       document.querySelector('#textData').value = content;
23:     }
24:     document.querySelector('#save').addEventListener("click", save);
25:     document.querySelector('#load').addEventListener("click", load);
26:   </script>
27: </body>
28: </html>
```



Im Beispiel wird ein Text Eingabefeld und 2 Knöpfe erzeugt, die das Speichern und Laden des Inhalts des Feldes erlauben. Wird auf den Save-Knopf geklickt so wird im entsprechenden Event Handler, Zeile 12, vom Browser das Storage Objekt bezogen und der Inhalt des Textfeldes darin gespeichert. Beim Bezug des Objekts wird dem Browser dabei mitgeteilt, dass es sich um ein local Storage Objekt handeln soll. Alternativ könnte auch ein session-Storage-Objekt bezogen werden. Der Unterschied zwischen local- und session-Storage liegt in der Speicherdauer und in der Sichtbarkeit der Daten. Bei einem local-Storage Objekt werden die Daten über die aktuelle Session hinausgespeichert und jedes Tab oder Fenster derselben Domain, das im Browser geöffnet ist, kann auf den Inhalt des Objekts zugreifen. Bei einem session-Storage-Objekt hingegen werden die Daten nur für das aktuelle Tab gespeichert und beim Schließen desselben, wieder gelöscht. [Lubbers11]

Auf diese Weise lassen sich einige Probleme lösen. Zum Beispiel, wenn Applikationen dem Kunden ermöglichen, in einem Textfeld einen Beitrag zu verfassen, um diesen auf einer Webseite zu veröffentlichen. So kann es passieren, dass während der Benutzer noch mit dem Verfassen des Beitrags beschäftigt ist, die ihm zugeordnete Session verfällt, und beim Absenden des Textes der Server die Annahme verweigert. Viele Benutzer ergreift in solchen Momenten die Panik und es wird versucht mittels des Zurück Knopfes im Browser wieder auf die Seite mit dem in mühevoller Arbeit verfassten Beitrag zu kommen. Abhängig davon wie die Applikation implementiert ist kann es passieren dass schließlich der Beitrag verschwunden ist. Mit Hilfe der Storage API können solche Szenarien vermieden werden. Der Text wird in regelmäßigen Abständen in ein session Objekt gespeichert.

Browser-Hersteller sind dazu angehalten, die Schnittstelle bezüglich ihrer Sicherheit möglichst wasserdicht zu gestalten. So ist es zum Beispiel nicht möglich, dass ein Skript das von einem anderen Server geladen wurde, auf die lokal gespeicherten Daten einer Anwendung zugreifen kann. Ermöglicht wird dies durch die „same-origin-policy“, ein Sicherheitskonzept, das schon in früheren Versionen von HTML Anwendung fand.

#### **4.1.7 AJAX 2.0**

Wie schon zu Beginn in der Geschichte erwähnt, war AJAX und das Aufkommen Sozialer Interaktions Plattformen, für den Trend der Web 2.0 Anwendungen verantwortlich. Diese Technologie wurde ursprünglich von Microsoft, gegen Ende der

90er Jahre entwickelt. AJAX ist die Abkürzung von asynchronous JavaScript and XML und beschreibt dabei eine Methode, Daten zwischen einer Web Anwendung und deren Server auszutauschen, ohne dass der Benutzer davon etwas bemerkt. Der Name mag etwas irreführend sein, da nicht nur XML Daten sondern jedes beliebige Datenformat für die Kommunikation verwendet werden kann. So hat es sich mittlerweile etabliert, dass vorwiegend JSON als Datenformat, für diese Art der Kommunikation zum Einsatz kommt. JSON ist eine Abkürzung für JavaScript Simple Object Notation. Die XMLHttpRequest Schnittstelle erlaubt das Anwenden der AJAX Methode. Der Browser stellt dafür ein Objekt vom Typ XMLHttpRequest zur Verfügung. HTML5 hat an diesem Sachverhalt nichts verändert, jedoch einige wichtige Methoden der Schnittstelle hinzugefügt. So ist es mit der erweiterten Schnittstelle nun auch möglich Dateien zu übertragen, Übertragungsinformationen zu erhalten und Formular-Daten einfach zu senden. Bisherige Applikationen, die auf die Verwendung des XMLHttpRequest Objektes bauen, müssen nicht verändert werden und funktionieren weiterhin. Für diese ist die Funktionserweiterung komplett transparent. Im folgenden Beispiel wird gezeigt, wie Dateien durch die neue Schnittstelle auf einen Server hochgeladen und wie Informationen über den Fortschritt verfügbar gemacht werden können.

Das Beispiel basiert auf dem im Abschnitt für Drag & Drop gezeigten Code für das Einlesen von Dateien. Dafür wird zunächst die Funktion des read abgeändert.

```
01:  function read(file)
02:  {
03:      var reader = new FileReader();
04:      reader.onload = function (event)
05:      {
06:          var bytes = new Uint8Array(event.target.result);
07:          var blob = new Blob([bytes], {type: 'application/octet-
           stream'});
08:          upload(blob);
09:      };
10:      reader.readAsArrayBuffer(file);
11:  }
```

Das Einlesen der Binärdatei erfolgt weiterhin über einen File-Reader. Als Zielformat wird diesmal jedoch ein Array-Buffer verlangt. Dieser wird in Zeile 6 in ein unsigned 8 bit integer Array konvertiert und damit anschließend in Zeile 7 ein Blob

Objekt erstellt. „Blob“ ist die Abkürzung für Binary Large Object. Es dient hier als Container für die eingelesenen Daten. Durch die Angabe des Type-Attributs wird dem Inhalt des Blob Objekts noch ein Typ zugewiesen. Diese Metainformation wird später noch benötigt. In Zeile 8 wird schließlich die Funktion für das Hochladen aufgerufen die im folgenden Beispiel gezeigt wird.

```
01:  function upload(blob, chunkSize, start)
02:  {
03:      if (typeof chunkSize === "undefined") chunkSize = 1024;
04:      if (typeof start === "undefined") start = 0;
05:
06:      var limit = chunkSize * (start + 1);
07:      limit = limit < blob.size ? chunkSize * (start + 1) :
          blob.size;
08:      var smallBlob = blob.slice(chunkSize * start, limit,
          blob.type);
09:
10:      var xhr = new XMLHttpRequest();
11:      xhr.open('POST', '/test', true);
12:
13:      xhr.onreadystatechange = function (evt)
14:      {
15:          if (xhr.readyState == 4 && xhr.status == 200) if (limit <
16:              blob.size) upload(blob, chunkSize, start + 1);
17:      };
18:
19:      xhr.addEventListener("progress", function (evt)
20:      {
21:          console.log("bytes sent so far: " + evt.loaded);
22:      });
23:
24:      var formData = new FormData();
25:      formData.append('binaryData', smallBlob);
26:      xhr.send(formData);
27:  }
```

Die Funktion upload empfängt drei Argumente. Ein Blob-Objekt das die zu sendenden Daten beinhaltet. Das optionale chunk-size-Argument definiert wie groß maximal ein Datenpaket ist das zum Server gesendet wird und das optional-start-Argument definiert wie viele Datenpakete bereits gesendet wurden. Die Funktion schickt nicht den gesamten Blob-Inhalt auf einmal zum Server sondern lädt die Daten in Paketen hoch. In den Zeilen 5 - 7 wird zunächst berechnet welche Daten im aktuellen Durchlauf zum Server hochgeladen werden sollen. In den Zeilen

8 und 9 wird ein Kommunikationsobjekt instanziiert und in der Zeile 20 wird die Anfrage an den Server abgesetzt. Zeile 18 und 19 zeigen die Verwendung des neuen HTML5 Formular Daten Element. Dies kann neben normalen Textwerten auch mit einem Blob Container befüllt werden. Nachdem das aktuelle Datenpaket erfolgreich zum Server geladen wurde, wird in Zeile 12 ein rekursiver Aufruf durchgeführt und mit dem nächsten Paket fortgefahren, sofern nicht bereits alle Pakete, daher der gesamt Inhalt des Blobs, übertragen wurde. Die Zeilen 14 - 17 zeigen wie der Fortschritt des aktuellen Sendevorgangs beobachtet werden kann. Die anonyme Funktion wird in Abständen vom Browser aufgerufen und bekommt als Argument ein Event welches nach der Anzahl der übertragenen Bytes gefragt werden kann.

Im Abschnitt über das HTML5 Canva Element wurde ein Beispiel gezeigt das die Skalierung eines Bildes ermöglicht. In Kombination mit der neuen Schnittstelle für AJAX 2.0 kann nun der Inhalt eines Canvas nach der Skalierung an den Server gesendet werden. Das folgende Beispiel zeigt dies.

```
01:  function upload()
02:  {
03:      var data =
          document.getElementById('myCanvas')
          .toDataURL('image/jpeg', 1.0);
04:      var bytes = new Uint8Array(data.length);
05:      for (var i = 0; i < data.length; i++) bytes[i] =
          data.charCodeAt(i);
06:      var blob = new Blob([bytes], {type: 'application/octet-
          stream'});
07:      var formData = new FormData();
08:      formData.append('binaryData', blob);
09:      var xhr = new XMLHttpRequest();
10:      xhr.open('POST', '/test', true);
11:      xhr.onreadystatechange = function (evt)
12:      {
13:          // to be implemented ...
14:      };
15:      xhr.send(formData);
16:  }
```

Nachdem in Zeile 3 der Inhalt des 2D Kontext als URL in der data Variable gespeichert wird, folgt die Konvertierung in einen Blob in den Zeilen 4 bis 6. Anschließend wird der Blob noch in ein Form-Data-Element verpackt, Zeile 7 - 8, und in Zeile 15 zum Server gesendet.

Das neue XMLHttpRequest-Objekt erleichtert die Implementierung datenintensiver Anwendungen sehr. Nun muss nicht mehr mit versteckten iframe Elementen gearbeitet werden, um Dateien im Hintergrund hochzuladen. Dies vereinfacht nicht nur den Client-Code sondern auch die Implementierung der Gegenstelle am Server.

### 4.1.8 Server Sent Events

Moderne Web Applikationen sind sehr datenlastige Applikationen. Die Daten, die diese Applikationen benötigen, kommen größtenteils von Servern, mit welchen im Hintergrund durch Einsatz des vorgestellten AJAX Konzept, kommuniziert wird. Ein Nachteil des AJAX 1.0 Konzept war, dass Anfragen immer vom Browser aus gestartet werden mussten. Es entstand das Konzept des Long-Polling. Durch die Server Sent Events Funktion ist es nun möglich, eine Verbindung zum entfernten Server aufzubauen und diese für die Dauer der Benutzer-Session offen zu lassen. Der Server kann, über die Verbindung, Daten zum Client senden, worauf in diesem ein Event für jeden Sendevorgang ausgelöst wird. Diese Methode ist vergleichbar mit einem Messaging Prinzip. Bei diesem Prinzip gibt es einen Verbraucher oder Beobachter, der sich bei einer Datenerzeugerstelle als Interessent registriert. Die Erzeugerstelle nimmt den Verbraucher in die Liste der Interessenten auf. Entstehen im Erzeugersystem Nachrichten, seien diese selbst erzeugt oder von einem dritten System erhalten, so werden sie an alle registrierten Verbraucher gesendet. Dieses Prinzip ist vergleichbar mit dem Observer-Pattern. [Gamma14] Das folgende Beispiel zeigt wie ein Client sich am Server für das Empfangen von Events anmelden kann.

```
01: <!DOCTYPE html>
02: <html>
03: <head lang="en">
04:   <meta charset="UTF-8">
05:   <title>HTML5 Server Sent Events</title>
06: </head>
07: <body>
08:   <div id="container"><p></p></div>
09:   <script type="text/javascript">
10:     var source = new EventSource("/events");
11:     source.addEventListener("message", function (event)
12:     {
13:       document.querySelector("#container").firstChild.textContent +=
         event.data + "\n";
14:     });
```

```
15: </script>
16: </body>
17: </html>
```

In diesem Beispiel wird in Zeile 10 ein Event-Source-Objekt erzeugt und dabei im Konstruktor die Adresse des Nachrichtenerzeugers mitgegeben. Danach wird ein Event-Handler am Objekt registriert und bei jeder eingehenden Nachricht aufgerufen. Sobald eine Nachricht eintrifft, wird diese im angegebenen Div Container angezeigt.

Auf der Server-Seite sind einige Dinge zu beachten:

- Die Antwort des Servers muss den Header Content-Type enthalten, dessen Wert auf text/event-stream gesetzt ist.
- Auch der Header Cache-Control muss in der Server Antwort vorhanden sein mit dem Wert no-cache
- Des Weiteren sollte die Antwort Kodierung im Format UTF-8 sein und dies dem Client auch mit entsprechenden Header mitgeteilt werden.

Nachrichten können danach mit folgender Formatierung gesendet werden:

```
01: data: {time: '123456'}\n\n
```

Hier wird ein JSON Objekt als Event versendet. Wichtig dabei ist, das Prefix-data muss enthalten sein und danach zweimal ein Zeilenvorschub. Idealerweise sendet der Server Event Nachrichten im JSON Format, da diese im Browser sehr leicht dekodiert werden können. Aber auch jedes andere Datenformat wäre denkbar, so auch binär codierte Daten.

## 4.2 Client und Server Entwicklung mit Java

Im TIOBE Ranking der beliebtesten Programmiersprachen ist Java seit Jahren unter den Top 2 vertreten. Nur die low Level Sprache C kommt im Ranking noch vor Java. [Tiobe14]

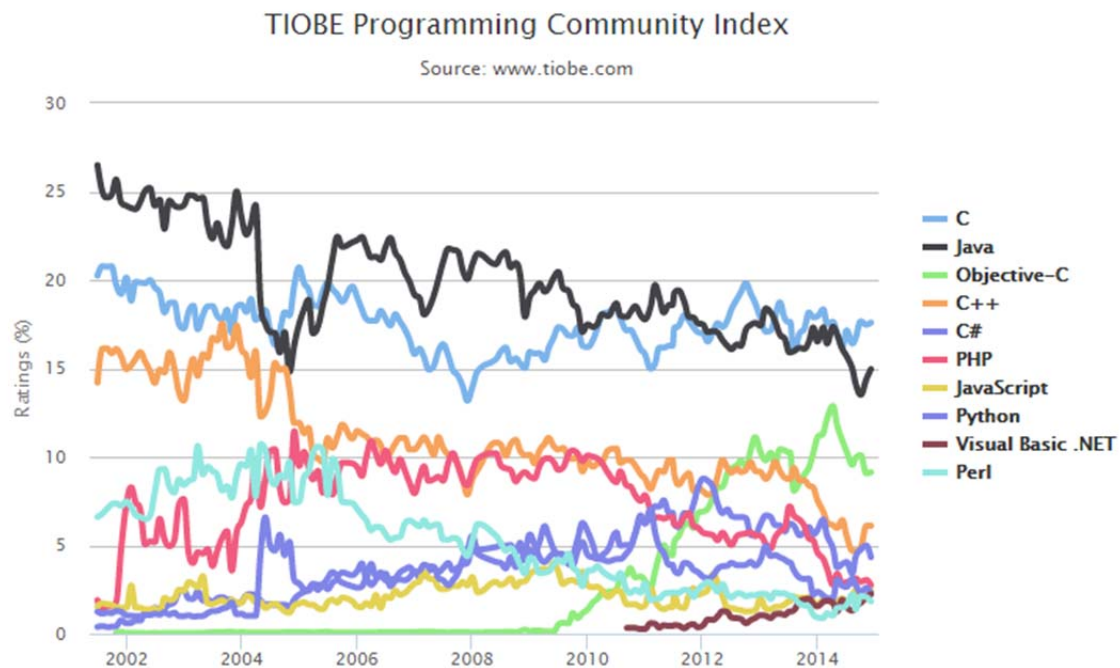


Abbildung 7: TIOBE Index der populärsten Programmiersprachen (Quelle [Tiobe14])

Das TIOBE Ranking wird auf Basis der populärsten Programmiersprachen erstellt. Einmal monatlich wird der Index auf Basis gemessener Metriken aktualisiert. Dabei werden Quellen wie Google, Bing, Amazon, Yahoo, YouTube und Baidu zur Berechnung herangezogen. Das Ranking gibt jedoch nur Aufschluss über die Popularität und nicht darüber was die beste Programmiersprache ist. [Tiobe14] Ob eine Programmiersprache gut oder schlecht ist, hängt natürlich von den Anforderungen der Aufgabenstellung und der Eignung der Sprache zur Lösung dieser abhängig von der gebotenen Funktionalität.

Für die Entwicklung von Web Applikation ist Java nach PHP und ASP.NET die drittbeliebteste Sprache. [W3tech14] Viele Content Management Systeme wie

WordPress, Joomla, Drupal oder Magento, welche auch die beliebtesten Systeme im Internet sind [W3tech14], sind mit der Programmiersprache PHP implementiert. Dieser Umstand schlägt sich durch die angewandte Messmethode im Index nieder.

Programmiersprachen, über einen längeren Zeitraum betrachtet, verhalten sich wie ein Ökosystem. Neue Sprachen entstehen und alte sterben aus, sofern sich diese nicht weiterentwickeln. Java hat sich in den letzten 15 Jahren ständig weiterentwickelt. Mit der am 18. März 2014 erschienen Version, Java 8, wurden die größten Veränderungen der Sprache veröffentlicht. Die größten Änderungen spiegeln einen Paradigmenwechsel, weg von der Objektorientierung hin zur funktionalen Programmierung. [Urma14] Im Folgenden sollen die wichtigsten Neuerungen vorgestellt werden die auch bei der Entwicklung von Web Applikationen angewendet werden können.

#### **4.2.1 Anonyme Funktionen**

Eine große Herausforderung in der Softwareentwicklung stellen sich ändernde Anforderungen dar. Behavior parameterization ist ein Entwurfsmuster das helfen soll sich ändernde Anforderungen leichter zu bewältigen. Es empfiehlt die Erstellung von Wiederverwendbaren Code Blöcken die dazu verwendet werden können um das Verhalten von Funktionen zu modifizieren. Das Verhalten (Behavior) wird in solche Code Blöcke gekapselt. Um dieses Entwurfsmuster in Java umzusetzen bot die Programmiersprache bisher nur ungenügend Unterstützung durch anonyme Klassen. Mit Java 8 wurden Lambda Ausdrücke eingeführt. Das sind anonyme Funktionen die die Kapselung von Code erlauben. Sie zeichnen sich durch folgende Merkmale aus: [Urma14]

- Anonym. Lambda Ausdrücke besitzen keinen Namen wie herkömmliche Funktionen oder Methoden.
- Funktion. Lambda Ausdrücke sind Funktionen die, zum Unterschied von Methoden, keiner Klasse oder Objekt angehören.
- Lambda Ausdrücke sind first class Funktionen. Das bedeutet sie können als Argument an Methoden übergeben oder als Ergebnis von Methoden zurückgegeben und in Datenstrukturen gespeichert werden.
- Prägnant. Lambda Ausdrücke benötigen weniger Code als anonyme Klassen

Im Folgenden wird anhand eines Beispiels die Behavior parameterization mittels Java Lambda Ausdrücke gezeigt.



```

01: public static void main(String[] args)
02: {
03:     final List<Picture> pictures = getPictures();
04:     final Comparator<Picture> byResolution = (p1, p2) ->
        (p1.getWidht() * p1.getHeight()) - (p2.getWidht() *
        p2.getHeight());
05:     pictures.sort(byResolution);
06:     for (final Picture picture : pictures)
07:     {
08:         System.out.println(picture.getWidht() + "x" +
        picture.getHeight());
09:     }
10: }

```

Das Beispiel zeigt die Sortierung einer Liste von Bildern anhand ihrer Auflösung. In Zeile 4 wird ein Lambda Ausdruck erzeugt der das Verhalten des Sortieralgorithmus kapselt. Die Struktur eines Lambda Ausdrucks folgt dabei immer folgendem Schema.

```

01: ( Lambda Parameter ) -> Lambda Körper

```

Die Sprachdesigner von Java haben sich für diese Syntax entschieden da sie in anderen Sprachen wie C# und Scala sehr gut angenommen wurde. Lambda Ausdrücke können nur im Kontext eines functional Interface verwendet werden. So ein Interface besteht aus nur einer abstrakten Methode. Die Signatur der Methode definiert dabei zugleich die Signatur des Lambda Ausdrucks und wird function descriptor genannt. Der Ausdruck implementiert das funktionale Interface. Der Typ des Lambda Ausdrucks wird dabei vom Kontext abgeleitet, in dem dieser verwendet wird, und als target type bezeichnet. Das Konzept nennt sich „target typing“. [Urma14]

Im oben gezeigten Beispiel wurde der Typ der Parameter nicht explizit angegeben. Dies erfolgte implizit durch den Compiler unter Anwendung von type inference, Typ Deduktion. Dazu hat der Compiler zunächst den target type des Ausdrucks bestimmt und dadurch auch das function Interface, wodurch auch die Parameter Typen des Lamba Ausdrucks bekannt sind. Aus Gründen der Leserlichkeit ist die Angabe von Typen aber optional und obliegt der Entscheidung der Entwickler. [Urma14]

## 4.2.2 Methoden Referenzen

Durch die Verwendung von Lambda Ausdrücken ist es möglich, Verhalten zu kapseln, was Wiederverwendbarkeit von Code erleichtert und die Lesbarkeit im Allgemeinen erhöht. Referenzen auf Methoden erlauben es bestehendem Code, der nicht explizit für die neue Java 8 Plattform geschrieben wurde, als Lambda Ausdruck zu verwenden. Darüber hinaus wird durch die Verwendung solcher Referenzen eine noch kompaktere Schreibweise erzielt. Grundsätzlich werden drei Arten der Methoden Referenz unterschieden. Eine Referenz auf: [Urma14]

- eine statische Methode einer Klasse, wie zum Beispiel `Integer::parseInt`
- eine instanz Methode eines beliebigen Typs, wie zum Beispiel `String::length`
- eine Instanz Methode eines existieren Objekts, wie zum Beispiel `myObject::myMethod`

In Anlehnung an das vorhergehende Beispiel soll die Verwendung von Methoden Referenzen in einem erweiterten Beispiel gezeigt werden.

```
01:  public static void main(String[] args)
02:  {
03:      final List<Picture> pictures = getPictures();
04:      pictures.sort(Sorter::byResolution);
05:      for (final Picture picture : pictures)
06:      {
07:          System.out.println(picture.getWidth() + "x" +
picture.getHeight());
08:      }
09:  }
10:
11:  public class Sorter
12:  {
13:      static public int byResolution(Picture p1, Picture p2)
14:      {
15:          return (p1.getWidth() * p1.getHeight()) - (p2.getWidth() *
p2.getHeight());
16:      }
17:  }
```

Dieses Beispiel zeigt die Sortierung einer Liste von Bildern anhand ihrer Auflösung durch die Verwendung einer Methoden Referenz auf die statische Methode der Sorter

Klasse. In Zeile 4 erfolgt der Aufruf und Übergabe der Methoden Referenz an die Sortiermethode der Liste.

Durch Lambda Ausdrücke und Methoden Referenzen kann das Fabrik Entwurfsmuster [Gamma14] sehr leicht implementiert werden. Das Folgende Beispiel zeigt die Erzeugung einer Fabrik für Bild Objekte.

```
01: Supplier<Picture> factory = Picture::new;
```

Das Supplier Interface ist eines der von Java 8 gelieferten functional interfaces. Wie bereits beschrieben, besitzt dieses nur eine abstrakte Methode, die durch die Angabe des Lambda Ausdrucks implementiert wird.

### 4.2.3 Collections und die Java Stream Processing API

Die Java Collection API ist die am meisten verwandteste aller Java Schnittstellen. Mit Collections ist es möglich Daten zu gruppieren und zu verarbeiten. Damit geht einher dass das Arbeiten mit dieser API und deren Komponenten ein wichtiger Stellenwert zukommt. In Java 8 wurde der Verarbeitung von Collections daher hohe Aufmerksamkeit gewidmet und die Möglichkeit der Streambasierten Verarbeitung von Elementen aus Collections eingeführt. Durch Streams ist die Verarbeitung von Daten einer Collection auf deklarative Weise möglich. Das bedeutet, dass eine Serie von Kommandos in Form einer Kette definiert wird, die beschreibt was mit den Daten passieren soll im Gegensatz zur Implementierung der Verarbeitungsschritte und iterativen Abarbeitung. [Urma14]

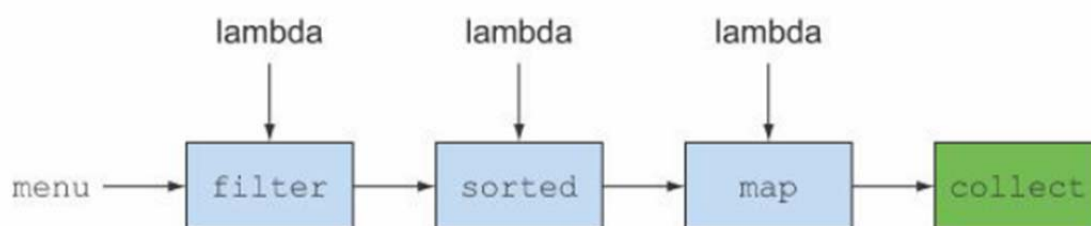


Abbildung 8: Deklarative Verarbeitung von Daten durch Streams und Lambda Ausdrücken. Quelle [Urma14]

Ein Stream ist eine Sequenz von Elementen aus einer Quelle und ermöglicht die Verarbeitung dieser Elemente. Wichtig sind 2 Charakteristiken der Stream basierten

Verarbeitung: Pipelining und interne Iteration. Pipelining beschreibt die Möglichkeit, dass einige Operationen des Streams wiederum einen Stream als Rückgabewert liefern, was eine Verkettung und lazy evaluation erlaubt. Durch interne Iteration werden die Elemente einer Collection in der Collection selbst traversiert im Kontrast zu externer Iteration mittels eines Iterators. [Urma14]

Im Folgenden werden Beispiele gezeigt und besprochen die das Arbeiten mit den neuen Java Streams für Collections demonstrieren.

Sollen bestimmte Elemente aus einer Collection gefiltert werden, kann die Filtermethode verwendet werden:

```
01: // filter (119)
02: List<Picture> pictures = getPictures();
03: pictures.stream()
04:     .filter(Picture::isRgbMode)
05:     .collect(toList());
```

Die Filter Methode kann mit einem Lambda Ausdruck oder einer Methoden Referenz parametrisiert werden. In diesem Beispiel werden alle Bilder die das RGB Farbmodell unterstützen in die Ergebnisliste aufgenommen. Um einzigartige Elemente einer Collection zu finden kann die Methode distinct verwendet werden:

```
01: // distinct (120)
02: pictures = getPictures();
03: pictures.stream()
04:     .filter(p -> p.getResolution() > 1000000)
05:     .distinct()
06:     .forEach(System.out::println);
```

In diesem Beispiel wird zu nächst jedes Bild, das weniger als ein Megapixel Auflösung hat, und danach jedes doppelt vorhandene, aussortiert. Jedes verbleibende Bild wird auf der Konsole ausgegeben. Durch die Verwendung der Limitmethode kann der Stream auf eine gewünschte Länge gekürzt werden:

```
01: // limit (120)
02: pictures = getPictures();
03: pictures = pictures.stream()
04:     .filter(p -> p.getResolution() > 1000000)
```

```
05:     .limit(3)
06:     .collect(toList());
```

Im vorangegangenen Beispiel werden maximal drei Bilder die eine Auflösung von mehr als einem Megapixel haben, mittels collect in eine Ergebnisliste gesammelt. Soll der Beginn eines Streams übersprungen werden, so bietet sich hierfür die Skipmethode an:

```
01:     // skip (121)
02:     pictures = getPictures();
03:     pictures = pictures.stream()
04:         .skip(2)
05:         .collect(toList());
```

Die Skipmethode erwartet nur einen Parameter, der bestimmt wie viele Elemente ausgelassen werden sollen, bevor Elemente dem Stream weitergereicht werden. Im letzten Beispiel werden die ersten beiden Bilder aus dem Stream entfernt und die restlichen vorhanden in einer Liste gesammelt. Ein häufiger Anwendungsfall ist das Transformieren von Objekten einer Collection. Dafür wurde die Methode map bereitgestellt:

```
01:     // map (123)
02:     pictures = getPictures();
03:     List<String> pictureTitles = pictures.stream()
04:         .map(Picture::getTitle)
05:         .collect(toList());
```

Das Beispiel transformiert eine Liste von Bildern in eine Liste von Titeln der Bilder. Die Mapmethode kann entweder mit einer Methodenreferenz oder mit einem Lambda Ausdruck parametrisiert werden. Oft ist es nötig den Inhalt eines Streams zu verändern. Ein Beispiel für so einen Anwendungsfall wäre die Erzeugung eines Streams von Buchstaben aus einem Stream von Wörtern. Die Methode “flatMap” erlaubt so eine Transformation:

```
01:     // flatMap (126)
02:     String[] words = {"Online", "Picture", "Application"};
03:     List<String> characters = Arrays.stream(words)
04:         .map(w -> w.split(""))
05:         .flatMap(Arrays::stream)
06:         .collect(toList());
```

Aus den Wörtern des Arrays in Zeile 2 wird durch Anwendung der flatMap-Methode in Zeile 5 ein Stream von Zeichen. Es passiert quasi eine Entfaltung bzw. Entpacken. Wird ein bestimmtes Element oder Elemente mit einem bestimmten Merkmal gesucht so können die Methoden anyMatch, allMatch und noneMatch verwendet werden:

```
01: // anyMatch, allMatch, noneMatch (129/130)
02: pictures = getPictures();
03: boolean anyMatch =
    pictures.stream().anyMatch(Picture::isRgbMode);
04: boolean allMatch =
    pictures.stream().allMatch(Picture::isRgbMode);
05: boolean noneMatch =
    pictures.stream().noneMatch(Picture::isRgbMode);
```

Die anyMatch-Methode durchsucht den Stream solange bis ein Objekt gefunden wird das dem durch den durch Lambad-Ausdruck oder Methoden Referenz definiertem Umstand entspricht. Die allMatch-Methode überprüft ob alle Elemente des Streams dem Umstand entsprechen und die noneMatch Methode überprüft ob alle Elemente dem Umstand nicht entsprechen. Alle drei Methoden sind terminal, daher liefern sie ein skalares Ergebnis, bestehend aus einem einzigen Wert. Ähnlich verhalten sich die Methoden findAny und findFirst:

```
01: // findAny (130) findFirst (131)
02: pictures = getPictures();
03: pictures.stream()
04:     .filter(Picture::isRgbMode)
05:     .findAny()
06:     .ifPresent(d -> System.out.println(d.getTitle()));

07: pictures.stream()
08:     .filter(Picture::isRgbMode)
09:     .findFirst()
10:     .ifPresent(d -> System.out.println(d.getTitle()));
```

Im Vergleich zu den zuvor erwähnten Methoden sind findAny und findFirst nicht terminal. Sie suchen jeweils ein Element das dem Lambda Ausdruck entspricht oder von der referenzierten Methode akzeptiert wird. Als Ergebnis liefern diese Methoden ein Optional Objekt was als Container für das gefundene oder nicht gefundene Objekt dient. Durch die ifPresent-Methode des Objekts kann definiert werden was passieren soll, sollte ein Element gefunden werden. Obwohl sehr ähnlich, unterscheiden sich die Methoden findAny und findFirst. Die Methode findFirst ist viel restriktiver, da hier

wirklich nach dem ersten Vorkommen im Stream gesucht wird. Durch die Methode `reduce` ist es möglich die Elemente eines Streams auf ein einziges Objekt zu reduzieren:

```
01: // reducing (132)
02: long totalSize = pictures.stream()
03:     .map(Picture::getSizeInBytes)
04:     .reduce(0L, Long::sum);
```

Im gezeigten Beispiel wird durch Verwendung von „`reduce`“ der benötigte Speicherplatz aller Bilder berechnet. Zuvor wird noch der Stream von Bildern in einen Stream von Zahlen verwandelt, Zeile 3. Das erste Argument der `reduce`-Methode ist der Akkumulator. Dieser wird bei jedem wiederholten Aufruf mit dem Wert der letzten Berechnung neu befüllt. Eine alternative Methode um Elemente eines Streams zu summieren zeigt folgendes Beispiel:

```
01: // sum (145)
02: totalSize = pictures.stream()
03:     .mapToLong(Picture::getSizeInBytes)
04:     .sum();
```

Auch hier findet eine Transformation des Streams statt. Diesesmal jedoch durch die Methode `mapToLong` die einen Stream von `long` Zahlen erzeugt durch die Anwendung des Lambda Ausdrucks, Zeile 3. Wird das Bild gesucht das am meisten Speicherplatz verbraucht, so kann dies durch Hilfe der `max` Aggregatmethode erfolgen:

```
01: // max (147)
02: long maxSize = pictures.stream()
03:     .mapToLong(Picture::getSizeInBytes)
04:     .max()
05:     .getAsLong();
```

Die Methode liefert ein `Optional`-Objekt auf dessen Inhalt in Zeile 5 mit der `getAsLong`-Methode zugegriffen wird. In Java 8 existieren unterschiedliche Versionen von `Optional` Typen. Neben `Optional` auch `OptionalInt` und `OptionalDouble` sowie `OptionalLong`. Letztere Variante wird in diesem Beispiel retourniert. Sehr häufig werden `Collections` von Zahlenbereichen für Berechnungen gebraucht. Bisher wurden solche durch eine Schleife erzeugt. In Java 8 ist es nun möglich

Collections mittels einer Generatorfunktion zu erzeugen. Das folgende Beispiel zeigt die Erzeugung von Zahlenbereichen:

```
01: // range (147), rangeClosed (150)
02: List<Long> range = LongStream.range(1, 100)
03:     .boxed()
04:     .collect(toList());

05: List<Long> rangeClosed = LongStream.rangeClosed(1, 100)
06:     .boxed()
07:     .collect(toList());
```

In den Zeilen 2 - 4 wird die Erzeugung eines offenen Zahlenbereichs gezeigt. Offen bedeutet hier, dass die Zahlen 1 und 100 nicht im Ergebnis Stream vorkommen. Zeilen 5 - 7 demonstrieren die Erzeugung eines geschlossenen Zahlenbereichs. Geschlossen bedeutet hier, dass die Zahlen 1 und 100 auch im Stream vorkommen. In Java 8 stehen zwei weitere Generatoren zur Erzeugung von Streams beliebigen Typs zur Verfügung, `iterate` und `generate`:

```
01: // iterate (153)
02: Stream.iterate(0, a -> a * 2)
03:     .limit(5)
04:     .forEach(System.out::println);

05: // generate (155)
06: Stream.generate(() -> Math.random() * 100)
07:     .limit(10)
08:     .forEach(System.out::println);
```

Der Unterschied zwischen „`iterate`“ und „`generate`“ ist dass `iterate` einen initialen Wert für den Akkumulator erwartet und sukzessive den Lambda Ausdruck auf diesen Wert anwendet. Die Ausführung erfolgt also in einer Kette ausgehend von einem initialen Wert. Die `generate`-Methode im Gegensatz dazu erwartet ein Objekt des Typs `Supplier` und ruft bei jedem Aufruf die `factory`-Methode von diesem auf. Werden parallele Streams verwendet so muss darauf geachtet werden, dass im `Supplier` kein Zustand gespeichert wird.

Die Kombination aus Streams und Lambda Ausdrücke sind eine mächtige Erweiterung der Java Sprache. Programmteile können damit sehr präzise formuliert werden und erleichtern somit die Wartung und das Testen.



#### 4.2.4 Web Applikationen auf Basis von Spring und Hibernate

Für die Entwicklung von umfangreichen Web und Enterprise Applikationen auf Basis von Java, werden in der Geschäftswelt vornehmlich entweder die Java Enterprise Edition, kurz Java EE, oder der Spring Entwickler Stack eingesetzt.

Java EE ist eine Entwicklungsplattform für die Umsetzung komplexer Applikationen. Solche Applikationen bestehen zumeist aus unterschiedlichen Schichten wie der Frontend Schicht, der Mittelschicht und der Backend Schicht. Die Frontendschicht besteht meist aus einem Web Framework oder einem GUI zur Bedienung der Applikation. In der Mittelschicht ist meist die Logik für Sicherheitskritische Aspekte und Transaktionen zu finden. Das Backend stellt Zugriff auf Datenbanken sowie Systemen von dritten zur Verfügung. Die Java EE Plattform definiert eine Schnittstelle für Komponenten aus jeder Schicht der Anwendung und stellt zusätzlich Dienste wie naming, injection oder Ressourcen-Management zur Verfügung. Die Komponenten von Web oder Enterprise Anwendungen werden in einem Java EE kompatiblen Container installiert. Kompatibel in diesem Sinne bedeutet, dass er die Spezifikation der Plattform vollständig unterstützt. Die Komponenten der Applikation kommunizieren niemals direkt miteinander, sondern ausschließlich über die vom Container zur Verfügung gestellten Schnittstellen. Dies hat den Vorteil, dass benötigte Dienste den Komponenten transparent vom Container bei Instanziierung übergeben werden können. Des Weiteren bietet das Container basierte Modell eine Abstraktion des Zugriff auf Ressourcen. Dadurch werden Entwickler von sich wiederholenden Aufgaben entlastet. Jede Komponente der Plattform wird in einer Spezifikation definiert, welche die Methoden, Dokumentation und erwartete Laufzeitumgebung beschreibt. Die aktuelle Version Java EE 7 basiert auf den Wegbereitern Java EE 5 und 6 die, beeinflusst durch die Spring Technologie, die Entwicklung deutlich vereinfachten. [Gupta13]

Das Spring Framework, oder besser die Spring Technologie, ging ursprünglich aus dem Buch „Expert One-on-One J2EE Development without EJB“ (Wrox 2002), von Rod Johnson und Jürgen Höller, hervor. Aktuell in Version 4 ist es nicht mehr nur ein Framework sondern ein kompletter Stapel von Technologien. Jede Schnittstelle kann optional in den Entwicklungsprozess eingebunden werden. Spring stellt dafür eine Reihe von Komponenten zur Verfügung. [Schaefer14]

In fast jeder Applikation gibt es Funktionalität die nicht eindeutig einer Schicht zugeordnet werden kann. Ein Beispiel für so eine Schichtenübergreifende Logik wäre Logging. Um solche Anforderungen besser und sauber implementieren zu können, erlaubt es Spring das Konzept der „Aspekt orientieren Programmierung“ (AOP) einzusetzen. Der Standard Mechanismus in Spring ist dabei die Erzeugung von dynamischen Proxies aber auch der Einsatz von AspectJ ist möglich. [Schaefer14]

Der Java Web Stack beinhaltet die Technologien Java Server Pages (JSP) und Java Server Faces (JSF). Beide stellen eine jeweils eigene Expression Language zur Verfügung mit einer jeweils unterschiedlichen Syntax. Um das Problem zu lösen wurde die Unified Expression Language (EL) eingeführt mit dem JSP Standard 2.1. Da sich Spring jedoch viel schneller entwickelt als der Java EE Stack wurde mit Spring 3 die Spring Expression Language (SpEL) veröffentlicht. Mit SpEL lassen sich Spring beans und Java Objekte zur Laufzeit verändern. Mehr zu Spring Beans später. [Schaefer14]

Bevor Benutzerdaten weiter verarbeitet werden können, müssen diese validiert werden. Spring bietet dazu eine Schnittstelle die erlaubt Logik zur Validierung zu kapseln. Des Weiteren existiert eine Klasse ValidationUtils von der Implementierungen der Schnittstelle bezogen werden können, die die gängige Validierungslogik beinhalten. [Schaefer14]

Spring bietet Unterstützung für viele Persistenz Mechanismen wie der Java Database Connectivity (JDBC), Java Data Objects (JDO), Java Persistence Architecture (JPA) oder Hibernate. Durch den Big Data Trend motiviert werden auch einige NoSQL, Graph und Dokumentenbasierte Datenbanken unterstützt. [Schaefer14]

Viele Applikationen müssen mit fremden Systemen kommunizieren. Ein kritischer Punkt ist dabei das Datenaustauschformat das bei der Kommunikation eingesetzt wird. Eines der gängigsten Formate ist dabei XML. Um Objekte in XML Dokumente zu serialisieren bietet Spring Unterstützung für einige Bibliotheken die diese Aufgabe erfüllen. Darunter sind die Java Architecture for XML Binding (JAXB), Castor, XStream, JiBX und XMLBeans. [Schaefer14]

Gleichzeitiger Zugriff auf Daten erfordert den Einsatz eines Transaktionsmechanismus um die Korrektheit dieser sicher zu stellen. Spring bietet Unterstützung für die Verwendung programmatischer und deklarativer Transaktionen. Durch die Abstraktion

die die Plattform bietet ist es für Entwickler leicht den verwendeten Transaktionsmechanismus zu tauschen. So kann zu Beginn mit lokalen Transaktionen begonnen werden und später auf globale oder auf mehrere Ressourcen übergreifende Transaktionen gewechselt werden. [Schaefer14]

Wie bereits erwähnt hatte der Spring Ansatz zur Gestaltung der Architektur von Applikationen einen erheblichen Einfluss auf die Java Enterprise Plattform. Spring unterstützt einige Java EE Technologien und integriert diese in den Technologie Stack wodurch das Zusammenarbeiten ermöglicht wird. So ist es zum Beispiel ohne Probleme möglich via JNDI Referenzen auf Java EE Container Ressourcen zu beziehen und diese im Spring Context zu verwenden. [Schaefer14]

Eine neue Funktionalität in Spring 4 ist die Unterstützung für die Java API für Web Sockets. Diese API ermöglicht den Aufbau eines bidirektionalen Kommunikationskanal zwischen einem Client und einem Server. Dadurch wird ein rascher Nachrichtenaustausch verfügbar, was die Latenz in solchen Anwendungsszenarien verringert und die Antwortzeit dadurch erhöht. [Schaefer14]

Die Spring Remoting Unterstützung erlaubt die einfache Kommunikation mit entfernten Diensten. Zur Verfügung stehen unter anderem Java Remote Method Invocation (RMI), JAX-WS, Caucho Hessian und Burlap, Java Messaging Service, Advanced Message Queuing Protocol (AMQP) und REST. [Schaefer14]

Der Versand von E-Mails ist eine typische Anforderung in Applikationen. Durch die Unterstützung und Abstraktion der Java Mail API ist auch das kein Problem mit Spring. [Schaefer14]

Durch die Spring Job Scheduling Abstraktion ist es ein Leichtes häufig auftretende Aufgaben im Hintergrund der Applikation auszuführen. Spring unterstützt das Quartz Framework sowie die JDK Timer API. [Schaefer14]

Mit Java 6 wurde die Unterstützung für dynamische Skriptsprachen eingeführt. Damit wurde das Ausführen von Programmen die nicht in Java programmiert sind auf der virtuellen Java Maschine möglich. So zum Beispiel JRuby, Groovy oder Java Script. Spring nutzt diese Funktionalität sehr geschickt und erlaubt es sogar, Komponenten die nicht in Java entwickelt wurden, innerhalb einer Java Applikation zu verwenden. [Schaefer14]

Der Kern der Spring Technologie besteht aus einem Inversion of Control (IoC) Container. Das IoC Prinzip wird unterschieden in Dependency Lookup und Dependency Injection. Dependency Lookup kann auf eine von zwei Arten erfolgen: Dependency Pull und Contextualized Dependency Lookup (CDL). Beim Dependency Pull werden die Abhängigkeiten, welche für die Funktion einer Komponente benötigt werden, von der Komponente selbst, von einer zentralen Registrierungsstelle, bezogen. Bei der anderen Version, dem Contextualized Dependency Lookup, werden benötigte Abhängigkeiten von einem Container, der die Ressourcen verwaltet, geholt. Das Dependency Injection Prinzip kann unterteilt werden in Constructor und Setter Dependency Injection. Durch Constructor basierte DI werden die Abhängigkeiten der Komponente vom IoC Container direkt bei der Konstruktion durch den Konstruktor übergeben. Bei der Variante Setter Dependency Injection werden die Abhängigkeiten durch Aufrufe von Setter Methoden vom Container der Komponente übergeben. Welche Version des DI letztendlich verwendet werden soll hängt vom spezifischen Anwendungsfall ab. Constructor basierte DI bietet sich an wenn Abhängigkeiten unbedingt bei der Instanziierung benötigt werden. Das erlaubt auch die Erstellung von unveränderlichen (immutable) Komponenten. Setter basierte DI bietet sich sehr gut an wenn die Komponente über hausgemachte Abhängigkeiten verfügt, diese aber vom IoC Container überschreiben lassen möchte. [Schaefer14]

Der Inversion of Control Container von Spring ist hauptsächlich für Dependency Injection gedacht. Obwohl die Anwendung von Dependency Lookup auch möglich ist. Komponenten die in einer Spring basierten Applikation entwickelt werden, sollten immer das DI Prinzip bevorzugen anstelle Abhängigkeiten selbst zu instanzieren. Um einen Spring IoC Container zu erzeugen muss eine Instanz der BeanFactory Schnittstelle erzeugt werden. In Standalone Applikation hat dies gesondert in einer main Methode zu erfolgen. In Web Applikationen stehen dafür Mechanismen zur Verfügung die den Container automatisch beim Start der Anwendung instanzieren. Die Bean-Factory (Bohnen Fabrik) ist für das Management der Komponenten im Kontext, sowie für den Lebenszyklus dieser verantwortlich. Der Begriff bean und Komponenten sind im Kontext IoC gleich bedeutend. Wird für die Applikation nur ein Inversion of Control Container benötigt so genügt die Verwendung der Bean-Factory. Nachdem eine Fabrik erzeugt wurde muss diese mit Abhängigkeitsinformationen befüllt werden. Danach können die konfigurierten Komponenten von der Bean Factory bezogen werden (nach dem Dependency Pull

Prinzip). Die Konfiguration der Fabrik kann auf eine von zwei Arten erfolgen. Entweder programmatisch durch Methodenaufrufe oder durch Angabe einer Konfigurationsdatei. Intern wird die Konfiguration einer Komponente als Instanz der Schnittstelle BeanDefinition gespeichert. Dabei werden nicht nur Informationen über die Komponente selbst, sondern auch die konfigurierten Abhängigkeiten vermerkt. Bei Konfiguration durch eine Datei stehen zwei Möglichkeiten zur Verfügung. Die Konfiguration auf Basis einer Java Properties Datei und auf Basis einer XML Datei. Jede konfigurierte Komponente kann eine ID, ein Name oder beides zugeordnet werden. Wird keine ID festgelegt so wird eine anonyme Instanz erzeugt. Die ID wird für die Abhängigkeits Definition oder für den Bezug einer Referenz von der Fabrik, benötigt. Bei der Angabe von mehreren Namen durch ein Komma getrennt, dient der erste Name als eigentlicher Komponenten Name und die anderen Namen jeweils als Alias. [Schaefer14]

Eine Erweiterung der Bean-Factory ist der Application Context. Er fügt dem IoC Container zusätzliche Funktionen wie Lebenszyklusverwaltung, Internationalisierung, Messaging und weitere, hinzu. Gestartet werden kann der Kontext sowie die Factory entweder programmatisch oder in einem Web Container durch einen ContextLoaderListener.

Die Konfiguration der Komponenten kann, bei Verwendung eines Application Context, seit Spring 2.5 auch mittels Java Annotations erfolgen. Diese Möglichkeit löste viele Diskussionen in der Spring Community aus. Welche Variante der Konfiguration nun die besser ist hängt vom Anwendungsfall ab. Üblich ist es, die Infrastruktur der Applikation, wie zum Beispiel Transaktionen, Logging, JMS oder JMX, mittels XML Konfigurationsdatei zu definieren und Komponenten die diese Abhängigkeiten benötigen mittels Annotations. [Schaefer14]

Web Applikationen werden häufig durch die Anwendung des MVC Entwurfsmusters umgesetzt. Auch das Spring Framework unterstützt die Entwicklung nach diesem Prinzip mit Hilfe des Spring Web MVC Moduls. MVC, kurz für Model View Controller, ist ein Entwurfsmuster für die Präsentationsschicht. Es definiert eine Architektur und separiert dabei klar die Verantwortlichkeiten. Wie der Name schon verraten lässt besteht es aus drei Teilen. Das Model kapselt die Geschäftsdaten der Anwendung. Zum Beispiel in einer E – Commerce Anwendung wäre das der Warenkorb, Profil Informationen des Benutzers oder Daten einer Bestellung. Die View Komponenten sind dafür verantwortlich die

Daten des Models zu visualisieren und dem Benutzer Interaktionsmöglichkeiten zur Verfügung zu stellen. Weitere Funktionen wie Internationalisierung, Validierung oder themenbasierte Visualisierung können in den View Komponenten implementiert sein. Der Controller ist für die Bearbeitung von Benutzerinteraktionen verantwortlich. Er erzeugt einen Aufruf von Service Komponenten und befüllt das Model mit den erhaltenen Daten. Danach initiiert er den Aufruf der View Komponenten. Durch das Aufkommen der Ajax basierten Web Architektur wurde das MVC Muster etwas abgeändert. Vermehrt wird nicht mehr das Model durch eine View visualisiert, sondern die Daten des Models werden in der Rohfassung vom Controller an den Client geschickt der durch zusätzliche View Logik eine partielle Aktualisierung der dargestellten Webseite vornimmt. Die folgende Abbildung zeigt die Architektur des Spring Web MVC Moduls. [Schaefer14]

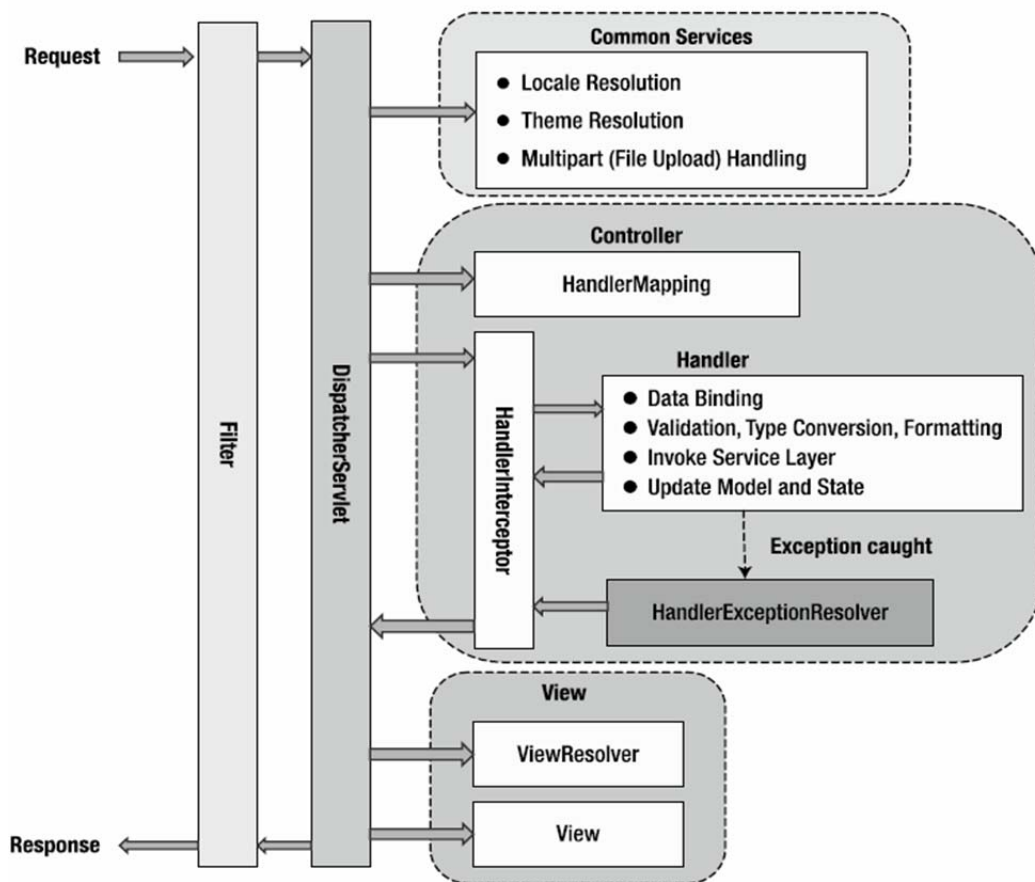


Abbildung 9: Spring MVC Architektur. Quelle [Schaefer14]

Alle Anfragen die an den Server gestellt werden, müssen zunächst die definierte Filterkette durchlaufen. Es können beliebig viele Filter definiert werden. Auch Einschränkungen bezüglich der Anwendung der Filter können anhand von URL Muster definiert werden. Die Verwendung von Filter ist optional.

Wurde die Anfrage gefiltert, oder auch nicht, so wird sie im nächsten Schritt vom Dispatcher Servlet entgegen genommen. Dies ist eine zentrale Komponente im Spring MVC Modul. Anfragen werden vom Dispatcher an entsprechende Controller weiter geleitet.

In einer Spring Web Applikation können beliebig viele Dispatcher Servlets konfiguriert werden. Jedes definierte Servlet erhält einen eigenen Web Applikations Kontext. Dieser ist ein Kindkontext des Root Kontext den sich alle Servlets teilen.

In der Abbildung unter Common Services gezeigte Komponenten sind jene, die im Web Applikations Kontext definert werden. Sie stehen bei der Bearbeitung einer jeden Anfrage zur Verfügung.

Das Handler Mapping definiert welche Anfragen von welchem Controller beantwortet werden sollen. Es wird in der Konfigurationsdatei des Web Applikations Kontext festgelegt. Seit Spring 2.5 ist es auch möglich das Mapping weg zu lassen und das Ziel der Anfragen in den Controllern selbst durch Annotations anzugeben.

Ähnlich dem Prinzip der Filter funktionieren Handler Interceptors. Sie fangen Anfragen an bestimmte Controller ab und können Aufgaben wie Validierung, Logging oder Sicherheitskritische Prüfungen durchführen.

Treten bei der Bearbeitung von Anfragen unerwartet Fehler auf, so kann es sein, dass eine Exception vom Controller geworfen wird. Zur Behandlung solcher Fehler stehen Handler Exception Resolver zur Verfügung. Mit diesen kann Verhalten im Fehlerfall definiert werden. Zum Beispiel kann die Anfrage auf eine Fehlerseite umgeleitet werden die erklärt was soeben passiert ist.

Wie bereits erwähnt dient die View Komponenten im MVC Muster zur Darstellung der Modelldaten. Im Laufe der Jahre haben sich viele verschiedene Frameworks entwickelt die dies auf unterschiedliche Weise erleichtern möchten. Java selbst bietet dafür die Java Server Pages Technologie an. Durch die Angabe einer View Resolver Komponente kann die verwendete View Technologie sehr leicht austauschbar gemacht werden. Auch können verschiedene Anfragen mit unterschiedlichen View Technologien beantwortet werden. Welche Technologie verwendet werden soll, wird vom Controller bestimmt, der durch Rückgabe einer Kennung, dem Resolver mitteilt, welche View verwendet werden soll.

Wie bereits erwähnt werden die Daten des Models vom Controller über die Service Schicht oder Business Schicht der Applikation bezogen. Diese wiederum führt meist komplexe Aufgaben durch um eine Antwort zu erzeugen. Dazu gehört oft auch der Aufruf an eine Persistenz Schicht. Dieser Schicht obliegt die Aufgabe Daten zu speichern, zu laden und zu löschen. Implementieren lässt sich diese Schicht auf verschieden Arten. Meist kommt dabei ein Persistenz Framework zum Einsatz das dem Entwickler viele Aufgaben abnimmt.



Hibernate ist ein Persistenz Framework das das Speichern, Laden und Löschen von Java Objekten aus einer relationalen Datenbank erlaubt. Frameworks mit dieser Funktionalität werden häufig auch als Objektrelationaler Mapper, kurz ORM, bezeichnet. Sie lösen das komplexe Problem hierarchische Datenstrukturen auf relationale, 2-dimensionale Datenstrukturen, abzubilden. Um Klassen oder ganze Klassen-Hierarchien in eine Datenbank speichern zu können muss zunächst ein sogenanntes Mapping angelegt werden. Dies kann mittels einer Mapping Datei oder durch Angabe von Java Annotations in den Klassen selbst erfolgen. Für jede Klasse wird dabei in der Mapping Datei definiert in welche Tabelle der Datenbank diese gespeichert werden soll. Auch für die Attribute der Klasse wird festgelegt in welche Spalte das Attribut gespeichert werden soll. Zur Erzeugung von Primärschlüsseln und Indizes gibt es besondere Angaben die Hibernate anweisen wie ein Schlüssel erzeugt oder ein Index angelegt werden soll. Im Fall der Abbildung von Klassen Hierarchien gibt es vier Möglichkeiten dies zu tun. Bei der Strategie Tabelle pro konkrete Klasse, wird für jede konkrete Klasse eine eigene Tabelle verwendet. Dies kann entweder mit implizitem Polymorphismus oder durch expliziten Polymorphismus erfolgen. Bei der Tabelle pro Klassen-Hierarchie wird jeweils die komplette Hierarchie in eine eigene Tabelle gespeichert. Auch benachbarte Klassen die sich eine Super-Klasse teilen werden in die selbe Tabelle gelegt. Dies hat den Nachteil dass Attribute nicht mit einer Nicht-Null Einschränkung definiert werden können. Auf der anderen Seite bringt es Performanzvorteile. Beim Abrufen von Objekten müssen keine aufwändigen Join Operationen durchgeführt werden. Eine andere Form der Speicherung von Hierarchien ist die Tabelle pro Subklasse Strategie. Hier wird jede Klasse der Hierarchie gesondert in eine eigene Tabelle gespeichert. Im Vergleich zur Tabelle pro konkrete Klasse Strategie, werden hier in einer Tabelle nur jeweils jene Attribute gespeichert, die zu einer Klasse gehören und nicht alle Attribute der gesamten Hierarchie. [Bauer07]

Hibernate bietet auch eine eigene Abfragesprache um Objekte aus der Datenbank zu laden oder Werte abzuleiten. Die Hibernate Query Language (HQL) erlaubt es datenbankunabhängige Abfragen zu formulieren und ermöglicht auf diese Weise eine Entkoppelung von spezifischen Datenbanksystemen. Eine andere Art der Abfrageformulierung erfolgt durch die Verwendung von Kriterienabfragen. Hierbei wird ein Musterobjekt mit Werten befüllt und Hibernate übergeben. Alle Objekte die dem Muster entsprechen werden von der Session geladen und mit dem Persistenz Kontext assoziiert. Für jeden Thread der Ausführung einer Anwendung welcher auf die

Persistenzschicht zugreifen möchte, muss eine eigene Session erzeugt werden. Dazu muss aber zunächst einmal eine Factory instanziiert werden. Die Session Factory ist ein quasi Singleton das beim Start der Applikation erzeugt wird. Konfiguriert wird sie mit einer Liste von Mappings und einer Datenbankverbindung sowie mit einem Transaktionsmanager. Nach dem Start der Factory kann diese um Sessions gebeten werden. Spring bietet dabei eine eigene Abstraktion, die jedem Thread genau eine Session zuteilt, um zu verhindern dass pro Thread mehrere Sessions existieren. Durch eine Session ist Interaktion mit den persistieren Objekten möglich. Alle geladenen oder gespeicherten Objekte sind, solange eine Session aktiv ist, mit deren Persistenz-Kontext assoziiert. Dieser bemerkt automatisch Veränderungen von Objekten und speichert diese ab sobald eine Session oder Transaktion geschlossen wird. Ein explizites Auffordern der Session zum Speichern der Objekte ist nicht nötig. Werden Objekte die persistiert sind in einer Session zwischengespeichert, so werden sie automatisch bei der Schließung der Session vom Persistenz-Kontext gelöst und gehen damit in den Status Detached über. Solche Objekte müssen, sofern dies gewollt ist, nachdem ihr Zustand verändert wurde, wieder explizit mit einem Kontext verbunden werden. Spring bietet eine Vielzahl von Unterstützungen die das Arbeiten mit Hibernate als Persistenz Mechanismus der Wahl erleichtert. [Bauer07]

Im Folgenden Beispiel wird gezeigt wie Spring, das Spring Web MVC und Hibernate verwendet werden können um moderne Web Applikationen umzusetzen.

Das Beispiel verwendet eine Klasse die Daten eines Bildes kapselt. Aus Gründen der Übersichtlichkeit wurden nur zwei Felder hinzugefügt.

```
01: package master.tomcat;
02:
03: public class Picture
04: {
05:     private long id;
06:     private String name;
07:     public Picture() {}
08:     public long getId() { return id; }
09:     public void setId(long id) { this.id = id; }
10:     public String getName() { return name; }
11:     public void setName(String name) { this.name = name; }
12: }
```

Das Attribut id dient als Primärschlüssel des Domain Objekts und das name Attribut identifiziert das Bild mit einem Namen. Das Mapping File für dieses Domain Objekt sieht wie folgt aus:

```
01: <?xml version='1.0' encoding='utf-8' ?>
02: <!DOCTYPE hibernate-mapping PUBLIC "-//Hibernate/Hibernate
    Mapping DTD 3.0//EN"
    "http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">
03: <hibernate-mapping package="master.tomcat" default-
    access="field">
04: <class name="Picture" table="pictures">
05: <id name="id" column="id" type="java.lang.Long" length="255">
    <generator class="assigned"/>
06: </id>
07: <property name="name" column="name" type="java.lang.String" not-
    null="true" length="255"/>
08: </class>
09: </hibernate-mapping>
```

Zeile 3 definiert das Package für welches das Mapping Gültigkeit hat und legt die Standard Zugriffsmethode fest um die Werte des Domain Objekt lesen zu können. Zeile 4 definiert den Namen der Tabelle indem Daten des Objekts gespeichert werden. In den Zeilen 5 - 6 wird festgelegt wie der Primärschlüssel erzeugt werden soll. Für dieses Beispiel wird er programmatisch von der Anwendungslogik zugewiesen. Zeile 7 legt fest, in welcher Spalte der Tabelle der Name des Bildes gespeichert werden soll.

```

01: package master.tomcat;
02: import org.apache.catalina.Context;
03: import org.apache.catalina.LifecycleException;
04: import org.apache.catalina.startup.ContextConfig;
05: import org.apache.catalina.startup.Tomcat;
06: import java.io.File;
07: public class TomcatMain
08: {
09: public static void main(String[] args) throws LifecycleException
10: {
11:     final Tomcat tomcat = new Tomcat();
12:     tomcat.setBaseDir("./tomcat");
13:     tomcat.setPort(8080);
14:     final File applicationPath = new File("./tomcat/webapp");
15:     Context rootContext = tomcat.addContext("/",
applicationPath.getAbsolutePath());
16:     rootContext.addLifecycleListener(new ContextConfig());
17:     Tomcat.initWebappDefaults(rootContext);
18:     tomcat.start();
19:     tomcat.getServer().await();
20: }
21: }

```

Als Servlet Container wird ein embedded Tomcat 8 gestartet. In den Zeilen 11 - 17 wird der Container konfiguriert. Zeile 12 definiert ein Verzeichnis in dem temporäre Dateien erzeugt werden können. Zeile 13 legt das Port des HTTP Connector fest. In Zeile 14 wird der Pfad zu den Web Ressourcen der Applikation gesetzt. Zeile 15 - 17 konfiguriert den Servlet Kontext und fügt Standard Servlets hinzu. In Zeile 18 wird der Server gestartet und in Zeile 19 wird solange gewartet bis der Server runter gefahren wird.

Die Definition des Spring Web Kontext sieht für dieses Beispiel wie folgt aus:

```

01: <beans xmlns="http://www.springframework.org/schema/beans"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:context="http://www.springframework.org/schema/context"
xmlns:tx="http://www.springframework.org/schema/tx"
xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-
beans.xsd
http://www.springframework.org/schema/context

```

```

    http://www.springframework.org/schema/context/spring-
    context.xsd
    http://www.springframework.org/schema/tx
    http://www.springframework.org/schema/tx/spring-
    tx.xsd">
02: <context:annotation-config/>
03:
04: <context:component-scan base-package="master.tomcat"/>
05:
06: <bean id="dataSource"
    class="org.apache.commons.dbcp2.BasicDataSource" destroy-
    method="close">
07: <property name="url"
    value="jdbc:postgresql://localhost:5432/master-sample"/>
08: <property name="username" value="christian"/>
09: <property name="password" value="secret"/>
10: </bean>
11:
12: <bean id="sessionFactory"
    class="org.springframework.orm.hibernate4.LocalSessionFactoryBean">
13: <property name="dataSource" ref="dataSource"/>
14: <property name="mappingLocations">
    <list>
    <value>classpath:/master/tomcat/Picture.hbm.xml</value>
    </list>
15: </property>
16: <property name="hibernateProperties">
    <value>
    hibernate.dialect=org.hibernate.dialect.PostgreSQL9Dialect
    hibernate.hbm2ddl.auto=create
    </value>
17: </property>
18: </bean>
19:
20: <bean id="transactionManager"
    class="org.springframework.orm.hibernate4.HibernateTransactionManag
    er">
21: <property name="sessionFactory" ref="sessionFactory"/>
22: </bean>
23: <tx:annotation-driven transaction-manager="transactionManager"/>
24: </beans>

```

In Zeile 1 wird festgelegt welches Schema zur Validierung der Kontextkonfiguration verwendet werden soll. Zusätzlich wird für jeden verwendeten Namespace ein Schema

zugewiesen. In diesem Beispiel wurden Funktionen aus dem Context und Transaction Namensraum verwendet. In Zeile 2 und 4 wird dem IoC Container mitgeteilt das Teile der Applikation mittels Annotations konfiguriert wurden und dass dafür das angegebene Package nach annotierten Klassen durchsucht werden soll. Zeile 6 - 10 definiert ein Data Source Bean das zum Bezug von Datenbankverbindungen herangezogen werden kann. Zeile 12 - 18 definiert die Konfiguration der Hibernate Session Factory. Es wird in Zeile 13 die zu verwendende Quelle für Datenbankverbindungen konfiguriert, in Zeile 14 wird die Liste von Mapping Dateien festgelegt. In Zeile 16 - 17 wird festgelegt für welches Datenbanksystem die Factory optimiert und ob ein Schema beim Starten angelegt werden soll. In den Zeilen 20 - 22 wird der Transaktionsmanager konfiguriert und der Session Factory zugewiesen. Zeile 23 weist den Spring Container an das für dieses Beispiel deklarative Transaktionen verwendet werden sollen. Um den Web Applikations Kontext beim Starten des Containers zu starten wird ein Web Applikations Initializer benötigt:

```
01: package master.tomcat;
02: import org.springframework.web.WebApplicationInitializer;
03: import
    org.springframework.web.context.support.XmlWebApplicationContext;
04: import org.springframework.web.servlet.DispatcherServlet;
05: import javax.servlet.ServletContext;
06: import javax.servlet.ServletException;
07: import javax.servlet.ServletRegistration;

08: public class Initializer implements WebApplicationInitializer
09: {
10:     @Override
11:     public void onStartup(ServletContext servletContext) throws
        ServletException
12:     {
13:         XmlWebApplicationContext appContext = new
            XmlWebApplicationContext();
14:         appContext.setConfigLocation("file:./tomcat/webapp/WEB-
            INF/spring/dispatcher-config.xml");

15:         ServletRegistration.Dynamic dispatcher =
            servletContext.addServlet("dispatcher", new
                DispatcherServlet(appContext));
16:         dispatcher.setLoadOnStartup(1);
17:         dispatcher.addMapping("/");
18:     }
19: }
```

Die onStartUp Methode des Initilaizers wird ausgeführt, sobald der Container fertig konfiguriert wurde und bereit für die Instanziierung von Web Applikationen ist. In den Zeilen 13 und 14 wird daraufhin ein Web Context erzeugt und in den Zeilen 15 bis 17 ein Dispatcher Servlet im Servlet Context registriert welches auf den Root Pfad der Applikation hört und sofort nach dem Hinzufügen gestartet wird.

Das einzige Service der Persistenz Schicht ist ein data access Objekt (DAO):

```
01: package master.tomcat;
02: import org.hibernate.SessionFactory;
03: import org.springframework.stereotype.Service;
04: import javax.annotation.Resource;
05: @Service
06: public class PictureDao
07: {
08:     private SessionFactory sessionFactory;
09:     @Resource
10:     public void setSessionFactory(SessionFactory sessionFactory) {
11:         this.sessionFactory = sessionFactory; }
12:     public Picture load(final long id)
13:     {
14:         return (Picture) sessionFactory.getCurrentSession()
15:             .createQuery("from Picture p where p.id = :id")
16:             .setLong("id", id)
17:             .uniqueResult();
18:     }
19:     public void save(final Picture picture) {
20:         sessionFactory.getCurrentSession().save(picture); }
21: }
```

Das Service wird in Zeile 5 als Spring-Bean durch Angabe der Annotation Service markiert. Damit steht es für die Verwendung im Applikations Kontext zur Verfügung. In den Zeilen 9 - 10 wird dem Objekt eine Refernz zur Hibernate Session Factory via Dependency Injection vom IoC Container durch Angabe der Annotation Resource zugewiesen. In den Zeilen 11 - 14 ist die Methode load implementiert die zunächst eine Hibernate Session von der Factory bezieht, danach ein HQL Objekt erzeugt und diesem als Parameter den Primärschlüssel des gewünschten Objekts, das geladen werden soll, mitteilt. Durch den Aufruf von uniqueResult wird sichergestellt das nur ein Objekt

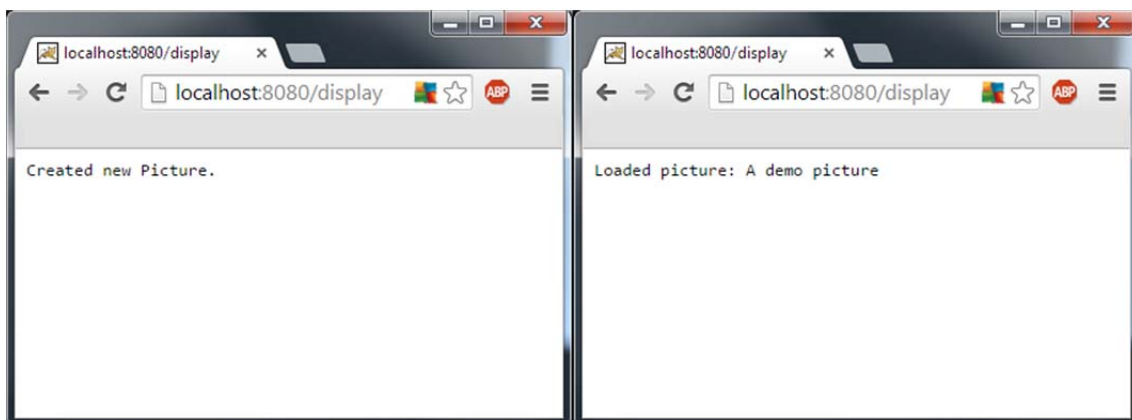
retourniert wird. Obwohl für diesen konkreten Anwendungsfall nur eines erwartet werden kann, so muss dies doch explizit dem Query Objekt mitgeteilt werden. In Zeile 15 wird die Methode zum Speichern von Domain Objekten implementiert.

Was nun noch fehlt ist ein Controller der es erlaubt Bilder über einen Web Browser in der Datenbank anzulegen und wieder zu laden:

```
01: package master.tomcat;
02:
03: import org.springframework.stereotype.Controller;
04: import org.springframework.web.bind.annotation.RequestMapping;
05: import org.springframework.web.bind.annotation.ResponseBody;
06: import javax.annotation.Resource;
07: import javax.servlet.http.HttpServletRequest;
08: import javax.transaction.Transactional;
09:
10: @Controller
11: public class FrontController
12: {
13:     private PictureDao pictureDao;
14:
15:     @Resource
16:     public void setPictureDao(final PictureDao pictureDao) {
17:         this.pictureDao = pictureDao;
18:     }
19:
20:     @RequestMapping("/display")
21:     @ResponseBody
22:     @Transactional
23:     public void render(final HttpServletRequest request) throws
24:         Exception
25:     {
26:         Picture p = pictureDao.load(1);
27:         if (p == null)
28:         {
29:             p = new Picture();
30:             p.setName("A demo picture");
31:             p.setId(1);
32:             pictureDao.save(p);
33:             response.getWriter().print("Created new Picture.");
34:         }
35:         else
36:         {
37:             response.getWriter().print("Loaded picture: " +
38:                 p.getName());
39:         }
40:     }
41: }
```



In Zeile 15 und 16 wird dem Controller mittels Dependency Injection (Resource Annotation) eine Referenz auf das DAO der Persistenz Schicht gegeben. Normalerweise bedient sich der Front Controller der Business Schicht, aber für die Einfachheit des Beispiels wurde diese ausgelassen. Die Zeilen 18 - 30 zeigen die Implementation der einzigen Web Methode des Controllers. In Zeile 18 wird der Methode ein Pfad zugewiesen und in Zeile 19 dem Container gesagt, dass die Antwort auf Anfragen die an diese Methode gerichtet sind in der Methode selbst vollständig beantwortet werden. Durch die Transactional Annotation in Zeile 20 wird der Aufruf der Methode mittels AOP in eine Transaktion verpackt. In den Zeilen 23 - 29 wird die eigentliche Logik der Methode implementiert. Es wird zunächst versucht ein Bild mit der ID 1 aus der Datenbank zu laden. Schlägt dies fehl, so wird ein Bild mit dieser ID erzeugt und eine Nachricht an den Client gesendet. Ist bereits ein Bild in der Datenbank mit dieser ID vorhanden, so wird dessen Name an den Client gesendet. Ab dem zweiten Aufruf gibt diese Methode daher immer die gleiche Zeichenkette an den Client zurück. Das Ergebnis sieht wie folgt aus.



**Abbildung 10: Der Aufruf des Demo Beispiels.**

Das Bild zeigt links den ersten Aufruf und rechts einen Wiederholten Aufruf der Methode im Browser.

Um für das Beispiel alle notwendigen Bibliotheken zur Verfügung zu haben, wird am besten ein Maven Projekt (pom.xml) mit folgenden Abhängigkeiten angelegt.

```
01: <dependencies>
02:
03: <dependency>
04:   <groupId>org.apache.tomcat.embed</groupId>
05:   <artifactId>tomcat-embed-core</artifactId>
06:   <version>8.0.15</version>
07: </dependency>
08: <dependency>
09:   <groupId>org.apache.tomcat.embed</groupId>
10:   <artifactId>tomcat-embed-el</artifactId>
11:   <version>8.0.15</version>
12: </dependency>
13: <dependency>
14:   <groupId>org.apache.tomcat.embed</groupId>
15:   <artifactId>tomcat-embed-logging-juli</artifactId>
16:   <version>8.0.15</version>
17: </dependency>
18: <dependency>
19:   <groupId>org.apache.tomcat.embed</groupId>
20:   <artifactId>tomcat-embed-jasper</artifactId>
21:   <version>8.0.15</version>
22: </dependency>
23:
24: <dependency>
25:   <groupId>org.springframework</groupId>
26:   <artifactId>spring-context</artifactId>
27:   <version>4.1.1.RELEASE</version>
28: </dependency>
29: <dependency>
30:   <groupId>org.springframework</groupId>
31:   <artifactId>spring-orm</artifactId>
32:   <version>4.1.1.RELEASE</version>
33: </dependency>
34: <dependency>
35:   <groupId>org.springframework</groupId>
36:   <artifactId>spring-webmvc</artifactId>
37:   <version>4.1.1.RELEASE</version>
38: </dependency>
39:
40: <dependency>
41:   <groupId>org.hibernate</groupId>
42:   <artifactId>hibernate-core</artifactId>
43:   <version>4.3.6.Final</version>
44: </dependency>
45:
```

```
46: <dependency>
47:   <groupId>postgresql</groupId>
48:   <artifactId>postgresql</artifactId>
49:   <version>9.1-901.jdbc4</version>
50: </dependency>
51:
52: <dependency>
53:   <groupId>org.apache.commons</groupId>
54:   <artifactId>commons-pool2</artifactId>
55:   <version>2.2</version>
56: </dependency>
57:
58: <dependency>
59:   <groupId>org.apache.commons</groupId>
60:   <artifactId>commons-dbcp2</artifactId>
61:   <version>2.0.1</version>
62: </dependency>
63:
64: </dependencies>
```

# 5. Konzeption und Umsetzung der Ausarbeitungsplattform

Im letzten Kapitel wurden Methoden und Technologien vorgestellt, die für die Implementierung komplexer Multi Schichten Anwendungen verwendet werden können. Auf Basis der Analyse existierender Plattformen in Kapitel 2, werden in diesem Kapitel zunächst Aspekte der Konzeption und darauf folgend das Ergebnis der Implementierung vorgestellt. [Oestereich04]

## 5.1 Anforderungsanalyse

Für die Entwicklung eines Systems müssen zunächst einige grundlegende Aufgaben erfüllt werden. Dazu gehören die Formulierung der Idee und die Festlegung von Zielen.

Die Grundidee der in dieser Arbeit umgesetzten Plattform ist eine Fotoausarbeitungsplattform die es erlaubt Kunden über das Internet digitale Bilder in verschiedenen Formaten in gedruckter Form zu bestellen. Dafür wird eine grafische Benutzeroberfläche angeboten die eine einfache Interaktion mit dem System erlaubt. Für die Verwaltung von Bildmaterial wird ein Client zur Verfügung gestellt, der auch bei einer großen Anzahl von Bildern, einen benutzerfreundlichen Umgang ermöglicht. Bilder können mit diesem nicht nur verwaltet werden, sondern auch zusätzlich in einem Cloud Speicher abgelegt werden. Die Implementierung des Clients erlaubt darüber hinaus den Einsatz auf mehreren Computern und Geräten gleichzeitig. Dies ermöglicht die Zusammenarbeit von Gruppen die sich einen Zugang teilen können.

Aus der Systemidee lassen sich Anwendungsfälle ableiten die als Basis zur Erstellung einer Architektur und Implementierung dienen. Geschäftsanwendungsfälle definieren Abläufe unabhängig von einer konkreten Umsetzung. [Oestereich04] Die Attribute eines Anwendungsfalls sind Name, Beschreibung, Vorbedingung, Ergebnis und Akteure. Der Name gibt dem Anwendungsfall eine treffende Bezeichnung. Die Beschreibung erklärt die Aktivität bzw. den konkreten Inhalt des Vorgangs. Die Vorbedingung legt fest welche Umstände erfüllt sein müssen damit der Anwendungsfall durchgeführt werden kann. Akteure sind jene Personen die bei der

Aktivität beteiligt sind.  
Für das System dieser Arbeit wurden folgende Anwendungsfälle herausgearbeitet.

Name	Registrierung
Beschreibung	Ein Besucher möchte einen Zugang für die Ausarbeitungsplattform anlegen.
Vorbedingung	<ul style="list-style-type: none"> <li>• Besucher verfügt über Internet</li> <li>• Besucher verfügt über eine E-Mail Adresse</li> </ul>
Ergebnis	Es wurde ein Zugang für den Besucher im System angelegt und die Zugangsdaten wurden per E-Mail an den Kunden geschickt.
Akteure	Besucher

Name	Anmelden
Beschreibung	Der Kunde möchte sich Anmelden um mit der Plattform zu interagieren.
Vorbedingung	Der Kunde hat bereits einen Zugang zur Plattform registriert und kennt seine Zugangsdaten.
Ergebnis	Der Kunde ist auf der Plattform angemeldet.
Akteure	Besucher, Kunde;

Der Unterschied zwischen einem Besucher und einem Kunden ist, dass der Besucher dem System unbekannt ist. Ein Kunde verfügt allerdings über einen registrierten Zugang und kann sich damit am System anmelden.

Name	Album anlegen
Beschreibung	Der Kunde möchte für seine Bilder ein Album anlegen.
Vorbedingung	Der Kunde hat sich auf der Plattform angemeldet.
Ergebnis	Ein neues leeres Album wurde erstellt.

Akteure	Kunde
Name	Album löschen
Beschreibung	Der Kunde möchte ein vorhandenes leeres oder bereits mit Bildern gefülltes Album wieder löschen.
Vorbedingung	Ein Album wurde erstellt.
Ergebnis	Ein ausgewähltes Album wurde aus der Sammlung des Kunden gelöscht.
Akteure	Kunde

Name	Fotos hinzufügen
Beschreibung	Der Kunde möchte Fotos zu einem Album hinzufügen.
Vorbedingung	Es wurde bereits ein Album erstellt.
Ergebnis	Fotos wurden dem Album hinzugefügt und können wieder runtergeladen oder gelöscht werden.
Akteure	Kunde

Name	Fotos löschen
Beschreibung	Der Kunde möchte Fotos aus einem Album wieder löschen.
Vorbedingung	Es wurden bereits Fotos einem Album hinzugefügt.
Ergebnis	Ein oder mehrere Fotos wurden aus dem Album des Kunden gelöscht.
Akteure	Kunde

Name	Metadaten den Fotos hinzufügen
Beschreibung	Der Kunde möchte seine Fotos mit Namen, Beschreibung und Tags versehen.
Vorbedingung	Es wurden bereits Fotos einem Album hinzugefügt.
Ergebnis	Ein Foto wurde mit einem Namen, einer Beschreibung sowie mit Tags versehen.
Akteure	Kunde

Name	Metadaten dem Album hinzufügen
Beschreibung	Der Kunde möchte seine Alben nach Themen unterscheiden und dafür diese mit einer Beschreibung versehen.
Vorbedingung	Ein Album wurde angelegt.
Ergebnis	Das Album wurde mit einer Beschreibung versehen.
Akteure	Kunde

Name	Fotos veröffentlichen
Beschreibung	Der Kunde möchte seine Fotos im Internet veröffentlichen.
Vorbedingung	Fotos wurden in ein Album hochgeladen.
Ergebnis	Ein Album wurde für die Veröffentlichung markiert.
Akteure	Kunde

Name	Album durchsehen
Beschreibung	Der Kunde möchte die von ihm in ein Album zugeordneten Fotos durchsehen.
Vorbedingung	Fotos wurden einem Album hinzugefügt.
Ergebnis	Ein Album wurde durchsucht.
Akteure	Kunde

Name	Fotos zur Ausarbeitung auswählen
Beschreibung	Der Kunde möchte seine Alben durchsuchen nach Fotos die er zur Ausarbeitung bestellen will.
Vorbedingung	Fotos wurden einem Album hinzugefügt.
Ergebnis	Fotos wurden durchgesehen.
Akteure	Kunde

Name	Ausarbeitungsoptionen wählen
Beschreibung	Der Kunde möchte für ein von ihm gewähltes Foto oder Album bestimmte Ausarbeitungsoptionen festlegen.
Vorbedingung	Fotos wurden durchsucht.
Ergebnis	Ausarbeitungsoptionen für ein Foto oder Album wurden ausgewählt.
Akteure	Kunde

Name	Foto oder Album der Mappe hinzufügen
Beschreibung	Der Kunde möchte ein Foto oder ein Album mit den von ihm gewählten Ausarbeitungsoptionen seiner Bestellmappe hinzufügen.
Vorbedingung	Ausarbeitungsoptionen wurden festgelegt.
Ergebnis	Fotos wurden der Fotomappe hinzugefügt.
Akteure	Kunde

Name	Fotos aus der Mappe entfernen
Beschreibung	Der Kunde möchte ein oder mehrere Fotos aus der Fotomappe entfernen.
Vorbedingung	Fotos wurden der Fotomappe hinzugefügt.
Ergebnis	Fotos wurden aus der Mappe entfernt.
Akteure	Kunde

Name	Fotomappe bestellen
Beschreibung	Der Kunde will gerne seine Fotomappe bestellen.
Vorbedingung	Fotos wurden der Fotomappe hinzugefügt.
Ergebnis	Fotomappe wurde bestellt.
Akteure	Kunde



Name	Fotos lokal speichern
Beschreibung	Der Kunde möchte seine auf die Plattform hochgeladenen Fotos lokal auf seinem Gerät sichern.
Vorbedingung	Fotos wurden einem Album hinzugefügt.
Ergebnis	Fotos wurden heruntergeladen.
Akteure	Kunde

Name	Sehr große Fotos hinzufügen
Beschreibung	Der Kunde möchte Fotos mit einer überdurchschnittlichen Dateigröße einem Album hinzufügen.
Vorbedingung	Der lokale Cloud Client wurde installiert und die Anmeldung am System ist erfolgt.
Ergebnis	Sehr große Fotos wurden dem System hinzugefügt.
Akteure	Kunde

Die erwähnten Anwendungsfälle definieren im Großen die benötigten Funktionen der Plattform. Basierend auf diesen Funktionen und der Systemidee wurde eine Prototypenmaske mit einem Zeichenprogramm angefertigt. Sie dient als Orientierung für die Entwicklung. Eine solche Maske dient auch zur Motivation und lässt eventuelle Anwendungsprobleme schon früh in der Entwicklung erkennen.



Abbildung 11: Skizze der Grafischen Benutzeroberfläche

## 5.2 Systemarchitektur

Das System besteht in der engeren Definition aus drei Komponenten. Einem zentralen Serversystem, einer Single-Page Web Applikation und einem Desktop Client.

Als Architektur für den Server wurde eine 3-Schichtenmodell implementiert. Die Präsentationsschicht wurde auf der Basis des Spring MVC Musters entwickelt. Sie gliedert sich in zwei unterschiedliche Typen. Controller für die Präsentation von HTML Inhalten und Controller für die API die nur Rohdaten im JSON Format liefern. Front Controller empfangen die Anfragen, bereiten diese auf und delegieren diese an die Business Schicht. In dieser wurden Service Klassen implementiert die entsprechende Operationen auf die Domain Objekte ausführen und entsprechende Antworten für den Front Controller liefern. In der Persistenz-Schicht kommt Hibernate zum Einsatz. Das Mapping für die Domain Objekte wurde mit XML erstellt und am Klassenpfad in einem eigenen Package abgelegt.



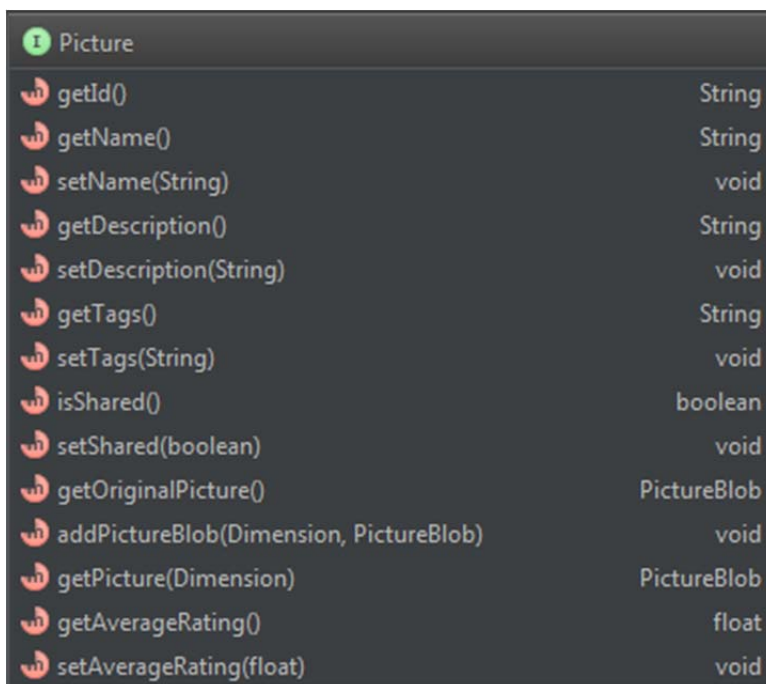
Abbildung 12: Server-Architektur

Die Single-Page Web-Applikation gliedert sich grundlegend wieder in drei Schichten. Die Präsentation-Schicht besteht aus der Browser-Umgebung. Sie stellt die Komponenten visuell dar und filtert ausgelöste Events. Die Logik-Schicht ist in Java Script implementiert. Sie empfängt die Interaktionen des Benutzers (gekapselt in Events) und verarbeitet sie entsprechend. Die unterste und dritte Schicht stellt die Kommunikation mit dem Server sicher und verpackt Anfragen in serialisierten JSON Text sowie deserialisiert Antworten vom Server in JSON Objekte.

Die Desktop Applikation besteht wiederum aus drei Schichten. Einer Präsentations-Schicht und einer Logik-Schicht. Die unterste Schicht besteht aus den Teilen Kommunikation und lokale Persistenz. Die Kommunikations-Schicht kommuniziert mit dem entfernten Server. Die lokale Persistenz-Schicht speichert Einstellungen und temporäre Daten. Informationen über Alben und Bilder werden jedoch am Server gespeichert. Entscheidet die Logik diese Daten zu laden, weil sie eventuell in der Präsentations-Schicht benötigt werden, so bedient sie sich der Kommunikations-Schicht.

Die wichtigsten Domain Objekte die in allen drei Teilen, Server, Web und Desktop-Client, vorkommen, sind Picture, Album und User.

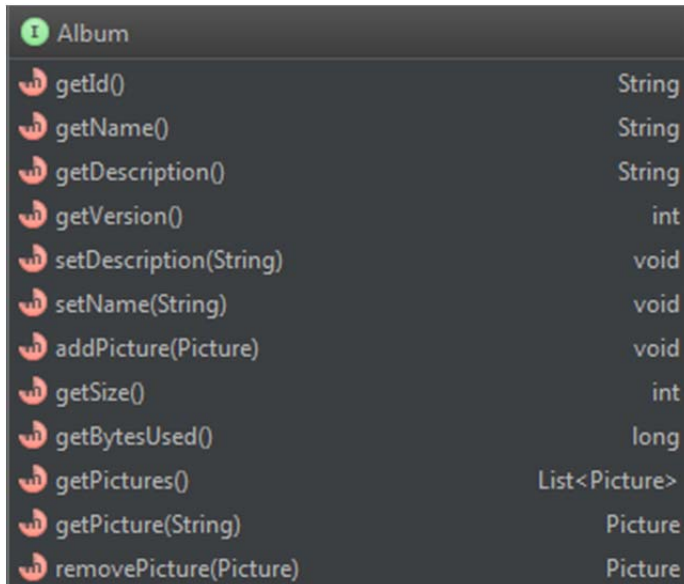
Das Picture-Objekt kapselt Informationen zu den einzelnen Bildern eines Kunden. Abgespeichert werden darin nicht nur die binäre Repräsentation sondern auch Metadaten wie Beschreibung, Tags oder Bewertung.



Method	Return Type
getId()	String
getName()	String
setName(String)	void
getDescription()	String
setDescription(String)	void
getTags()	String
setTags(String)	void
isShared()	boolean
setShared(boolean)	void
getOriginalPicture()	PictureBlob
addPictureBlob(Dimension, PictureBlob)	void
getPicture(Dimension)	PictureBlob
getAverageRating()	float
setAverageRating(float)	void

Abbildung 13: Schnittstelle für das Picture Domain Objekt

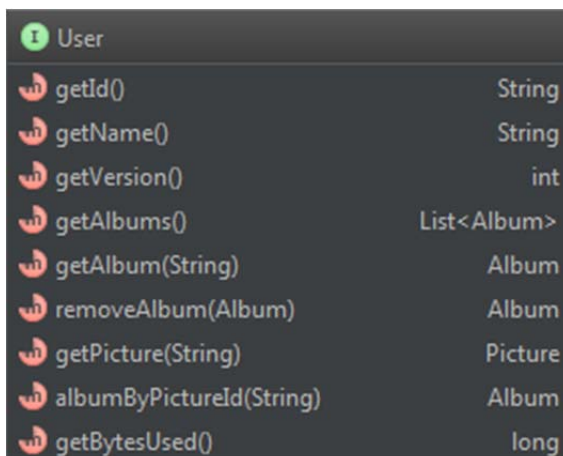
Das Album Domain-Objekt kapselt alle Daten eines vom Benutzer angelegten Albums. Es dient auch als Container-Objekt für Bilder. Jedem Album kann darüber hinaus eine Beschreibung gegeben werden



Album	
getId()	String
getName()	String
getDescription()	String
getVersion()	int
setDescription(String)	void
setName(String)	void
addPicture(Picture)	void
getSize()	int
getBytesUsed()	long
getPictures()	List<Picture>
getPicture(String)	Picture
removePicture(Picture)	Picture

Abbildung 14: Schnittstelle für das Album Domain Objekt

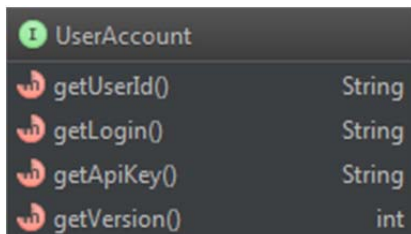
Das User Objekt kapselt Daten über den Benutzer und dient als Container von Alben.



User	
getId()	String
getName()	String
getVersion()	int
getAlbums()	List<Album>
getAlbum(String)	Album
removeAlbum(Album)	Album
getPicture(String)	Picture
albumByPictureId(String)	Album
getBytesUsed()	long

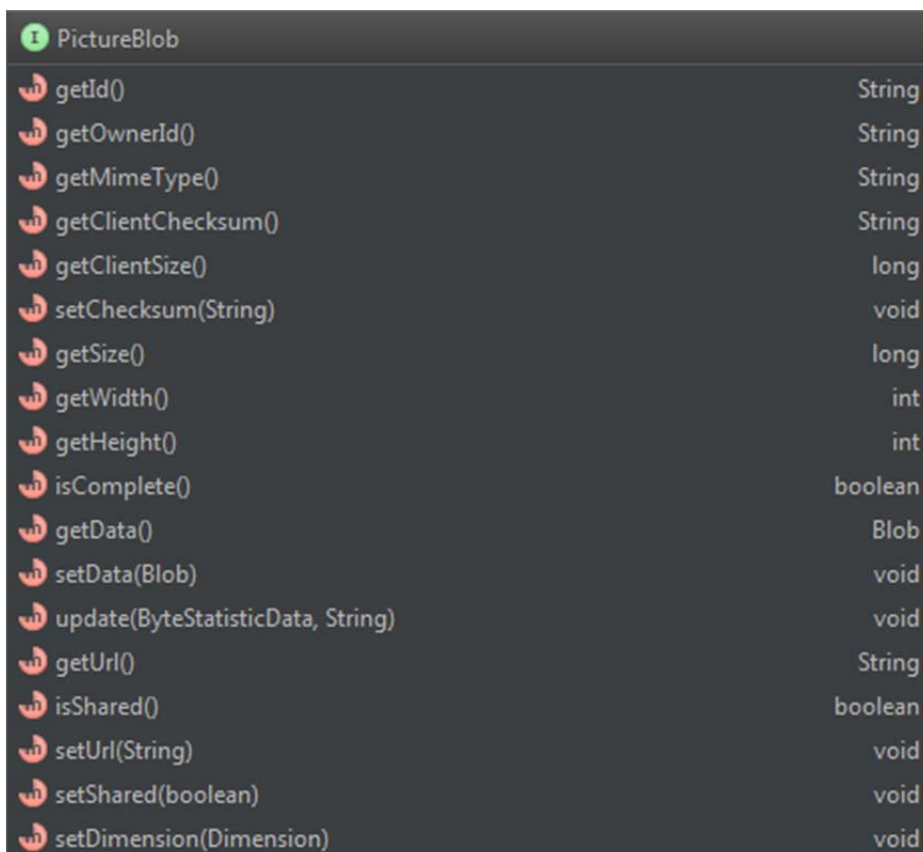
Abbildung 15: Schnittstelle für das User Domain Objekt

Für die Implementierung des Serversystems sind zwei weitere wichtige Objekte von Bedeutung. Das UserAccount- und das PictureBlob-Objekt. Die Zugangsdaten eines Benutzers sowie dessen API-Schlüssel werden im UserAccount-Objekt gekapselt. Die binären Bild Daten werden im PictureBlob Objekt abgelegt. Für jede verfügbare Dimension besitzt das Picture-Objekt ein eigenes Blob-Objekt.



UserAccount	
getId()	String
getLogin()	String
getApiKey()	String
getVersion()	int

Abbildung 16: Schnittstelle für das UserAccount Domain Objekt



PictureBlob	
getId()	String
getOwnerId()	String
getMimeType()	String
getClientChecksum()	String
getClientSize()	long
setChecksum(String)	void
getSize()	long
getWidth()	int
getHeight()	int
isComplete()	boolean
getData()	Blob
setData(Blob)	void
update(ByteStatisticData, String)	void
getUrl()	String
isShared()	boolean
setUrl(String)	void
setShared(boolean)	void
setDimension(Dimension)	void

Abbildung 17: Schnittstelle für das PictureBlob Domain Objekt

## **5.3 Verwendete Werkzeuge und Technologien**

Die für die Implementierung verwendeten Werkzeuge und Technologien werden im Folgenden kurz vorgestellt.

### **5.3.1 IntelliJ IDEA**

IDEA ist eine integrierte Entwicklungsumgebung (IDE) von der Firma JetBrains s.r.o. Aktuell liegt die Umgebung bereits in Version 14 vor und bietet umfangreiche Unterstützung für die Entwicklung komplexer Applikationen. Für die Web-Entwicklung wird neben der Unterstützung von HTML, CSS und Java Script auch Frameworks wie Spring, Play, Grails, Flex, Node.js, HTML5, AngularJS, jQuery, less und Coffee Script unterstützt. IDEA unterstützt auch die gängigsten Enterprise Technologien und Server sowie Web-Container wie zum Beispiel Glass Fish, Spring, Hibernate, Tomcat, JBoss und Web Sphere. [JetBrains14]

Der große Vorteil dieser Entwicklungsumgebung gegenüber freien IDEs wie Eclipse oder Netbeans ist, dass alle etablierten Technologien ohne die Verwendung und Konfiguration von Plugins unterstützt werden. Damit geht einher, dass das Gesamtpaket abgestimmt ist und perfekt zusammen spielt. Weiters ist zu erwähnen, dass die Umgebung viel durchdachter und benutzerfreundlicher ist als jede andere. Aufgaben werden im Hintergrund durchgeführt, Optimierungen im Code werden vorgeschlagen und Navigation zwischen den unterschiedlichen Teilen der Applikation wird durch Kontext sensitive Links erleichtert.

### **5.3.2 Apache Chainsaw**

Chainsaw ist eine Java basierte Applikation zur Überwachung von Log-Nachrichten. Bei der Entwicklung moderner Applikationen können spezifische Frameworks eingesetzt werden die es erlauben, in Echtzeit, Statusmeldungen zu erzeugen. Diese Frameworks können extern konfiguriert werden. Als Ziel für diese Nachrichten kann eine Datei auf einer Festplatte oder auch ein Logserver in einem Netzwerk angegeben werden. Chainsaw bietet auch die Möglichkeit des Logserver-Betriebs. Eingehende Nachrichten können nach Paketen gefiltert und durch unterschiedliche Farben markiert werden. Das erleichtert das Auffinden von Fehlernachrichten oder bestimmter gesuchter Meldungen sehr. [Chainsaw14]

### **5.3.3 Telerik Fiddler**

Fiddler ist ein Proxy der zwischen den Browser und einem Web-Server geschaltet wird und jede Anfrage und Antwort die an den Server bzw. von diesem gesendet wird, visualisiert. Durch intelligente Filter, die der Entwickler selber konfigurieren kann, können nur bestimmte Anfragen gefiltert werden. Eine Replay-Funktion erlaubt das erneute senden einer Anfrage, ohne dass diese durch einen Browser ausgelöst werden muss. Visualisierungsunterstützungen bieten den Komfort, den Inhalt der Anfrage oder der Antwort, nach unterschiedlichen Aspekten zu analysieren. So können JSON Objekte die als Text übertragen werden in einem Baum angezeigt werden. [Telerik14]

### **5.3.4 Oracle JavaFX Scene Builder**

Der Scene-Builder von Oracle erlaubt es grafische Benutzeroberflächen für JavaFX Anwendungen nach dem WYSIWYG (What You See Is What You Get)-Prinzip zu gestalten. Alle in JavaFX vorhanden Komponenten können per Drag & Drop zu einer fertigen Oberfläche kombiniert werden. Auch die Einbindung eigener erstellter Komponenten ist möglich. [Oracle14]

### **5.3.5 Regex Buddy**

Regex Buddy ist ein Programm das die Erstellung von regulären Ausdrücken erleichtert. Dazu stehen Unterstützungen für die Formulierung, sowie für das Auffinden von Fehlern in solchen zur Verfügung. Reguläre Ausdrücke werden in Applikationen oft für die Validierung von Werten verwendet, aber auch für die Extraktion von bestimmten Daten aus einer Datenbasis, so zum Beispiel bei der Aufgabe des Web-Crawlings. [RegexBuddy14]

### **5.3.6 Adobe Photoshop**

Photoshop ist der Marktführer der Bild Bearbeitung. Das Einsatzgebiet ist jedoch weitaus umfassender. Mit Photoshop können mittlerweile nicht nur 2D Rastergrafiken sondern auch komplexe 3D-Szenen erzeugt werden. Sehr Hilfreich ist die Applikation auch bei der Erstellung von Designs für Web-Seiten sowie zur Erstellung grafischer Komponenten von Benutzeroberflächen.



### **5.3.7 SQL Lite Database Browser**

Diese Applikation kann verwendet werden um SQL Lite Datenbanken zu durchsuchen und zu modifizieren. Eine grafische Benutzeroberfläche beschleunigt das Arbeiten mit der Datenbank da keine SQL Abfragen mehr manuell geschrieben werden müssen. [SqlLite14]

### **5.3.8 Java 8 und JavaFX**

Mit Java erfolgte die Implementierung sowohl der Desktop-Applikation als auch der Web-Applikation. Java ist eine ursprünglich von Sun Microsystems entwickelte Programmiersprache. Mittlerweile wird die Sprache von Oracle weiterentwickelt und bietet neben dem ursprünglich rein objektorientierten Paradigma auch die Möglichkeit funktionale Methoden anzuwenden.

### **5.3.9 Apache Tomcat 8 Embedded, JSP, JSTL und EL**

Apache Tomcat ist ein Servlet-Container der vollständig kompatibel zur Java-Servlet Spezifikation ist. Er wird sehr gerne auch in Java EE Servern als Web-Engine eingesetzt. Die Implementierung des Servers dieser Applikation wurde mithilfe der einbettbaren Version von Tomcat 8 entwickelt. Dieser erlaubt eine vollständig programmatische Konfiguration des Containers und der Applikation.

Teile der Applikation wurden mit Java Server Pages (JSP) unter Zuhilfenahme der Java Standard Tag Library (JSTL) und der Unified Expression Language (EL) umgesetzt. Die JSP Technologie ist angelehnt an HTML und erweitert das Konzept um dynamische Elemente, die es erlauben, datenlastige Web-Seiten zu entwickeln. Die JSTL Bibliothek kann in Verbindung mit JSP verwendet werden und bietet eine Sammlung häufig benötigter Funktionen. Die Unified Expression Language erweitert das Konzept von JSTL und bietet eine kompaktere Schreibweise für die Verwendung von Funktionen. Auch der Zugriff auf Java Beans und Serverseitige Komponenten wird durch die EL erleichtert.

### **5.3.10 Spring**

Die Umsetzung der Server Architektur wurde mit Hilfe des Spring-Frameworks durchgeführt. Das Spring-Framework wurde bereits im vorangegangenen Kapitel erklärt.

### **5.3.11 Hibernate**

In der Serverseitigen Persistenz-Schicht wurde Hibernate verwendet. Auch Hibernate wurde bereits im vorangegangenen Kapitel erklärt.

### **5.3.12 Hypersonic SQL**

Hypersonic SQL ist ein relationales Datenbanksystem das vollständig in Java implementiert ist. Es unterstützt den eingebetteten Betrieb. Auch den Betrieb als Server und unterstützt die Java Database Connectivity (JDBC) Schnittstelle. Tabellen können temporär im Speicher oder persistent auf der Festplatte abgelegt werden. HSQL wird seit 2001 entwickelt und liegt mittlerweile in Version 2.3.1 vor. [Hsql14]

Der Serverteil der Applikation verwendet zur Persistierung der Domain-Objekte durch Hibernate als Datenspeicher das HSQL-System.

### **5.3.13 Hikari DBCP**

Der Hikari Database Connection Pool ist, wie der englische Name schon sagt, ein Pool für Datenbankverbindungen. Er gehört zu den schnellsten am Markt befindlichen kostenfreien Pools und kann mit den gängigsten Datenbank-Systemen verwendet werden. [Hikari14]

Der Server der Applikation verwendet diesen Pool als Verbindungspool für die Hibernate Session Factory.

### **5.3.14 SQLite**

SQL Lite ist ein in C programmiertes, im Jahr 2000 entstandenes, relationales Datenbanksystem. Mittlerweile in der Version 3.8.7.4 bietet es auch Unterstützung für die JDBC Schnittstelle. [SqlLite1402]

Dieses System verwendet die Desktop Applikation für die Speicherung von Konfigurationsdaten.

### **5.3.15 Simple Logging Facade und Log4j**

Log4j ist ein Logging Framework. Es erlaubt Nachrichten aus der Anwendungslogik zu filtern und in extern konfigurierte Ziele weiter zu leiten. Ziele können dabei Dateien, Drucker, Log-Server und vieles mehr sein. [Log4j14]

Die Simple Logging Facade (SL4J) ist eine Abstraktion die es erlaubt das verwendete Logging Framework auch nach der Implementierung der Anwendungslogik zu tauschen. [SimpleLog4j14]

### **5.3.16 JUnit**

JUnit ist ein von Kent Beck und Erich Gamma entwickeltes Framework zum Testen von Softwarecode. Oft verwechselt mit der ägyptischen Gottheit Iunit und ursprünglich für die Sprache Java entwickelt, existieren mittlerweile Portierungen in einer Unmenge an Programmiersprachen. [JUnit14]

Um Fehler bei Änderungen vorzubeugen wurden für die Anwendungslogik dieser Applikation viele JUnit Testfälle erstellt.

### **5.3.17 Apache Maven**

Apache Maven ist eigentlich ein Framework für die Unterstützung des gesamten Projektentwicklungsprozesses, wird jedoch vielmehr lediglich als Build-Automatisierung eingesetzt. [Maven14]

Für die Abhängigkeitsverwaltung der Applikation dieser Arbeit wurde Maven verwendet.

### **5.3.18 JQuery**

JQuery ist eine populäre Java Script Bibliothek die die Clientseite-Entwicklung von Web-Applikationen erheblich vereinfacht. [Jquery14]

Die Single Page Web Applikation verwendet JQuery zur Gestaltung der Interaktion sowie zur Visualisierung von Animationen.

## 5.4 Entwicklung grafischer Komponenten mit JavaScript

Der im Abschnitt 5.1 gezeigte Prototyp zeigt grafische Komponenten, die die in den Anwendungsfällen geforderten Funktionen zur Verfügung stellen. Diese Elemente wurden zunächst mit Java Script entwickelt und getestet. Es handelt sich dabei um drei Komponenten die das Interagieren mit der Applikation möglichst angenehm für den Kunden gestalten soll.

Zum Durchsuchen der Bilder eines Albums wurde ein vertikaler Sucher entwickelt, der es erlaubt, die Bilder zu durchblättern und nach verschiedenen Reihenfolgen zu sortieren.

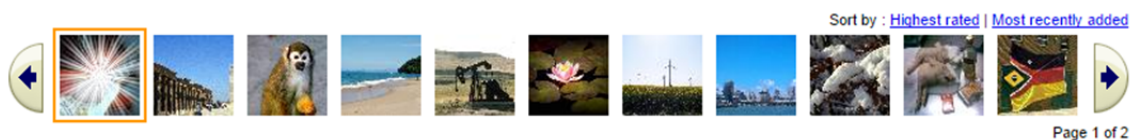


Abbildung 18: Pager Java Script Komponente

Der Sucher zeigt auf einer vertikalen Bühne den Inhalt des Albums. Jedes Bild wird durch eine verkleinerte 58 x 58 Pixel-Version dargestellt. Das Aktuell ausgewählte Bild wird durch einen gelben Rand markiert. Die Komponente verfügt auch über einen Selektionsmechanismus, der das Auswählen mehrere Bilder zugleich erlaubt. Links und rechts der Komponenten werden Schaltflächen zum Blättern eingeblendet. Die Reihenfolge der Bilder kann, Abhängig von einem konfigurierten Sortieralgorithmus, verändert werden. Das Blättern selbst, so wie das Sortieren, wird animiert visualisiert. Die Komponente wurde als Java Script Objekt realisiert und erlaubt eine einfache Konfiguration und Wiederverwendbarkeit in anderen Projekten.

Das folgende Beispiel zeigt die Verwendung der Komponente:

```
01: // Image Pager
02: var pager = new ImagePager('pagerContainer', 'myPager');
03: pager.setResourcePath("/images/");
04: pager.setSlotCount(11);
05:
06: // add sort option
07: var func = function (a, b)
08: {
09:     if (!a.options || !b.options) return 0;
10:     if (!a.options.ml || !b.options.ml) return 0;
```

```

11:     if (a.options.m1 == b.options.m1) return 0;
12:     return a.options.m1 < b.options.m1 ? -1 : 1;
13: };
14: pager.addSortOption("Highest rated", func);

15: pager.setCustomPageDataFunction(null);
16: pager.setEmptySlotClickHandler(function ()
17: {
18:     var data = $('#' +
    aim.nodes.currentAlbumNameHeadlineId).prop('album');
19:     var albumId = data && data.album ? data.album.id : null; //
    obtain album id
20:     if (albumId) aim.eventAddAlbumPicture(albumId);
21: });

22: pager.display();

```

Bevor die Komponente konfiguriert werden kann muss zunächst eine Instanz erzeugt werden. Zeile 2 erstellt eine Instanz und gibt dem Konstruktor zwei Parameter mit. Durch den ersten Parameter wird definiert in welchem Container die Komponenten dargestellt werden sollen. Der Wert muss dabei eine ID eines DOM Elements sein. Durch den zweiten Parameter wird den Komponenten ein Name zugewiesen der auch als ID verwendet wird. Damit der Sucher die Bildvorschau erstellen kann, wird in Zeile 3 ein Pfad konfiguriert unter welchem die binären Bilddaten abgerufen werden können. Zeile 4 definiert die Breite bzw. die Anzahl der Bilder die vertikal dargestellt werden können.

Da es möglich ist, die Bilder des Suchers in unterschiedlichen Reihenfolgen anzeigen zu lassen, muss den Komponenten eine Sortierfunktion gegeben werden. Diese wird, im funktionalen Stil, auf die Liste der Bilder angewendet. Zeile 7 - 13 definiert eine solche Sortierfunktion. Empfangen werden 2 Objekte, die unter dem Attribute „options“ ein Sortierattribut m1 besitzen. Die genauere erforderliche Struktur kann im Quellcode der Komponente nachgesehen werden. Zeile 14 übergibt die Funktion schließlich der Komponente. Nachdem auf eine andere Seite geblättert wurde muss, der Sucher die Bilder der neuen Seite auf der vertikalen Bühne darstellen, und Änderungen am internen Zustand der Komponente vornehmen. Nachdem dieser Prozess abgeschlossen ist, gibt es die Möglichkeit, eine Benachrichtigung in einer Callback Funktion zu erhalten. In Zeile 15 könnte so ein Callback konfiguriert werden, in diesem Beispiel wurde dies jedoch ausgelassen, daher wird als Parameter-Wert Null übergeben. Wird auf eine Seite geblättert, die nicht voll befüllt ist so wird in den leeren Slots der Bühne ein Platzhalter angezeigt. Diesen Platzhaltern kann ein Ereignis-Handler zugewiesen werden der beim

Klick auf den freien Slot ausgeführt wird. Zeile 16 - 21 zeigt die Implementierung eines solchen. Beim Klick auf einen leeren Slot wird erwartet, dass der Kunde dem Album ein neues Foto hinzufügen möchte. Dafür wird in Zeile 18 zunächst das aktuell gezeigte Album ermittelt. In Zeile 19 wird die ID des Album festgestellt, und in Zeile 20 der Prozess zum Hinzufügen eines Fotos eingeleitet. Abschließend wird in Zeile 20 die Komponente angewiesen den Inhalt im angegeben Container darzustellen.

Eine weitere Komponente die für den Web-Client entwickelt wurde, ist der Bild Browser. Mit diesem ist es möglich Bilder des Albums durchzublättern.

**abstrakt\_0020.jpg**



tags: sonne gitarre rot  
rated: 3.56/5.00, sonnen gitarre

**Abbildung 19: Bild Browser Komponente**

Der Browser wird mit Bildern befüllt und erlaubt es dem Kunden diese in der Füllreihenfolge, von links nach rechts oder von rechts nach links, durchzusehen. Nach jeder Änderung des gezeigten Bildes erfolgt ein Aufruf an einen konfigurierten Callback. Meist agiert der Callback als Mediator (siehe Mediator Entwurfsmuster [Gamma14]), der weitere Ereignisse veranlasst, wie zum Beispiel das Anzeigen der Metainformationen eines Bildes. In Abbildung 19 wird dies unter dem Bild dargestellt. Die Pfeile zum Blättern, bzw. die Schaltflächen, werden nur eingeblendet sofern der Kunde den Mauszeiger darüber bewegt.

Die Konfiguration der Komponente wird in folgendem Beispielcode gezeigt:

```
01: // Image Stage
02: var stage = new ImageStage("stageCell", "myStage");
03: stage.setResourcePath("/images/");
04: stage.display();
```

Wiederum muss von der Komponente zunächst eine Instanz erzeugt werden. Zeile 2 gibt dem Konstruktor zwei Parameter mit. Der erste Parameter ist eine ID, die definiert in welchem HTML Element der Browser dargestellt werden soll. Der zweite Parameter gibt der Komponente einen Namen, der auch als ID verwendet wird. Damit die Bilder auch dargestellt werden können, wird in Zeile 3 dem Browser ein Pfad gegeben, unter welchem die Binärdaten der Bilder abgerufen werden können. Zeile 4 schließlich, visualisiert die Komponente.

In diesem Beispiel wurde nur das initiale Anlegen der Komponenten gezeigt. Danach muss diese noch mit Daten befüllt werden. Dafür stehen einige Methoden zur Verfügung, wovon einige im folgenden Beispiel gezeigt werden:

```
01: stage.setOnShowHandler(callback);
02: stage.addImage(picture.id, thumbnailUrl, picture);
03: stage.clear(null, callback);
04:
05: stage.setAttribute("albumId", albumId);
06: stage.removePicWithAttribute(key, value);
07: stage.moveToPicWithAttribute(key, value);
```

Der Bild-Browser erlaubt die Registrierung eines Callbacks der jedes Mal wenn zu einem neuen Bild gebrowsst wurde, benachrichtigt wird. Oftmals ist dies ein Mediator, der weitere Aktivitäten veranlasst, wie zum Beispiel das Update der zuvor erwähnten Pager Komponente. Zeile 1 zeigt die Methode zur Registrierung des Mediators.

In Zeile 2 wird gezeigt wie Bilder hinzugefügt werden und wie diese wieder entfernt werden können, in Zeile 3. Der Aufruf der Methode `addImage` erwartet drei Parameter. Der erste Parameter definiert eine eindeutige ID für das Bild, der zweite Parameter den URL und der dritte Parameter erlaubt die Assoziation des Bildes mit einem benutzerdefinierten Objekt. In diesem Objekt können Daten (Attribute) gekapselt werden die dann nach dem Auslösen des Callbacks in diesem abgefragt werden können. Die Methode `Call` kann mit 2 optionalen Parametern aufgerufen werden. Durch den ersten Parameter wird angewiesen ob das aktuell gezeigt Bild ausgeblendet werden soll bevor alle Bilder entfernt werden, und der zweite Parameter kann eine Callback Funktion sein die aufgerufen wird sobald alle Bilder entfernt wurden. Die Bild Browser Komponente kann auch mit beliebigen Attributen befüllt werden. Zeile 5 zeigt wie Attribute hinzugefügt werden können. Nicht zu verwechseln mit den Attributen eines Bildes, wie bereits in Zeile 2 gezeigt. Sollen einzelne Bilder entfernt oder angezeigt werden kann das auch auf Basis der Bild Attribute programmatisch erfolgen. Zeile 6 zeigt wie ein Bild das über ein bestimmtes Attribut verfügt, entfernt werden kann. Zeile 7 zeigt, wie ein Bild mit einem bestimmten Attribut, angezeigt wird. Neben den besprochenen Methoden verfügt die Komponente noch über Weitere, die die Benutzung vereinfachen. Diese können im Quellcode eingesehen werden.

Die dritte wichtige Komponente die für den Web-Client entwickelt wurde ist die Fotomappe oder Folder Komponente. Sie agiert als Warenkorb und sammelt die Bilder die der Kunde gerne ausarbeiten lassen möchte.

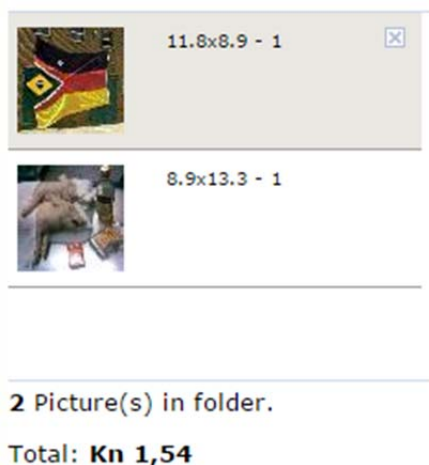
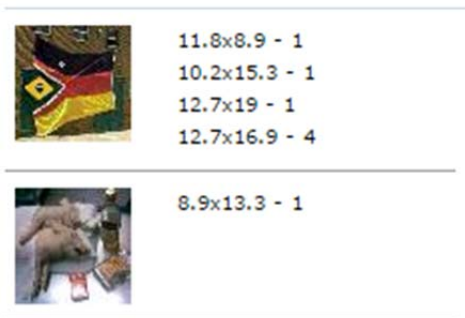


Abbildung 20: Foto Mappe Java Script Komponente



Für jedes Foto, das der Kunde der Mappe hinzufügt, wird ein Eintrag in der Mappe abgelegt. Wird ein Foto in unterschiedlichen Formaten bestellt, so wird dem Eintrag für jedes Format zusätzlich, ein Format Eintrag hinzugefügt. Die folgende Abbildung zeigt wie dies aussieht:



---

**8 Picture(s) in folder.**

**Total: Kn 29,16**

Abbildung 21: Fotomappe mit Bilder in unterschiedlichen Formaten

Jeder Formateintrag kann auch wieder entfernt werden. Die anderen Formate und Bilder bleiben davon unberührt. Entscheidet sich der Kunde die Web Seite zu verlassen, ohne seinen Warenkorb zu bestellen, so wird dieser mit Hilfe der HTML5 Local Storage Technologie persistiert und automatisch beim nächsten Besuch wieder geladen.

Das folgende Beispiel zeigt die Instanziierung der Fotomappe:

```
01: // create folder
02: var folder = new ImageFolder('folder1');
03: folder.setOnImageClickHandler(aim.eventClickOnFolderImage);
04: folder.setOnFormatClickHandler(aim.eventClickOnFolderImageFormat
);
05: folder.setOnUpdateHandler(aim.updateFolderTotals);
06:
07: folder.display();
```

In Zeile 2 wird eine Instanz der Fotomappe angelegt und im Konstruktor wiederum definiert, in welchem HTML Element die Komponente dargestellt werden soll. Darauf folgen eine Reihe von Callback Konfigurationen. Zeile 3 registriert einen Callback, der

aufgerufen wird, sobald der Anwender auf das verkleinerte Bild klickt. Meist ist dieser Callback wiederum ein Mediator, der veranlasst, dass das Album des Bildes geladen wird und zu dem entsprechenden Bild im Bild-Browser geblendet wird. Zeile 4 registriert einen Callback, der aufgerufen wird, sobald der Anwender auf einen Formateintrag eines Bild Eintrags klickt. Im Kontext dieser Applikation löst dies einen Mediator aus der die Ausarbeitungsoptionen des Format Eintrages in der GUI Maske ident einstellt. Zeile 5 registriert einen Callback, der aufgerufen wird, sobald ein Bild oder Format Eintrag hinzugefügt oder entfernt wurde. Das folgende Beispiel soll nun noch zeigen wie Bilder der Fotomappe hinzugefügt werden können und welche anderen Methoden zur Verfügung stehen:

```
01: folder.add(image.id, imageUrl, picture);
02: folder.addFormatData(image.id, width, height, pcs, price,
    printOptions);
03: folder.removeImageByParameter("id", image.id, callback);
04: folder.hasImage(image.id);
05: folder.getImageCount();
06: folder.getFolderValue();
07: folder.updateDisplay();
```

Die add Methode, in Zeile 1, dient dem Hinzufügen von Bildern. Sie erwartet drei Parameter. Der erste definiert eine eindeutige ID für das Bild, der Zweite den URL zur Miniaturversion des Bildes und der Dritte ist ein optionales Attribut Objekt. Zeile 2 zeigt wie durch die addFormatData-Methode ein zusätzlicher Formateintrag hinzugefügt werden kann. Diese Methode erwartet eine Reihe von Parametern. Die Bild ID des Bildes, die Abmessungen sowie die Stückzahl, den Preis und die Ausarbeitungsoptionen, gekapselt in einem Objekt. Zeile 3 zeigt wie Bildeinträge wieder aus der Fotomappe entfernt werden können. Die Parameter Liste der Methode removeImageByParameter besteht aus drei Parametern. Der erste definiert den Namen des Attributs das ein Bild besitzen muss, der zweite den Attribut-Wert und der dritte Parameter ist ein Callback, der aufgerufen wird sobald das Bild entfernt wurde. Das Attribut Schlüssel – Wert Paar muss zuvor definiert worden sein. Nach Entfernung des Bildes wird die übergebene Callback Methode aufgerufen. Zeile 4 zeigt die Verwendung der Methode hasImage, die prüft ob ein Bild mit einer bestimmten ID, in der Mappe enthalten ist. Die Methode getImageCount in Zeile 5 retourniert die Anzahl der Bilder und die Methode getFolderValue retourniert den Wert der gesamten Fotomappe. Wurden Bilder entfernt oder hinzugefügt, so muss danach die Methode

updateDisplay aufgerufen werden, um die Darstellung der Fotomappe am Bildschirm zu erneuern.

Die drei vorgestellten Komponenten ermöglichen die wichtigsten Interaktionen der Web Applikation. Sie sind in der Implementierung jeweils in drei separaten JavaScript-Dateien aufgeteilt, sodass auch eine Verwendung außerhalb dieser Arbeit denkbar wäre. Im folgenden Abschnitt wird der Web Client im Detail vorgestellt.

## 5.5 Der Web Client

Der Web Client wurde als Single Page Applikation umgesetzt. Damit Besucher Zugang zu diesem erhalten, ist zunächst eine Registrierung notwendig. Die folgende Abbildung zeigt den Willkommens Bildschirm der auch die Maske für die Registrierung und für das Anmelden beinhaltet.

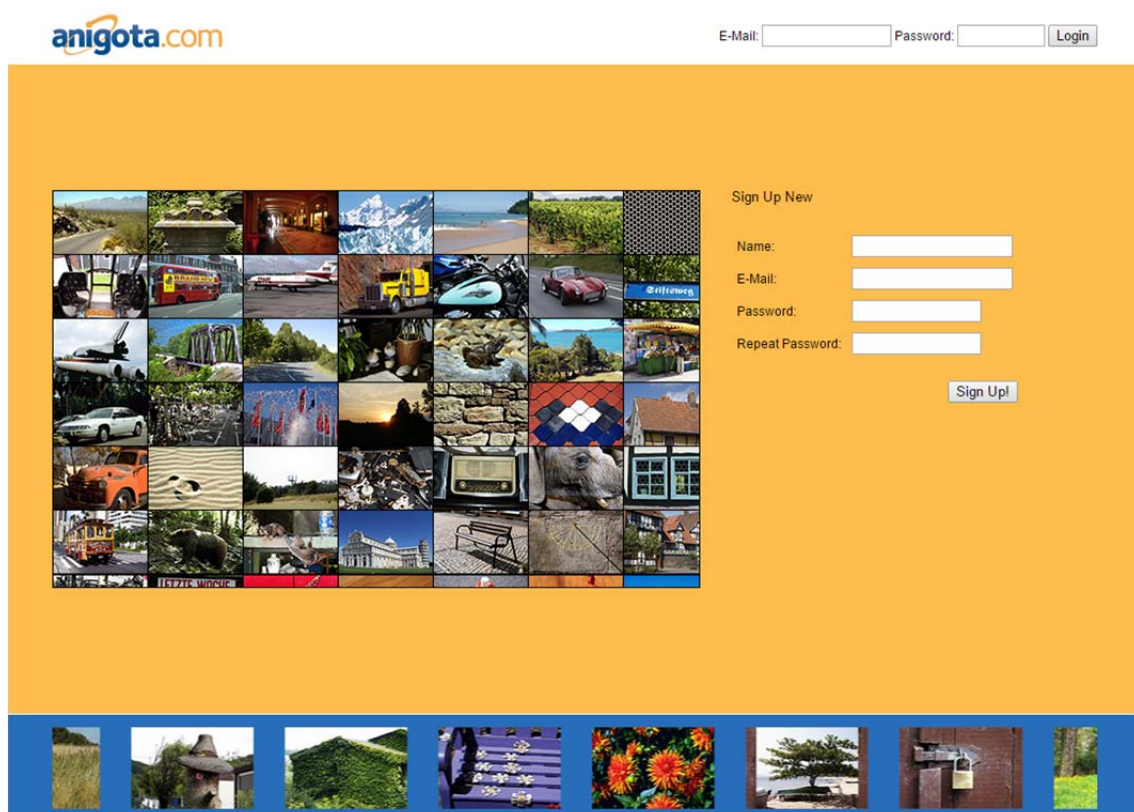


Abbildung 22: Anmeldung und Registrierung am System

Der Prototyp erfordert nur die Angabe eines Namens und einer E-Mail Adresse. Wird ein Feldwert ungültig ausgefüllt, so fordert das System zur Korrektur auf (serverseitige Validierung). Für die Anmeldung steht im oberen Bereich ein Anmeldeformular zur Verfügung.

Nach der erfolgreichen Anmeldung wird die Benutzeroberfläche der Applikation angezeigt. Für neu registrierte Kunden wird vom System ein Standard-Album angelegt und in der Albenliste (A) angezeigt.

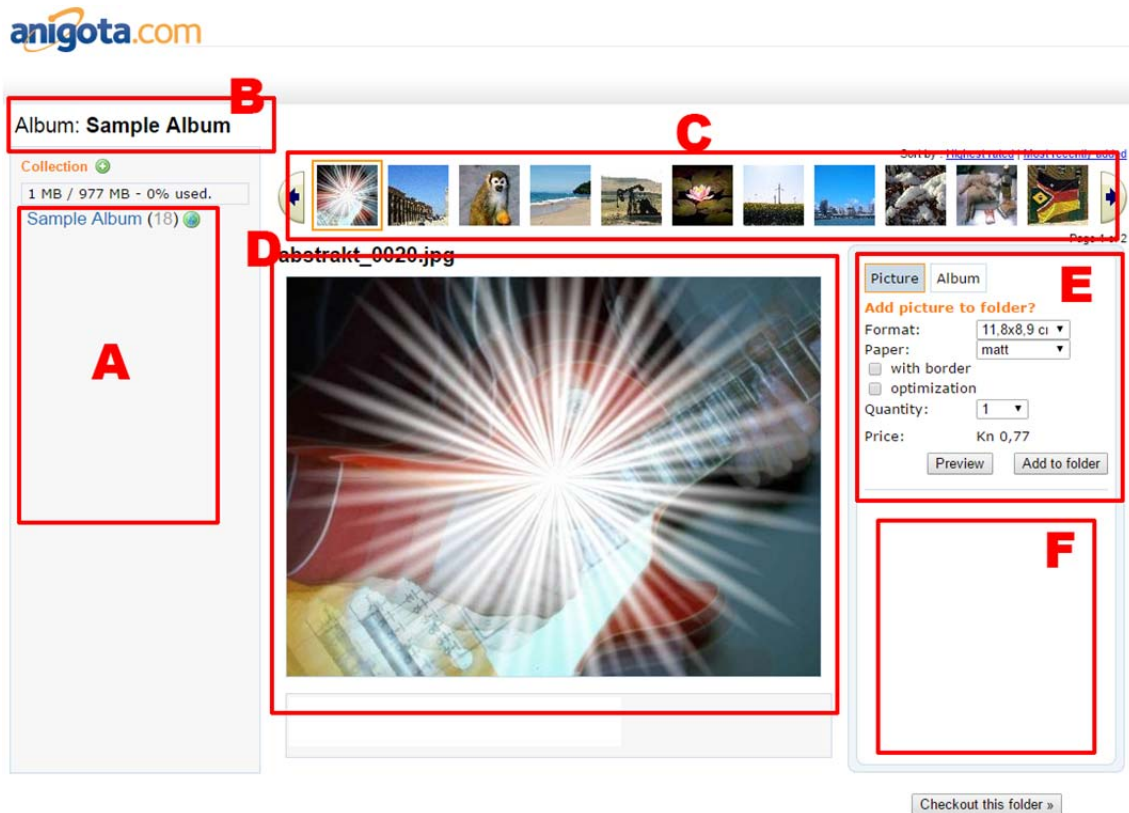
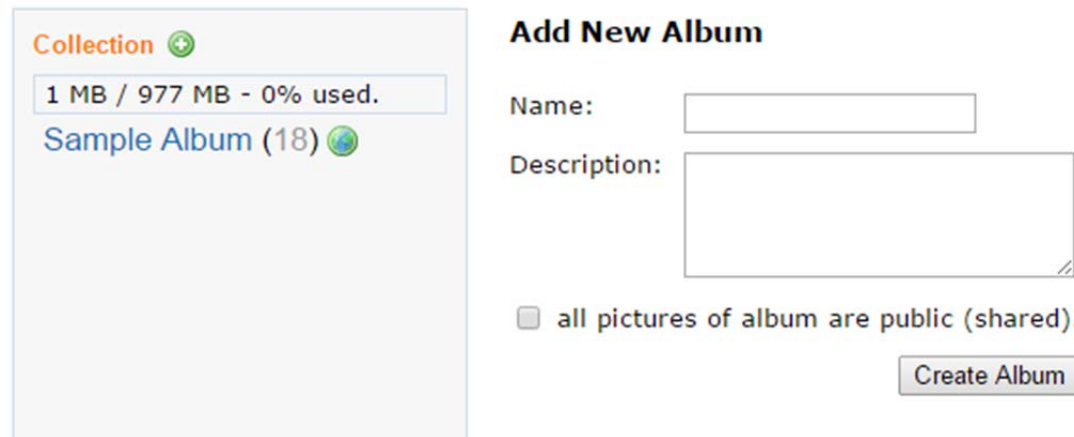


Abbildung 23: Benutzeroberfläche des Web Client

Das Feld A zeigt in einer Liste alle vom Kunden erzeugten Alben an. Neben dem Namen wird noch die Anzahl der im Album enthaltenen Bilder, sowie für den Fall dass das Album veröffentlicht ist, ein Weltkugelsymbol angezeigt. Der Name des aktuell geöffneten Albums wird im Feld B gezeigt. Alle enthaltenen Bilder des Albums werden im Feld C angezeigt. Dies erfolgt mithilfe der im vorigen Abschnitt entwickelten Sucher Komponente. Wird mit der Sucher Komponente ein Bild ausgewählt, so wird dieses im Bild Browser, im Feld D, angezeigt. Die verfügbaren Ausarbeitungsoptionen für das aktuelle Bild werden in Feld E gezeigt. Wurden Fotos der Fotomappe hinzugefügt, so werden diese von der Komponente in Feld F dargestellt.

Um neue Alben der Sammlung hinzuzufügen, muss das Plusymbol in der Album Liste geklickt werden.



**Collection** +

1 MB / 977 MB - 0% used.

Sample Album (18)

### Add New Album

Name:

Description:

all pictures of album are public (shared).

Create Album

Abbildung 24: Benutzeroberfläche für neues Album anlegen

Für jedes neue Album muss ein Name und eine treffende Beschreibung gewählt werden. Optional kann das Album für andere Besucher veröffentlicht werden. Sollen die Attribute eines Albums verändert werden, so kann dies durch klicken der kontextsensitiven Schaltfläche Edit Album erfolgen, die erscheint, sobald die Maus über den Album Namen bewegt wird.

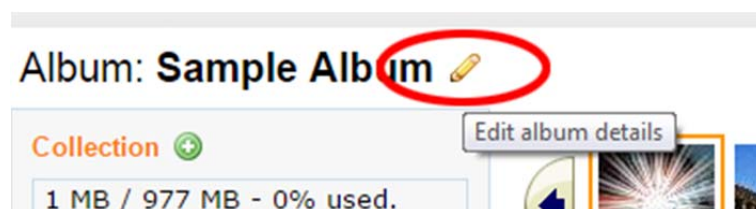


Abbildung 25: Album Attribute editieren

Die Maske zum Editieren der Attribute ist analog zu der des anlegens. Werden die Attribute eines Album von der Desktop-Applikation geändert, so werden diese unverzüglich, auch im Web Client erneuert. Dabei wird mit Hilfe eines Visualisierungseffekts der Vorgang nochmals hervorgehoben.

Der im vorigen Abschnitt vorgestellte Sucher erlaubt das durchsuchen des Albums nach Fotos. Wird ein Foto im Sucher geklickt, wird es im Foto Browser dargestellt und die Metadaten im Feld unter dem Browser angezeigt. Möchte der Kunde das Foto entfernen, oder einfach nur die Metadaten verändern, so stehen dafür kontextsensitive Schaltflächen zur Verfügung. Um diese anzuzeigen muss die Maus über den Bildnamen bewegt werden.

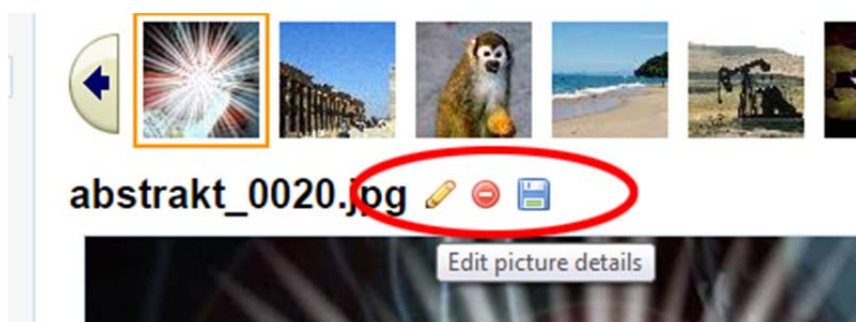


Abbildung 26: Kontext Menü für Bilder

Durch klicken des Diskettensymbols, welches ebenfalls kontextsensitiv eingeblendet wird, können Bilder auch aus dem System lokal heruntergeladen werden.

Durch klicken des Bleistiftsymbols können die Metadaten des Bildes in einer eigenen Maske bearbeitet werden. Dabei wird die Maske unterhalb des Symbols aufgerollt.

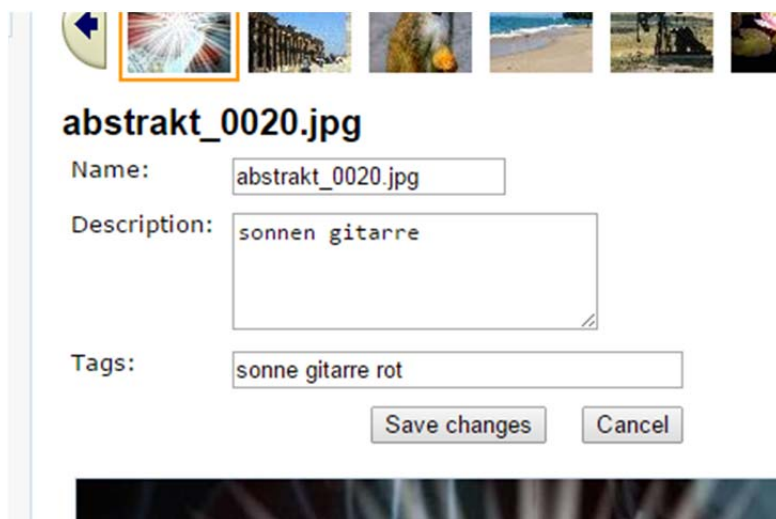


Abbildung 27: Bearbeiten der Metadaten eines Bildes

Die Maske erlaubt neben der Vergabe eines Namens und einer Beschreibung, die Angabe von Tags. Diese Tags werden beim Veröffentlichen des Bildes als Metainformationen für Suchmaschinen angegeben.

Der Bild Browser aus Feld D wurde bereits im vorigen Abschnitt erklärt. Er erlaubt es, die Bilder des Albums, der Reihe nach zu durchsuchen. Dies ist auch mit dem Sucher möglich, jedoch muss im Sucher jedes Bild extra geklickt werden, weiters muss auch manuell umgeblättert werden. Der Browser erlaubt das Durchsuchen ohne umzublättern. Wird ein Foto gefunden das ausgearbeitet werden soll, so stehen dafür in Feld E Ausarbeitungsoptionen zur Verfügung.

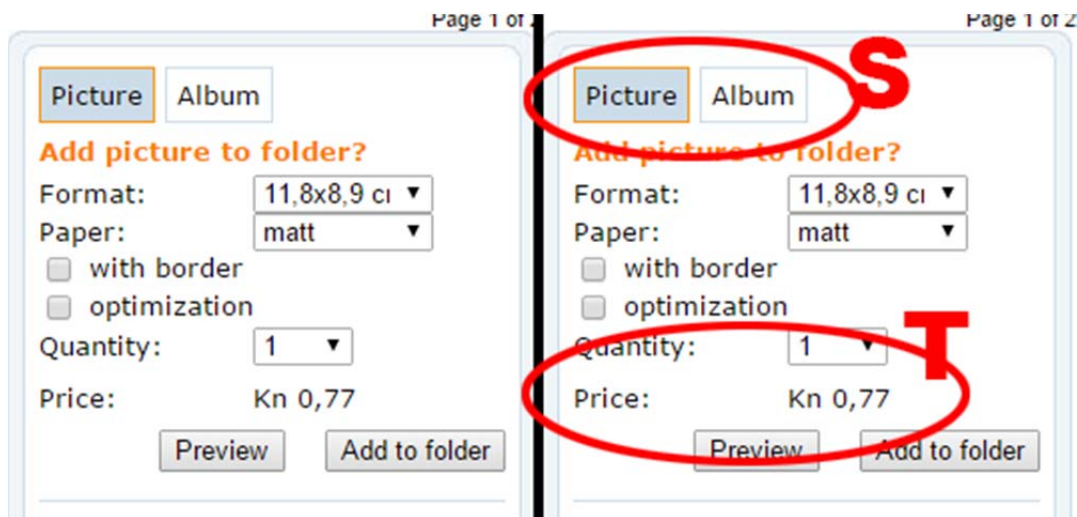


Abbildung 28: Ausarbeitungsoptionen

Links zeigt die Abbildung die verfügbaren Ausarbeitungsoptionen. Rechts sind zwei Regionen noch einmal hervorgehoben. Der Kunde hat die Wahl ob er nur das aktuell gewählte Bild, oder gleich das gesamte Album ausarbeiten möchte. Dafür stehen die Optionen in Feld S zur Verfügung. Im folgenden Schritt kann der Kunde das Format der Ausarbeitung wählen. Dafür stehen prinzipiell folgende Formate zur Verfügung:

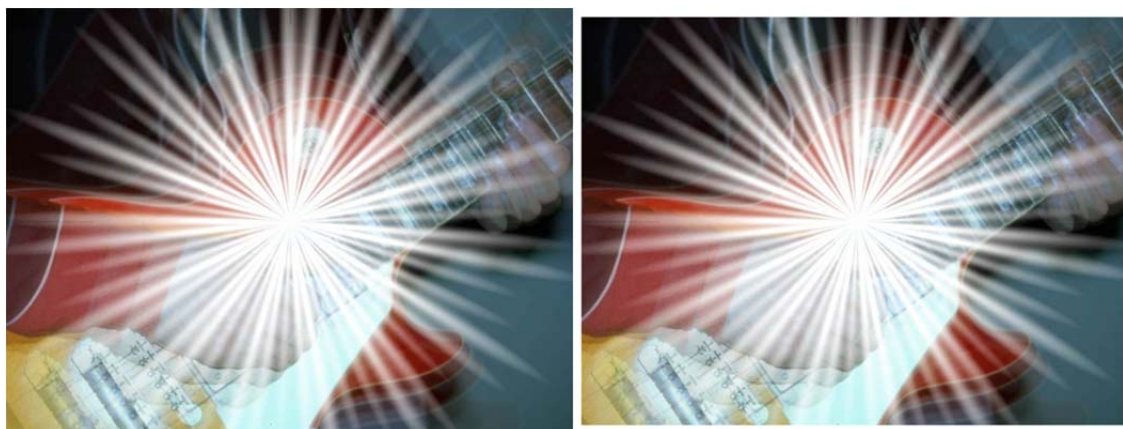
Format 4:3	Format 2:3	Format 3:4
11,8x8,9 cm	8,9x13,3 cm	8,9x11,8 cm
13x10 cm	10,2x15,3 cm	10,2x13,6 cm
17x12 cm	11,4x17,1 cm	11,4x15,2 cm
	12,7x19 cm	12,7x16,9 cm

Abbildung 29: Verfügbare Formate



Welche Formate für das aktuelle Bild letztendlich wirklich zur Verfügung stehen, wird vom Server entschieden. Nachdem ein Bild vom Kunden gefunden wurde, wird eine Anfrage an den Server gestellt, der in einer Antwort die verfügbaren Ausarbeitungsoptionen liefert.

Neben dem Format kann auch eines von zwei Papierformaten gewählt werden. Zur Auswahl stehen Mattß und Glanzpapier. Des Weiteren kann jedem gedruckten Bild ein weißer Rahmen hinzugefügt werden.



**Abbildung 30: Gegenüberstellung Bild mit und Bild ohne Rahmen**

Die Abbildung zeigt links ein Foto ohne Rahmen wobei das Foto gänzlich bis zum Rand des Papiers gedruckt wird. Auf der rechten Seite zu sehen ist ein Foto mit Rahmen. Dabei wird etwas Inhalt des Fotos verdeckt, daher erfolgt keine Verkleinerung des Bildes. Welche Option nun besser ist, kann der Kunde entscheiden, indem er die Vorschaufunktion nutzt.

Soll das Bild vor dem Druck noch durch automatisierte Algorithmen optimiert werden, so muss die Option optimize gewählt werden. Die Optimierung erfolgt jedoch nicht von dem hier vorgestellten System, sondern erfolgt in den nachgelagerten Bearbeitungsschritten. Nachdem der Kunde die Ausarbeitungsoptionen und die gewünschte Stückzahl festgelegt hat, wird sofort der Preis für die Ausarbeitung eingeblendet. Sollten die Kosten zu hoch sein, so kann der Kunde solange die Optionen anpassen, bis ein akzeptabler Preis erreicht wurde.

Durch klicken der Schaltfläche „Add to folder“ wird das Bild mit den gewählten Ausarbeitungsoptionen der Fotomappe hinzugefügt. Die Funktionsweise der Fotomappe wurde bereits im vorigen Abschnitt erklärt.

Sollen einem Album neue Bilder hinzugefügt werden, kann dies in der „Hinzufügen Maske“ erfolgen. Die Maske wird durch klicken der entsprechenden kontextsensitiven Schaltfläche, die erscheint sobald die Maus über ein Album in der Album Liste bewegt wird, geladen.

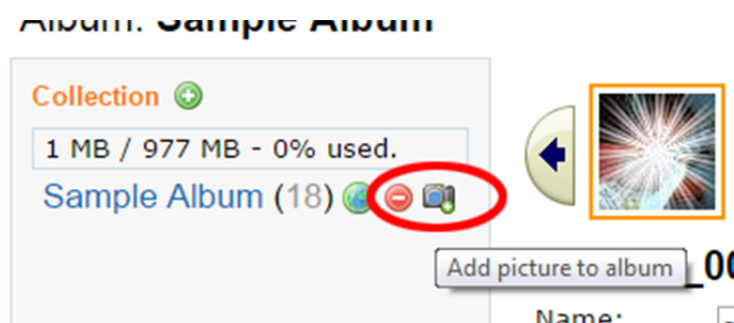


Abbildung 31: Kontext Menü für Album

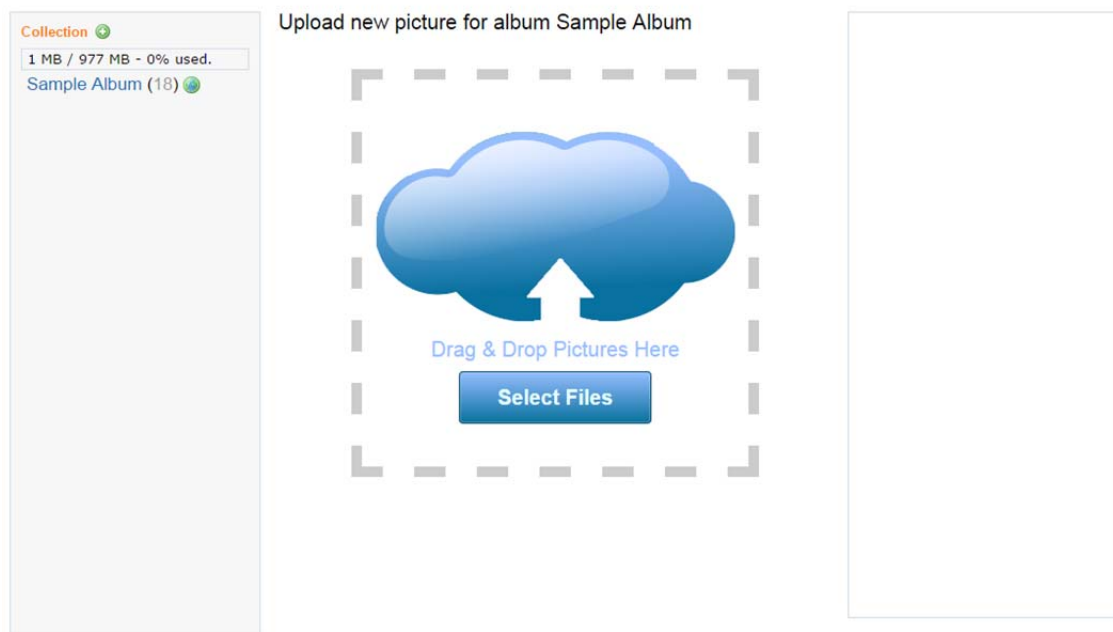
Die Abbildung zeigt das Kontext Menü für Alben.

Die Maske für das Hochladen erlaubt dies auf unterschiedliche Weise. Der Kunde kann Fotos durch klicken auf die Schaltfläche „Select Files“ zum Hochladen auswählen. Dieser Mechanismus verwendet die in vorigen Abschnitten vorgestellte HTML5 Technologie zum Auswählen mehrerer Dateien gleichzeitig. Alternativ können Dateien auch direkt über die durch Strichlierung markierte Fläche gezogen und fallen gelassen werden. Dieser Mechanismus basiert auf der HTML5 Drag & Drop Technologie, die bereits ebenfalls in vorigen Abschnitten vorgestellt wurde.

Nachdem ein Foto gewählt wurde, wird dies ohne Hochgeladen zu werden, direkt im Browser, in einer Vorschau angezeigt. Dieser Mechanismus basiert auf der HTML5 Canvas sowie der File Reader API Technologie. Dem Kunden wird damit ermöglicht das Foto nochmals anzusehen, um nicht aus Versehen, ein falsches Foto hochzuladen. Obwohl jedes neue Foto, als nicht öffentlich markiert, gespeichert wird, und auch wieder vom Kunden gelöscht werden kann, so lässt sich dadurch die Wartezeit für das Hochladen ersparen. Das gewählte Bild wird auch automatisch auf eine entsprechende

Größe skaliert, sodass es in das Anzeigefenster passt. Dabei wird auch auf das Seitenverhältnis Rücksicht genommen, sodass keine Verzerrung entsteht.

**anigota.com**



**Abbildung 32: Maske zum Hochladen von Bildern**

Die folgende Abbildung zeigt die Bild Vorschau die einen unerwünschten Upload verhindern soll. Die Maske offeriert zwei Schaltflächen die es entweder erlauben ein anderes Bild zu wählen und den aktuellen Vorgang abubrechen, oder mit dem Hochladen fortzufahren und das gewählte Bild zum Server hochzuladen. Sollte der Kunde mehrere Fotos ausgewählt haben, so entfällt die Vorschau und es wird direkt mit dem Hochladen begonnen.

Add this picture to album Sample Album



No, select other

Yes, upload this picture

Abbildung 33: Bildvorschau für das Hochladen

Neben der erwähnten Zeitersparnis, die dem Kunden zugutekommt durch Verhinderung eines ungeplanten Hochladens, werden nebenbei auch wertvolle Server Ressourcen gespart.

Der Upload Mechanismus erfolgt durch die HTML5 Ajax 2.0 Technologie. Der Fortschritt des Hochladens wird in einer Statusbox angezeigt.

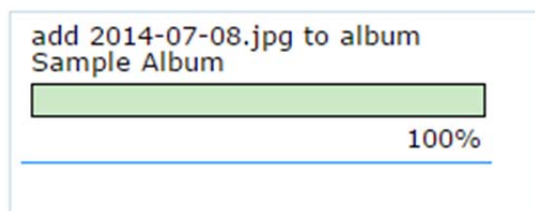


Abbildung 34: Fortschritt Statusbox

Eine solche Box wird für jedes gewählte Foto eingeblendet. War das Hochladen erfolgreich, so färbt sich der Balken grün. Bei Fehlern färbt sich der Balken rot und es wird eine Fehlermeldung eingeblendet.

Wurden alle Fotos, die der Kunde ausarbeiten möchte der Foto Mappe hinzugefügt, so kann dieser seine Bestellung durch klicken auf die Schaltfläche Checkout this folder abschließen. Er wird dann auf die E – Commerce Plattform weitergeleitet die das Abschließen der Bestellung erlaubt.

## 5.6 Der Desktop Client

Der Desktop Client erlaubt dem Kunden Zugang zum Cloud Speicherplatz. Um Zugang zu diesen zu bekommen, muss der Kunde bereits einen Zugang über die Web Oberfläche registriert haben. Nach dem Start des Client erfolgt eine Aufforderung zum Anmelden.

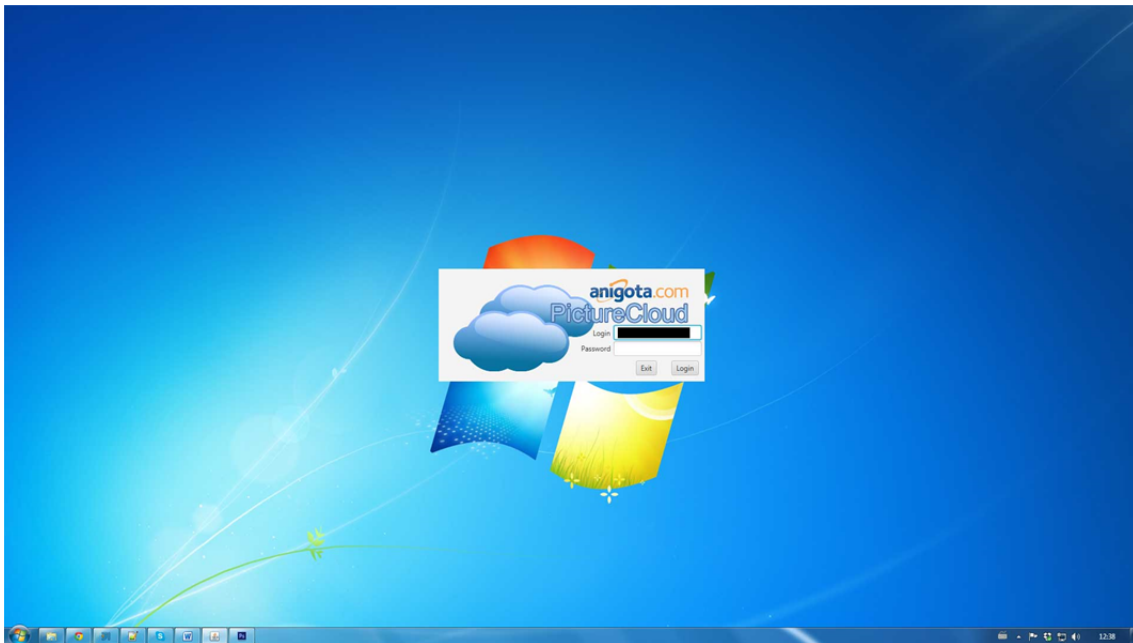
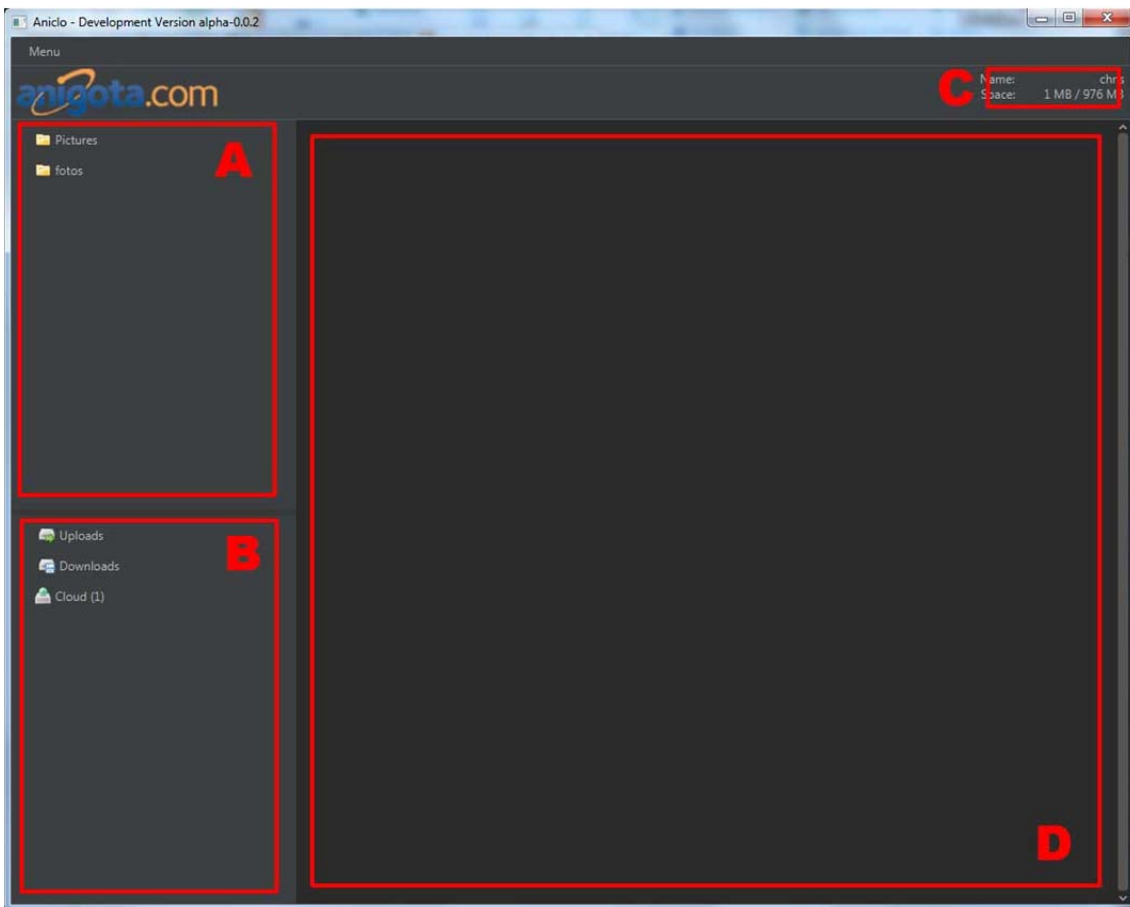


Abbildung 35: Anmeldung Cloud Client

Die Implementierung des Desktop Clients wurde mit JavaFX 8 umgesetzt. Voraussetzung für den Start des Clients ist daher eine korrekt installierte Java 8 Laufzeitumgebung.

Beim ersten Anmelden muss der Kunde ein Passwort angeben. Der Client bezieht daraufhin vom Server einen API Schlüssel, der zusammen mit dem Login, im geschützten Benutzerverzeichnis abgelegt wird. Dies hat den Vorteil, dass jeder weitere Anmeldevorgang ohne Passwordeingabe erfolgen kann sofern sich der Benutzername nicht ändert. Beim Auftreten eines Fehlers während des Anmeldens wird der Anwender darüber in der Maske informiert. Nach erfolgreichem Login öffnet sich die Benutzermaske des Cloud Client.



**Abbildung 36: Cloud Client Hauptmaske**

Die Maske verfügt über vier wichtige Zonen. Diese sind bei der Interaktion mit der Cloud von Bedeutung. Die in der Abbildung markierte Zone A stellt alle Orte in einer Baumstruktur dar, die vom Benutzer hinzugefügt wurden. Diese Orte sind Pfade zu Verzeichnissen auf der lokalen Festplatte die Bilder enthalten. Neue Pfade können per Drag & Drop hinzugefügt werden. Möchte der Anwender Verzeichnisse wieder entfernen, so erfolgt dies durch einen Rechtsklick auf das Verzeichnis und der entsprechenden Option im erscheinenden Kontextmenü.

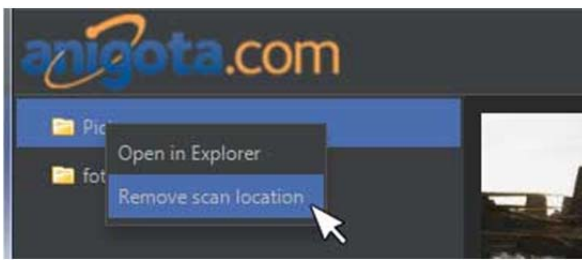


Abbildung 37: Bearbeiten der lokalen Bildverzeichnisse

Jedes im Baum befindliche Verzeichnis kann auch im Datei Browser angezeigt werden ohne erst mühsam dorthin navigieren zu müssen. Der Menüpunkt Open in Explorer erlaubt dies. Beim ersten Start der Applikation wird automatisch das Bildverzeichnis hinzugefügt das im Betriebssystem konfiguriert ist. Alle Änderungen an den konfigurierten Bildverzeichnissen werden lokal im Benutzerverzeichnis in einer SQL Lite Datenbank gespeichert und werden auch wieder nach einem Neustart der Applikation geladen. Wird auf ein Verzeichnis im Baum geklickt wird der Inhalt in Zone D dargestellt.

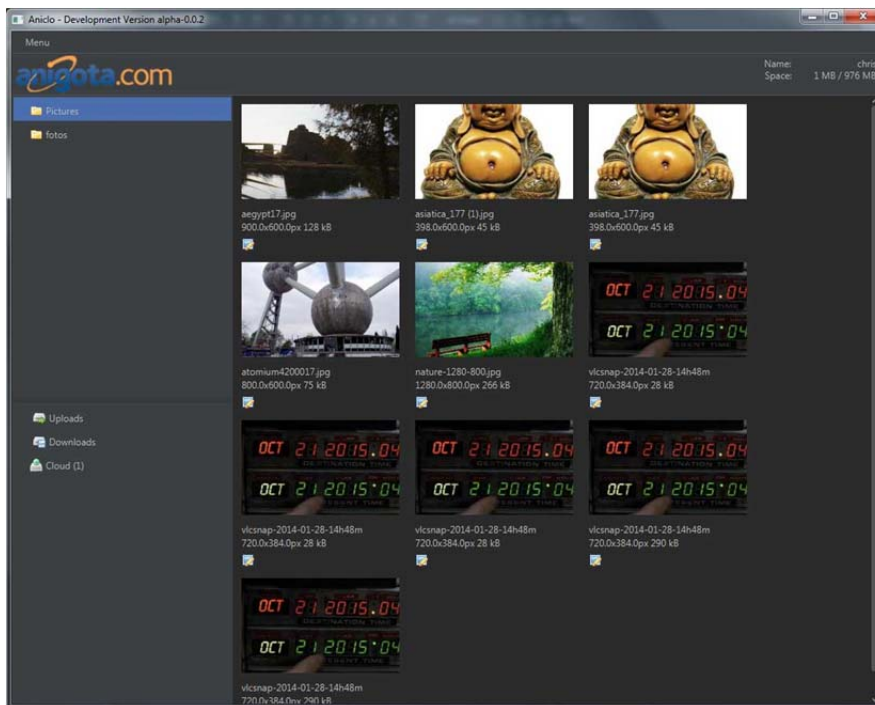
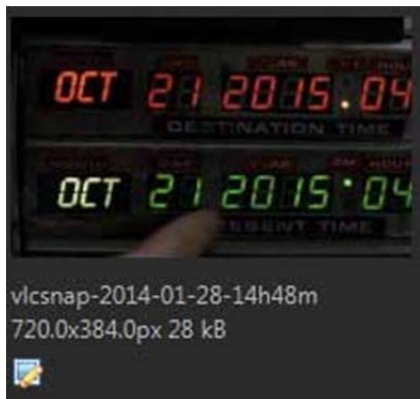


Abbildung 38: Bildverzeichnisse durchsuchen

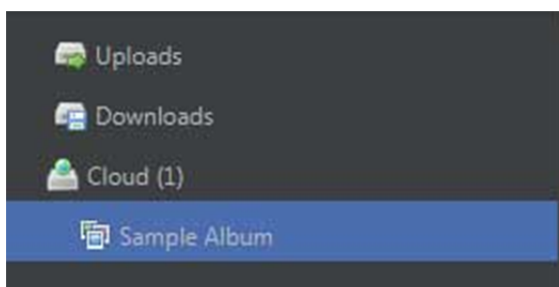


Für jedes Bild wird eine verkleinerte Vorschau angezeigt. Die Vorschau enthält auch einige Metainformationen.

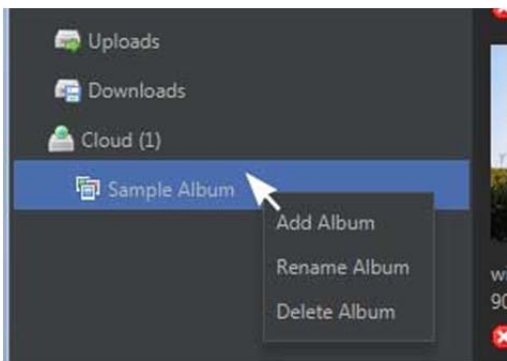


Neben dem Dateinamen wird auch noch die Dimension des Fotos sowie die Größe der Datei angezeigt. Möchte der Anwender das Bild editieren, so öffnet ein Klick auf das Edit Symbol den konfigurierten Bildeditor.

In Zone B der Maske werden Statusinformationen in einer Baumstruktur angezeigt.

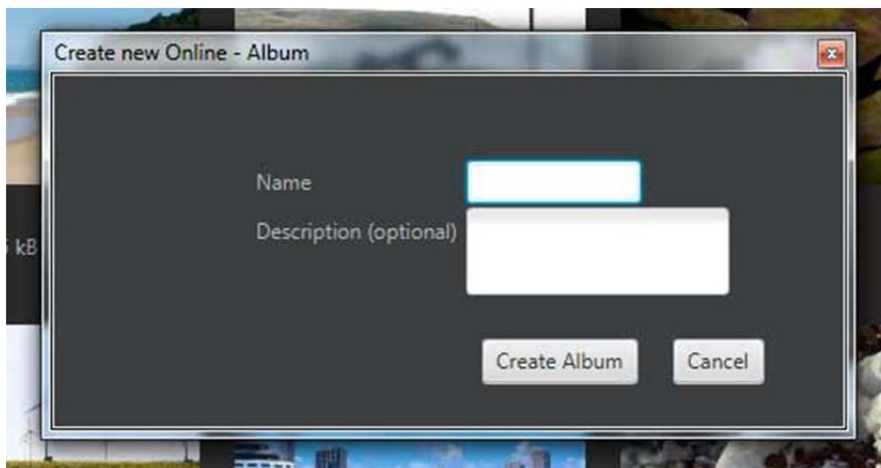


Beim Klick auf den Knoten Uploads, werden alle Bilder gezeigt die vom Benutzer gewählt wurden um in die Cloud hochgeladen zu werden. Analog zeigt der Knoten Downloads aller Bilder die aus der Cloud lokal auf der Festplatte gespeichert werden sollen, dies aber noch nicht sind. Der Cloud Knoten enthält als Kinder alle Alben die der Anwender bereits in der Cloud angelegt hat. Dies entspricht denselben Alben die auch im Web Client eingesehen werden können. Durch einen Rechtsklick auf ein Album erscheint ein Kontextmenü das Interaktionen anbietet um neue Alben zu erzeugen und bereits existierende wieder zu löschen.



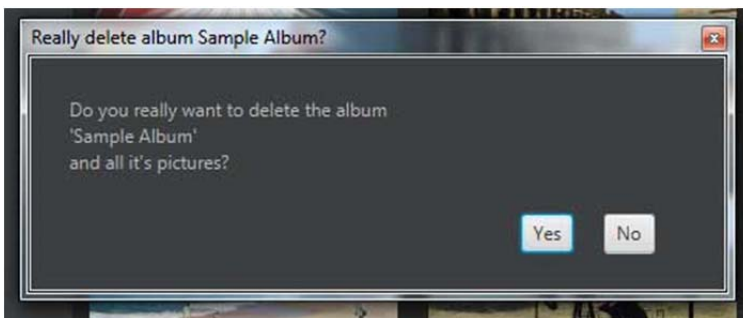
**Abbildung 39: Album Kontext Menü**

Existierende Alben können auch umbenannt werden. Jede hier ausgeführte Aktion wird auch unverzüglich im Web Client, sollte dieser parallel geöffnet sein, durch eine Animation visualisiert. Wird auf den Eintrag Add Album geklickt, so öffnet sich ein Formular um Daten für das neue Album eingeben zu können.



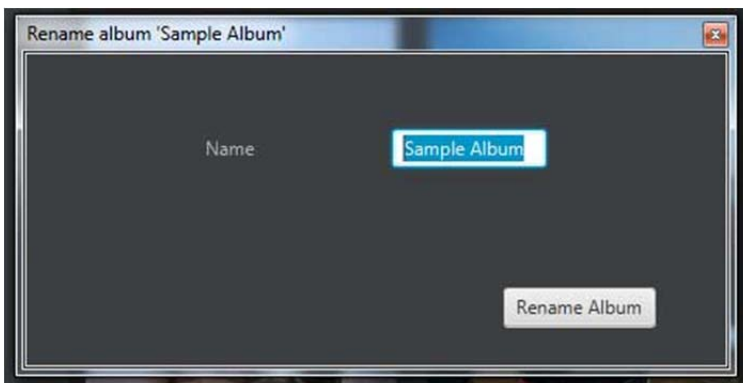
**Abbildung 40: Neues Album anlegen**

Für das Anlegen eines neuen Albums wird ein Name und eine Beschreibung verlangt. Der Kunde kann allerdings keine Fotos im Internet veröffentlichen. Dies ist nur über den Web Client möglich, der dazu auch das Passwort verlangt. Jedes Bild, das vom Cloud Client in die Cloud geladen wird, ist standardmäßig ein privates Bild. Ein entfernen eines Albums aus der Cloud muss vom Benutzer explizit bestätigt werden. Die Aufforderung erfolgt in einem Dialog.



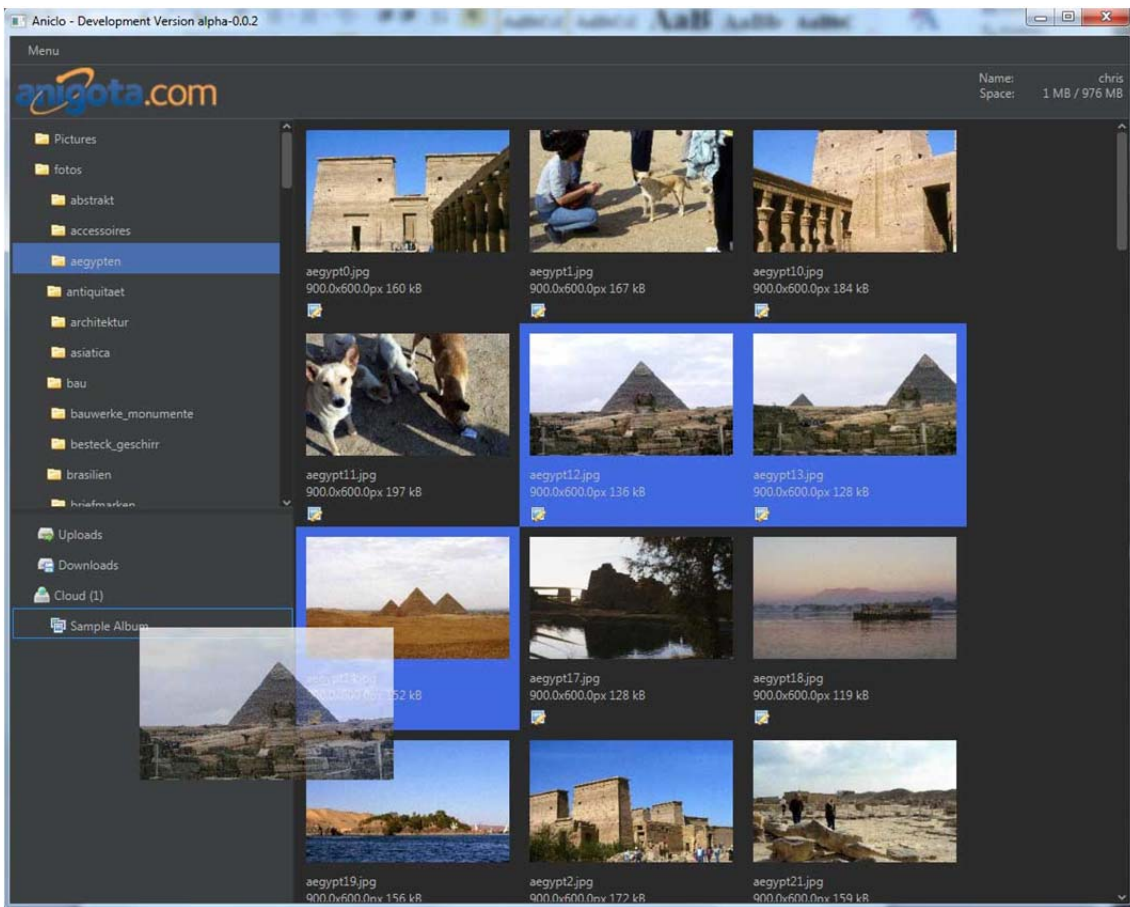
**Abbildung 41: Album entfernen**

Das Entfernen eines Bildes muss ebenfalls explizit bestätigt werden. Wird ein gesamtes Album gelöscht, so werden auch alle darin enthaltenen Bilder aus der Cloud entfernt. Alben können auch direkt umbenannt werden. Durch Auswahl des entsprechenden Menüpunkts im Kontextmenu erscheint dafür ein eigenes Formular in einem Dialogfenster.



**Abbildung 42: Album umbenennen**

Um Fotos in die Cloud hochzuladen, müssen diese zunächst in der Browserfläche markiert werden. Durch halten der Steuerungstaste und gleichzeitigem klicken auf die Fotos, ist auch eine Mehrfachauswahl möglich. Die markierten Fotos können anschließend über dem Album, in welches sie hinzugefügt werden sollen, fallen gelassen werden.



Das Zielalbum wird beim Drag & Drop Vorgang blau hervorgehoben, um kenntlich zu machen, welches Album gerade ausgewählt wurde. Nach dem fallenlassen der Bilder wird der Fortschritt unter Uploads angezeigt.

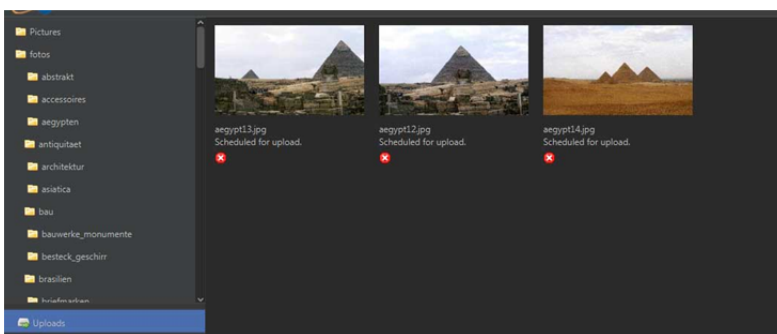
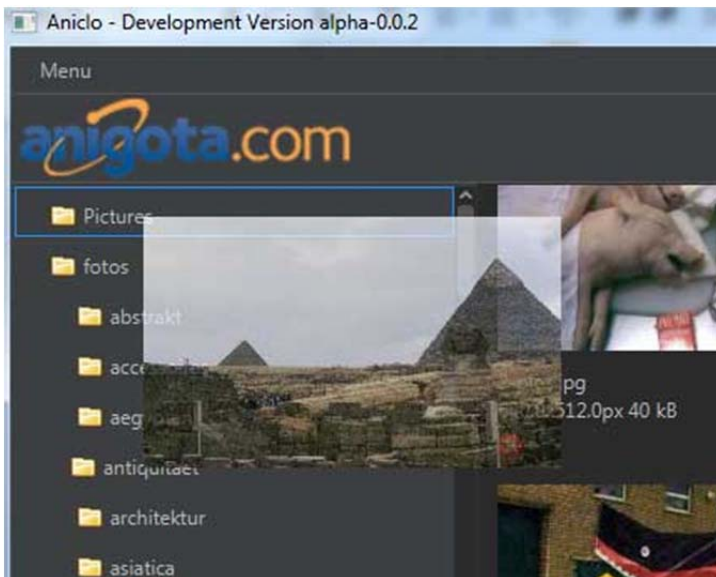


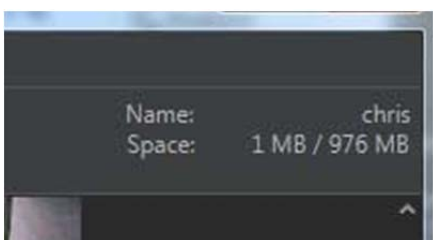
Abbildung 43: Upload Fortschritt

Sollen Bilder aus der Cloud lokal gespeichert werden so kann dies auch per Drag & Drop erfolgen. Dazu muss zunächst ein Album durchsucht werden und das gewünschte Foto über einen lokalen Order des Baums der Zone A gezogen werden.



Der Zielordner wird wiederum blau hervorgehoben.

Jeder Kunde verfügt über ein bestimmtes Maß an Speicherplatz. Der verbleibende Speicherplatz wird in der Maske rechts oben angezeigt.



**Abbildung 44: Speicherplatz Anzeige**

Neben dem Speicherplatz wird auch der Name, des aktuell in die Cloud eingeloggten, Benutzers angezeigt.

Ein Vorteil der Desktop Cloud Applikation ist, das große Bilder im Hintergrund in die Cloud geladen werden können. Der Client kann ganz einfach minimiert werden und

führt im Hintergrund den Vorgang des Hochladens durch. Auch das Speichern der Bilder kann im Hintergrund erfolgen. Geht dabei die Verbindung zum Internet verloren, so pausiert der Client und wartet bis diese wiederhergestellt ist und fährt dann mit dem Prozess automatisch wieder fort. Damit der Client im Hintergrund nicht alle Ressourcen verbraucht, kann die verfügbare Bandbreite in den Einstellungen konfiguriert werden.

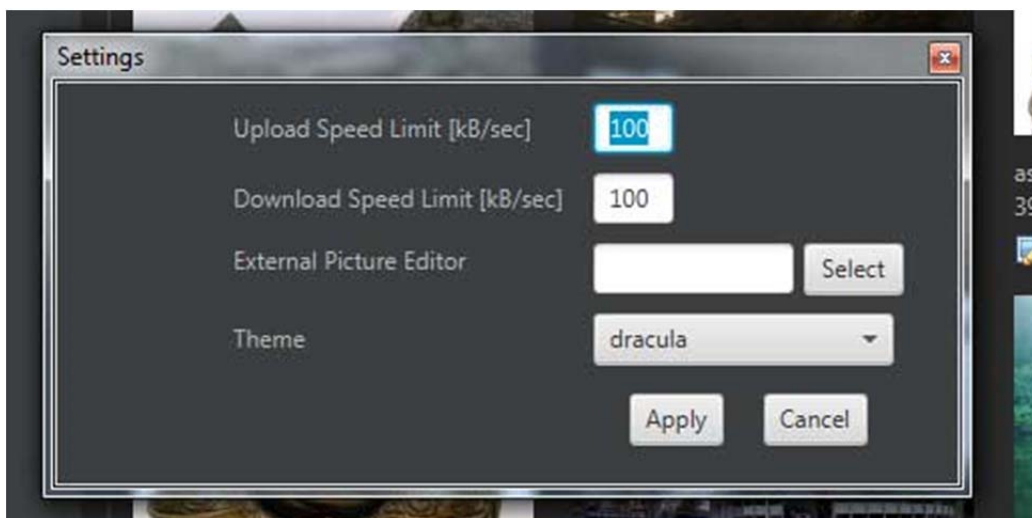
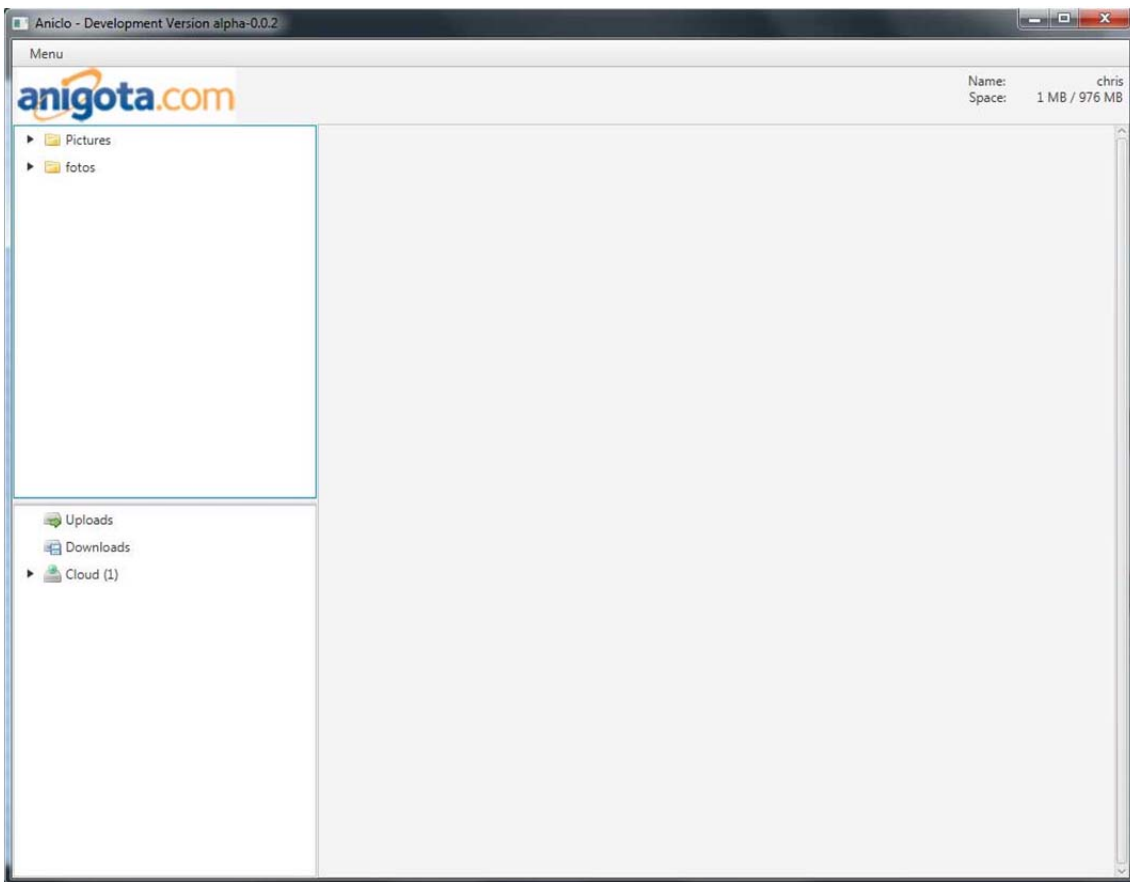


Abbildung 45: Einstellungen des Cloud Client

Neben den Einstellungen der verfügbaren Bandbreiten, kann ein externer Editor festgelegt werden. Klickt der Benutzer auf die Schaltfläche um Bilder zu editieren, so wird entweder der konfigurierte Editor geöffnet, oder es wird der vom System standardmäßige Editor verwendet.

Die Darstellung der Benutzeroberfläche kann mittels Themen angepasst werden. Die Beschreibung der Oberfläche in diesem Abschnitt benutzte das Dracula Thema. Folgende Abbildung zeigt den Client mit dem Standard Thema.



**Abbildung 46: Cloud Client mit alternativem Thema**

Auf die Erzeugung weiterer oder von Anwendern selbst erstellter Themen ist möglich. Die Definition von Themen erfolgt über zwei Konfigurationsdateien. Eine JavaFX CSS Datei und eine Java Properties Datei welche die verwendeten Ressource wie Icons und Hintergrundbilder definiert.

## 5.7 Die Implementierung der Server API

Die Server Seite der Plattform wurde auf Basis des Spring Frameworks implementiert. Beim Start des Servers wird zunächst ein Environment Objekt erzeugt, das die Konfiguration der Server Instanz enthält. Danach erfolgt der Start des Tomcat 8 Servers der nachdem er gestartet wurde, einen Spring Context erzeugt. Hierbei wurde der in der Servlet 3 Spezifikation neue ServletContainerInitializer Mechanismus bzw. die vom Spring Framework konkretere Version eines WebApplicationInitializer verwendet. Beim Startvorgang wird vom Servlet Container nach vorhanden Initialisieren gesucht und diese automatisch ausgeführt. Eine XML basierte Konfiguration der Applikation wird somit unnötig.

Der Startvorgang des Servers wird auf die Konsole geloggt.

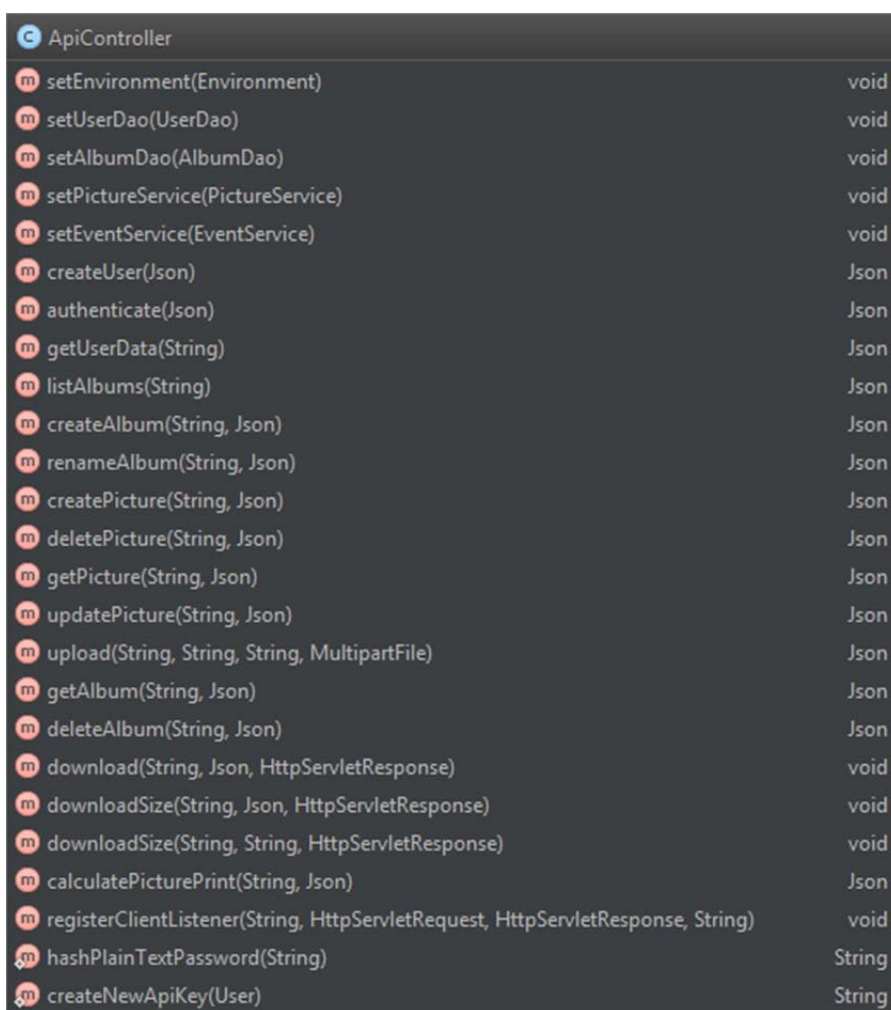
```
Dez 17, 2014 4:18:57 PM org.apache.coyote.AbstractProtocol init
INFO: Initializing ProtocolHandler ["http-nio-8080"]
Dez 17, 2014 4:18:57 PM org.apache.tomcat.util.net.NioSelectorPool getSharedSelector
INFO: Using a shared selector for servlet write/read
Dez 17, 2014 4:18:57 PM org.apache.catalina.core.StandardService startInternal
INFO: Starting service Tomcat
Dez 17, 2014 4:18:57 PM org.apache.catalina.core.StandardEngine startInternal
INFO: Starting Servlet Engine: Apache Tomcat/8.0.14
Dez 17, 2014 4:19:06 PM org.apache.catalina.core.ApplicationContext log
INFO: Spring WebApplicationInitializers detected on classpath:
[ais.web.ServletInitializer@110a5b49]
Dez 17, 2014 4:19:06 PM org.apache.catalina.core.ApplicationContext log
INFO: Initializing Spring root WebApplicationContext
Dez 17, 2014 4:19:06 PM org.apache.coyote.AbstractProtocol start
INFO: Starting ProtocolHandler ["http-nio-8080"]
[Server@56e07a08]: [Thread[main,5,main]]: checkRunning(false) entered
[Server@56e07a08]: [Thread[main,5,main]]: checkRunning(false) exited
[Server@56e07a08]: Initiating startup sequence...
[Server@56e07a08]: Server socket opened successfully in 1 ms.
[Server@56e07a08]: Database [index=0, id=0, db=file:data/aimcodb, alias=aimcodb] opened
sucessfully in 402 ms.
[Server@56e07a08]: Startup sequence completed in 404 ms.
[Server@56e07a08]: 2014-12-17 16:19:06.768 HSQLDB server 2.3.2 is online on port 8081
[Server@56e07a08]: To close normally, connect and execute SHUTDOWN SQL
[Server@56e07a08]: From command line, use [Ctrl]+[C] to abort abruptly
```

Im Spring Kontext ist auch der zu verwendende HSQL Datenbank Server konfiguriert. Dieser wird im Anschluss an den Servlet Container gestartet und wird für die Erzeugung und Befüllung des Datenbankverbindungs-Pools sowie der Hibernate Session Factory benötigt. Die Instanzen der Persistenz-Schicht werden im Anschluss erzeugt und von den Instanzen der Business-Schicht verlangt. Zuletzt werden noch die konfigurierten Controller der Präsentations-Schicht erzeugt.



Die gesamte Applikation wird daher, beginnend mit der untersten Schicht, fortlaufen und nach oben hin zu abstrakteren Schichten gestartet.

In der Präsentations-Schicht befinden sich zwei Controller. Ein Controller dient dem Starten des Web-Clients und der Authentifizierung von Besuchern durch den Web Client. Der zweite Controller implementiert die API die jeweils von Web Client und Cloud Client verwendet wird. Die folgende Abbildung zeigt die Schnittstelle des API Controller.



Method	Return Type
setEnvironment(Environment)	void
setUserDao(UserDao)	void
setAlbumDao(AlbumDao)	void
setPictureService(PictureService)	void
setEventService(EventService)	void
createUser(Json)	Json
authenticate(Json)	Json
getUserData(String)	Json
listAlbums(String)	Json
createAlbum(String, Json)	Json
renameAlbum(String, Json)	Json
createPicture(String, Json)	Json
deletePicture(String, Json)	Json
getPicture(String, Json)	Json
updatePicture(String, Json)	Json
upload(String, String, String, MultipartFile)	Json
getAlbum(String, Json)	Json
deleteAlbum(String, Json)	Json
download(String, Json, HttpServletResponse)	void
downloadSize(String, Json, HttpServletResponse)	void
downloadSize(String, String, HttpServletResponse)	void
calculatePicturePrint(String, Json)	Json
registerClientListener(String, HttpServletRequest, HttpServletResponse, String)	void
hashPlainTextPassword(String)	String
createNewApiKey(User)	String

Abbildung 47: Server API Schnittstelle

Die Implementierungen der wichtigsten Methoden wird im Folgenden vorgestellt.

Über die Methode `authenticate` wird für einen Benutzerzugang ein API Schlüssel bezogen. Jede Kommunikation mit dem API Controller erfordert einen API Schlüssel. Anfragen die keinen Schlüssel enthalten werden abgelehnt. Die Methode erwartet einen Benutzernamen und ein Passwort in einem JSON Objekt.

Die Methode `getUserData` wird für die Abfrage von Benutzerdaten verwendet. Zum Beispiel den Namen eines Benutzers oder der verfügbare sowie verbrauchte Cloud Speicherplatz kann über diese Methode abgefragt werden. Die Methode erwartet ein leeres JSON Objekt.

Die Methode `createAlbum` wird verwendet um ein Album zu erzeugen. Es erwartet in der Anfrage ein JSON Objekt welches den Namen und die Beschreibung des neuen Albums beinhaltet. Als Antwort wird dem Client ein JSON Objekt gesendet das alle Daten des neuen Album, wie ID, Name etc., beinhaltet.

Wurde ein Album erzeugt, so können die Methoden `renameAlbum` und `deleteAlbum` verwendet werden, um das Album umzubenennen oder zu löschen. Die Methoden erwarten jeweils ein JSON Objekt das die ID des betreffenden Albums beinhaltet, sowie im Fall von `renameAlbum` den neuen Name des Albums. Die Methoden liefern als Antwort jeweils ein JSON Objekt, das die ID sowie eine Statusmeldung, der Operation beinhaltet.

Mit der Methode `createPicture` werden leere Bilder erzeugt. Ein leeres Bild ist eines, das lediglich aus einer ID besteht, und noch keine Daten beinhaltet. Die Methode liefert in einem JSON Objekt eine ID, die ein Client verwenden kann, um binäre Daten für das Bild zu übertragen.

Um die binären Daten für ein Bild zu speichern wird die Methode `upload` verwendet. Sie erwartet eine Anfrage in einer speziellen Form. Neben den Binärdaten muss im Anfragekopf auch eine Angabe über den Umfang der Daten mittels, des Content-Range Kopfwertes, gemacht werden. Des Weiteren muss die Anfrage die ID des Bildes beinhalten für die die Daten im Anfragekörper gedacht sind.

Mit der Methode `deletePicture` können Bilder durch Angabe der ID im Anfrage JSON Objekt wieder gelöscht werden. Als Antwort liefert die Operation wiederum ein JSON Objekt, welches eine Statusmeldung der Operation beinhaltet.

Informationen über ein Bild, wie zum Beispiel Metadaten, können mit der Methode `getPicture` abgefragt werden, die die Angabe einer ID erwartet. Sollen Daten für ein Bild verändert werden, so steht dafür die Methode `updatePicture` zur Verfügung. Beide Methoden erwarten in der Anfrage die ID des Bildes sowie im Fall von `updatePicture` auch jene Daten, die geändert werden sollen. Neue Binärdaten können einem Bild nicht hinzugefügt werden.

Die Methoden `download` und `downloadSize` werden verwendet um die Binärdaten eines Bildes vom Server zu laden. Erwartet wird die ID des Bildes sowie die benötigte Größe.

Für alle Methoden der Server API wurden JUnit Testfälle erzeugt.

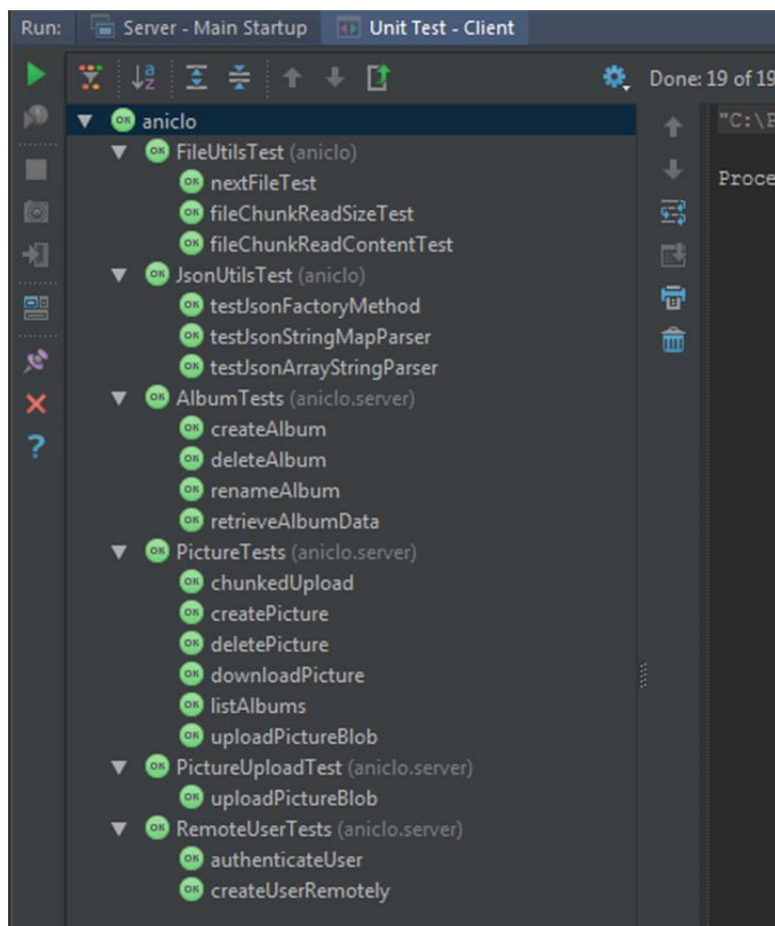


Abbildung 48: JUnit Server Test Suite

## 5.8 Quellcodestatistik

Der Quellcode der Server und Web Client Implementierung setzt sich wie folgt zusammen:

<b>Dateiformat</b>	<b>Anzahl Dateien</b>	<b>Anzahl Zeilen</b>
.java	62	12.979
.jsp	3	835
.html	2	50
.css	3	640
.js	9	5.703
.xml	11	680
.properties	3	141
.gif	2	
.png	19	
.jpg	36	

Abbildung 49: Quellcodestatistik von Server und Web Client

Der Quellcode der Cloud Client Implementierung setzt sich wie folgt zusammen:

<b>Dateiformat</b>	<b>Anzahl Dateien</b>	<b>Anzahl Zeilen</b>
.java	84	19.348
.fxml	8	401
.css	2	175
.properties	4	266
.png	10	
.jpg	2	

Abbildung 50: : Quellcodestatistik des Cloud Client

Die Angaben beinhalten ausschließlich selbst erstellten Inhalt. Frameworks und Bibliotheken wurden nicht berücksichtigt.

## 5.9 Kritik und Erweiterungsmöglichkeiten

Während der Entwicklung des Prototyps sind einige Verbesserungsmöglichkeiten sowie kritische Punkte aufgefallen, die bei der Entwicklung einer kommerziellen Version des Systems beachtet werden sollten.

Der Server Prozess liest aktuell beim Start die Konfiguration für das Environment aus einer statischen Konfigurationsdatei die lokal im Verzeichnissystem liegt. Um ein schnelleres Deployment auf mehreren Maschinen gleichzeitig zu ermöglichen, sollte dieser die Konfiguration beim Start von einer zentralen Registrierungsstelle abrufen.

Der Server verwendet aktuell eine eingebettete Datenbank. Diese wurde im Spring Kontext als Spring Komponenten konfiguriert und wird beim Start instanziiert. In einer produktiven Umgebung sollten diese Komponenten nicht Teil der Applikation sein. Die Änderung ist jedoch nicht sehr aufwändig, da nur die Konfiguration des Datenbank-Verbindungspool geändert werden muss. Die Einstellungen können über das zuvor erwähnte Environment gemacht werden. Daher ladet der Server die Konfiguration von der Registrierungsstelle herunter, so kann diese Stelle auch die zu verwendende Datenbank spezifizieren.

Ein weiter Kritikpunkt ist die verwendete Datenbank. Wie sich gezeigt hat, ist der Hypersonic SQL Server nicht sehr geschickt mit dem Umgang von Blob (Binary Large Object) Objekten. Es können zum Beispiel keine Daten zu einem existierenden Blob hinzugefügt werden, ohne dass dieser neu geladen wird. Als Ersatz würde sich der Postgre SQL Server anbieten.

Wie gezeigt, wurde erlaubt, dass die API des Servers eine Kommunikation mit jedem Client der das HTTP Protokolle unterstützt. Somit wäre es möglich einen Client für mobile Geräte zu entwickeln. Die Kommunikation erfolgt auf Basis von serialisierten JSON Objekten, wofür in fast jeder Programmiersprache zumindest eine Bibliothek existiert.

Eine weitere Möglichkeit wäre, den Desktop Client dahingehend zu erweitern, dass dieser automatisch beim Systemstart gestartet wird. Sind Fotos für das Hochladen vorgemerkt, so kann direkt mit dem Hochladen begonnen werden. Dazu sollte eine Zusätzliche Option in den Einstellungen des Clients angeboten werden.

## 6. Zusammenfassung

Diese Arbeit begann mit einer geschichtlichen Einführung der Fotografie. Dabei wurden beginnend mit Aristoteles im 4. Jahrhundert vor Christus, bis hin zur Gegenwart, die wichtigsten Entwicklungen genannt. Es wurden die gängigsten digitalen Bildformate im Internet, sowie Qualitätsmerkmale von Bildern, diskutiert. Den Abschluss der Einführung bildete ein Ausblick und aktuelle Trends der Digitalfotografie.

Im Anschluss an die Einführung wurden sechs etablierte Plattformen im Internet zur Bildausarbeitung analysiert. Die Auswertung der Analyse wurde diskutiert und in einer Matrix gegenübergestellt. Dabei wurden abschließend die Alleinstellungsmerkmale einer jeden Plattform betrachtet.

Basierend auf der Analyse und der aufgezeigten Schwächen und Stärken vorhandener Ausarbeitungsplattformen, wurden Technologien und Methoden vorgestellt, mit welchen moderne Web Applikationen entwickelt werden können und die gefundenen Probleme umgehen. Es wurde der HTML5 Standard vorgestellt und einige der mächtigsten Neuerungen anhand von Beispielen gezeigt. Im Anschluss daran wurde Java 8, als Basistechnologie, und die Entwicklung von reiner Objekt-Orientierung hin zum funktionalen Paradigma, anhand von Beispielen, behandelt. Des Weiteren wurde der Spring Entwicklungsbaukasten und das Persistenz Framework Hibernate als wichtige Technologien für die Entwicklung komplexer Applikationen vorgestellt und deren praktische Verwendung in einem umfangreichen Beispiel erklärt. Parallel wurden Entwurfsmuster wie das Model View Controller (MVC) erklärt und gezeigt wie dies in Spring Web Framework umgesetzt wurde, und wieso es in modernen Applikationen notwendig ist, dieses Muster zu erweitern.

Im anschließenden Kapitel wurde die Umsetzung des Prototyps durch Verwendung der gezeigten Technologien und Methoden, beginnend mit der Anforderungsanalyse und der Illustration der verwendeten Architektur, vorgestellt. Daneben wurden auch die wichtigsten Domain-Objekte der Implementierung identifiziert und betrachtet. Für die Implementierung wurden zahlreiche Werkzeuge und Technologien verwendet die oberflächlich erklärt wurden. Es wurde die Implementierung, der für den Prototyp erforderlichen grafischen

Benutzerkomponenten behandelt und anhand von Beispielen die Verwendung dieser gezeigt.

Den Abschluss der Arbeit bildeten die Vorstellung der Benutzeroberfläche des Web Clients, des Cloud Desktop Client und die Diskussion der Implementierung der Server API. Neben der erwähnten Kritik an der Implementierung und der Erwähnung von Verbesserungen, wurden auch zukünftige Erweiterungsmöglichkeiten für die entwickelte Plattform angesprochen.

## 7. Quellenverzeichnis

- [Bauer07] Bauer C., King G.: „Java Persistence with Hibernate“, Manning, UK (2007)
- [Bipa14] <http://de.wikipedia.org/wiki/Bipa> (Abruf vom 3.12.2014)
- [Caniuse14] <http://caniuse.com/#feat=svg> (Abruf vom 1.12.2014)
- [Caniuse1402] <http://caniuse.com/> (Abruf vom 5.12.2014)
- [Caniuse1403] <http://caniuse.com/#feat=input-file-multiple> (Abruf vom 5.12.2014)
- [Chainsaw14] <http://logging.apache.org/chainsaw/> (Abruf vom 15.12.2014)
- [Eisenberg02] Eisenberg, D. J.: „SVG Essentials“, Oreilly, USA (2002)
- [FirmenABC1401] [http://www.firmenabc.at/color-drack-gmbh-co-kg\\_HUAX](http://www.firmenabc.at/color-drack-gmbh-co-kg_HUAX) (Abruf vom 3.12.2014)
- [FirmenABC1402] [http://www.firmenabc.at/happy-foto-gmbh\\_GToI](http://www.firmenabc.at/happy-foto-gmbh_GToI) (Abruf vom 3.12.2014)
- [Förster11] Förster, K., Öggl, B.: „HTML5 Guidelines for Developers“, Addison-Wesley, USA (2011)
- [Foto14] <http://at.foto.com/> (Abruf vom 3.12.2014)
- [FotoCharly14] <http://www.fotocharly.at/> (Abruf vom 3.12.2014)
- [Frizot98] Frizot, M.: „Neue Geschichte der Fotografie“, Könemann, Köln (1998)
- [Fulton11] Fulton, S., Fulton, J.: „HTML5 Canvas“, Oreilly, USA (2011)
- [Gamma14] Gamma E., Helm R., Johnson R., Vlissides J.: „Design Patterns Elements of Reusable Object-Oriented Software“, Addison-Wesley, USA (2014)
- [Gupta13] Gupta A.: „Java EE 7 Essentials“, Oreilly, USA (2013)



- [HappyFoto14] <http://www.happyfoto.at/> (Abruf vom 3.12.2014)
- [Hikari14] <https://github.com/brettwooldridge/HikariCP> (Abruf vom 16.12.2014)
- [Hoy06] Hoy, A. H.: „Enzyklopädie der Fotografie“, National Geographic, Hamburg (2006)
- [Hsql14] <http://hsqldb.org/> (Abruf vom 16.12.2014)
- [JetBrains14] <https://www.jetbrains.com/idea/features/> (Abruf vom 15.12.2014)
- [Jquery14] <http://jquery.com/> (Abruf vom 16.12.2014)
- [JUnit14] <http://junit.org/> (Abruf vom 16.12.2014)
- [Konsument14] <http://www.konsument.at/computer-telekom/online-fotoausarbeitung> (Abruf vom 2.12.2014)
- [Log4j14] <http://logging.apache.org/log4j/1.2/> (Abruf vom 16.12.2014)
- [Lubbers11] Lubbers P., Albers B., Salim F.: „Pro HTML5 Programming“, Apress, USA (2011)
- [Maven14] <http://maven.apache.org/> (Abruf vom 16.12.2014)
- [Miano99] Miano, J.: „Compressed Image File Formats“, Addison-Wesley Professional, USA (1999)
- [Newhall98] Newhall, B.: „Geschichte der Photographie“, Schirmer Mosel, München (1998)
- [Oestereich04] Oestereich B.: „Objektorientierte Softwareentwicklung – Analyse und Design mit der UML 2.0“, Oldenbourg, München/Wien (2004)
- [Oracle14] <http://www.oracle.com/technetwork/java/javase/downloads/sb2download-2177776.html> (Abruf vom 15.12.2014)
- [Pixum14] <http://www.pixum.at/> (Abruf vom 3.12.2014)
- [Reelsen11] Reelsen, A.: „Play Framework Cookbook“, Packt Publishing, UK (2011)

- [RegexBuddy14] <http://www.regexbuddy.com/> (Abruf vom 15.12.2014)
- [Rowell11] Rowell, E.: „HTML5 Canvas Cookbook“, Packt Publishing, UK (2011)
- [Schaefer14] Schaefer C., Ho C., Harrop R.: „Pro Spring“, Apress, USA (2014)
- [SimpleLog4j14] <http://www.slf4j.org/> (Abruf vom 16.12.2014)
- [Snapfish14] <http://www.snapfish.at/> (Abruf vom 3.12.2014)
- [Sonka08] Sonka M., Hlavac V., Boyle R.: “Image Processing, Analysis, and Machine Vision”, Cengage Learning, USA (2008)
- [SqlLite14] <http://sqlitebrowser.org/> (Abruf vom 15.12.2014)
- [SqlLite1402] <http://sqlite.org/> (Abruf vom 16.12.2014)
- [Telerik14] <http://www.telerik.com/fiddler> (Abruf vom 15.12.2014)
- [Tiobe14] <http://www.tiobe.com/index.php/content/paperinfo/tpci/index.html> (Abruf vom 9.12.2014)
- [Urma14] Urma R., Fusco M., Mycroft A.: „Java 8 in Action“, Manning, USA (2014)
- [W3C14] <http://www.w3.org/html/wg/> (Abruf vom 5.12.2014)
- [W3C1402] <http://www.w3.org/TR/html5/the-xhtml-syntax.html#the-xhtml-syntax> (Abruf vom 5.12.2014)
- [W3tech14] [http://w3techs.com/technologies/overview/programming\\_language/all](http://w3techs.com/technologies/overview/programming_language/all) (Abruf vom 9.12.2014)
- [Westphalen13] Westphalen, C.: „Die große Fotoschule – Digitale Fotopraxis“, Galileo Design, Bonn (2013)
- [Wikipedia1401] [http://de.wikipedia.org/wiki/Camera\\_obscura](http://de.wikipedia.org/wiki/Camera_obscura) (Abruf vom 1.12.2014)
- [Wikipedia1402] <http://en.wikipedia.org/wiki/APNG> (Abruf vom 1.12.2014)
- [Wikipedia1403] <http://en.wikipedia.org/wiki/Snapfish> (Abruf vom 3.12.2014)

[Wikipedia1404] <http://de.wikipedia.org/wiki/Technologie> (Abruf vom 5.12.2014)

[Wikipedia1405] <http://en.wikipedia.org/wiki/NPAPI> (Abruf vom 5.12.2014)

[Wikipedia1406] <http://en.wikipedia.org/wiki/HTML> (Abruf vom 5.12.2014)

[Wikipedia1407] [http://en.wikipedia.org/wiki/Ajax\\_%28programming%29](http://en.wikipedia.org/wiki/Ajax_%28programming%29) (Abruf vom 5.12.2014)

[Wikipedia1408] [http://de.wikipedia.org/wiki/Ein\\_Bild\\_sagt\\_mehr\\_als\\_tausend\\_Worte](http://de.wikipedia.org/wiki/Ein_Bild_sagt_mehr_als_tausend_Worte) (Abruf vom 18.12.2014)