Michael Planitzer, MSc, BSc, BSc

Development of a mobile french language learning platform

Master's Thesis

to achieve the university degree of

Master of Science

Master's degree programme: Computer Science

submitted to

Graz University of Technology

Supervisor
Priv.-Doz. Dipl.-Ing. Dr.techn. Martin Ebner

Co-Supervisor
Dipl.-Ing. Markus Ebner

Institute of Interactive Systems and Data Science
Head: Univ.-Prof. Dipl.-Inf. Dr. Stefanie Lindstaedt

Graz, September 2020

Affidavit

Eidesstattliche Erklärung

Abstract

The possibility to offer students an own language learning application has been the impetus for this master thesis. As a co-product of the Uni Graz and the TU Graz a language learning platform to support students during their language classes was developed.

This master thesis is about a language learning platform named APPELL, which provides different features and tools. Students are able to use a glossary to look up vocabularies. They can practice and exercise different grammar types, which are supported by images, descriptions in form of documents and videos, voice records and related links. Multiple Choice (MC)-quizzes organized in different topics offer a playful approach to study general knowledge of languages. To utilize the actively and passively generated data and meta data, the platform uses Learning Analytics (LA) mechanisms to derive educational information for analytical purposes. LA is a very young field in education and teaching. It is getting more and more important in nowadays educational research. The theoretical part of this thesis covers the main topics of LA and tries to give a usage for language learning.

The main focus has been given to the development of the platform itself. To fit the needs for administrating the platform and providing a mobile application, the system was designed distributively with several independent software parts: a native mobile application, a Content Management System (CMS) developed as a web application and a REST API to offer a communication interface. With emphasis on the underlying software architecture and design patterns, the technical implementation, the applied software development process and descriptions of own and third party interfaces, this thesis provides deep insights into modern software development aspects.

Zusammenfassung

In Zusammenarbeit der Grazer Universitäten, der Uni Graz und der TU Graz, wurde im Zuge der vorliegenden Masterarbeit eine Sprachlernplattform entwickelt, die Studierende während Sprach-Lehrveranstaltungen unterstützt.

Die Plattform namens APPELL ist Kernelement der Arbeit und bietet unterschiedliche Funktionen: Ein Wörterbuch als Nachschlagewerk für Vokabeln, Lückentextübungen und Multiple-Choice (MC)-Fragen bilden den Grundstock der User-Applikation. Weiters stehen den Studierenden während dem Lernen Bilder, Sprachaufnahmen, Beschreibungen in Form von Dokumenten und Videos, sowie weiterführende, externe Informationen zur Verfügung. Um die sowohl aktiv als auch passiv generierten Daten und Metadaten zu verwerten, nutzt die Plattform verschiedene Learning Analytics (LA) Mechanismen, um bildungsrelevante Informationen für Analysezwecke zu erhalten. LA ist ein sehr junger Bereich der Bildungsforschung und nimmt einen immer größeren Bereich darin ein. Der theoretische Teil dieser Arbeit behandelt die Hauptthemen von LA und versucht Anwendungen für das Erlernen von Sprachen abzuleiten.

Um die verschiedenen Anforderungen zu erfüllen, wurde die Plattform als verteiltes System mit unabhängigen Softwarebestandteilen, bestehend aus nativ entwickelten Mobilapplikationen, einem Content-Management-System (CMS) in Form einer Webapplikation und einer Kommunikationsschnittstelle, entwickelt. Mit Fokus auf die zugrundeliegenden Softwarearchitektur, die Software-Entwurfsmuster, die technischen Umsetzung, den angewandten Softwareentwicklungsprozess, die Beschreibung eigener und externer Schnittstellen, nehmen technische Aspekte einen wesentlichen Teil der Arbeit ein.

Contents

List of Figures

List of Tables

List of Code Snippets

List of Abbreviations

API	Application Programming Interface
APPELL	App Easy Language Learning
CALL	Computer Assisted Language Learning
CMS	Content Management System
CRUD	Create, Read, Update and Delete
DBAL	Database Abstraction Layer
DNS	Domain Name System
EDM	Educational Data Mining
EER	Enhances entity-relationship
GNU GPL	GNU General Public License
GUI	Graphical User Interface
HMAC	Uash-based Message Authentication Code
HTML	Hypertext Markup Language
HTTP	Uyper Text Transfer Protocol
IDE	Integrated development environment
JSON	JavaScript Object Notation
LAK	Learning Analytics and Knowledge
LA	Learning Analytics
LMS	Learning Management Systems
MC	Multiple Choice
MIT	Massachusetts Institute of Technology
MVC	Model-View-Controller
MVP	Minimum Viable Product

List of Code Snippets

NLP	Natural Language Processing
ORM	Object Relational Mapper
OS	Operating System
PDF	Portable Document Format
PHP	PHP: Hypertext Preprocessor
PLE	Personal Learning Environment
PP	path parameter
QSP	query string parameters
REST	Representational State Transfer
SaaS	Software as a Service
SDK	Software Development Kit
SOAP	Simple Object Access Protocol
SQL	Structured Query Language
SRL	Self-regulated Learning
TU Graz	Graz University of Technology
UI	User Interface Design
URL	Uniform Resource Locator
UX	User Experience Design
Uni Graz	University Graz
ZF	Zend Framework

1 Introduction

The way how people - especially young people - are learning, is changing rapidly. When this project started in 2017, no one could have ever known that a few years later millions of students, from elementary schools up to universities, would be dependent on technology so much as it began during the corona crisis in 2020. The faster spread of the internet, its transmission rates, its availability, makes it possible for most of the students to access innumerable learning tools independently of place and time instantly. All those facts and developments were leading to a very young field of education and teaching in the last decades: **Learning Analytics (LA)**.

As a result and byproduct of the big data revolution, LA deals with learning data and meta data generated explicitly and implicitly from appropriate learning applications. The aim of LA is to improve learning methods by using big (learning) data to offer new ways to support learners. Implemented in different learning situation, LA provides extensive cuts especially for teachers and students at school. Teachers are no longer the only knowledge owners and providers. They are becoming more and more organizers of information and education tools. Learners may individually decide what, when, how and where to learn.

This master thesis is about a language learning platform that generates learning data to be used for analytics purposes, called **APPELL**. Emphasis has been put on technical aspects: the process, implementation and usage of a distributed software system in terms of learning analytics.

The need to develop a language learning platform arose by the *treffpunkt sprachen - Centre for Language, Plurilingualism and Didactics*[1], a division of the Uni Graz. It work along with the staff unit *Lehr- und Lerntechnologien - Educational Technology*[2] of the Graz University of Technology (TU Graz), which is working on a teaching and information community:

> *" „The Power of the People" – this is the motto of the Educational Technology Service Department. The aim is to build a great teaching and learning community at the TU Graz that meets the demands of the information society of the future. The service department provides access to media and technologies that can be used to support teaching, learning and the development of both didactic and pedagogical methods. "*
>
> [Educational Technology (2020)]

APPELL is part of the **Learning Lab**. It is a service, which is run by the staff unit that *"offers different tools, learning objects and research projects in the field of technology enhanced teaching and learning."* (Learning Lab, 2020)

The name APPELL has a double meaning: on the one hand APPELL is a german word (*"der Appell"*) which means *"the appeal"* in English - it should insistently remember students to start or continue with their exercises. On the other hand it is an abbreviation and stands for **APP Easy Language Learning**. It can support students during their language courses. Typical supporting tools are cloze exercises, MC-Quizzes, translations, voice records, video information and much more. Analytic tools provides the teacher various anonymized information, like how long and how often screens and elements are viewed or which exercises are interrupted.

The logo (see Figure 1.1) is showing a globe inside of a speech balloon, which symbolizes the different countries with their different languages.

[1] Visit: `https://treffpunktsprachen.uni-graz.at/` (accessed 20-June-2020)

[2] Visit: `https://www.tugraz.at/en/tu-graz/organisational-structure/service-departments-and-staff-units/educational-technology/` (accessed 20-June-2020)

Designed as a distributed platform, APPELL consists of three parts:

- natively developed mobile applications, downloadable in both Google's Playstore[3] and Apple's App Store[4],
- a CMS developed as a web application
- and a REST API.

The development of this thesis was divided into two parts: a practical and a theoretical part. The scope of the thesis is the natively developed iOS mobile application, the CMS and the REST API. It is important to clarify that a previous work was done concerning the Android application, which has been developed in the course of an independent Bachelor thesis, started after the completion of the practical parts of this thesis. The software project process itself is oriented on the agile process framework Scrum. Every step of each iteration has been covered by the scrum user roles described in section 3.1.

This thesis is structured in five parts. It starts to provide a theoretical background about learning analytics like definitions, approaches, aims and dimensions. The second part is about the software concept. The software concepts covers not only the design of the whole software platform and the Minimum Viable Product (MVP), but also the history and the involved persons are described there. The third part describes the technology stack and the fourth part gives a detailed insight in the technical aspects, the used environments, the database scheme, the different software applications and the API documentation. How LA methods are applied and might be applied in future is covered in chapter 6. The penultimate chapter evaluates the whole project. The thesis ends by giving a summary and an outlook.

The author of the thesis wants to invite readers to download APPELL from the appropriate store to get a better feeling about the described topics.

[3] Visit: `https://play.google.com/store/apps/details?id=at.tugraz.appell` (accessed 20-June-2020)

[4] Visit: `https://apps.apple.com/at/app/appell/id1449511915?l=en` (accessed 20-June-2020)

Figure 1.1: App logo [Image created by the author of this thesis.]

2 Learning Analytics

Learning Analytics (LA) is a very young field in education and teaching. It is getting more and more prominent by the ongoing spread and availability of the internet, computers, smartphones and its main product **data**. Learners for example, who are using software to receive learning contents, leaves clicks, navigation patterns, timestamps and many other useful data and metadata. But there are not only learners producing such data, all the different persons *(stakeholders)* involved in the learning process (see subsection 2.1.2) are producing learning data. To analyse this data and metadata produced actively or passively are given the main focus in LA investigations. In the last decade LA has preserved its legitimacy in education alongside the data revolution.

In the literature there are various definitions for LA, but there is still no agreement for one general definition. One of the most cited definition is that from the first conference for *Learning Analytics & Knowledge* held in Banff in 2011, where George Siemens and colleagues defined learning analytics as

> *" the measurement, collection, analysis and reporting of data about learners and their contexts, for purposes of understanding and optimising learning and the environments in which it occurs "*
>
> [1st International Conference on Learning Analytics and Knowledge 2011, 2010]

LA is usually embedded in an iterative process starting with collecting educational data. This raw data has to be *pre-processed* by reducing, transforming or modelling to achieve relevant data by doing analytics, visualizations and further actions. In the *analytics stage* different LA techniques and method can be performed to explore hidden patterns to receive new knowledge. The main goal in the analytics stage includes additional actions as predictions, assessments, monitoring, recommendations, reflections, interventions and personalization. To increase the success of the LA process a *post-processing* session is indispensable. Compiling new data by extending or modifying the pre-processed data provides new input factors for the next iteration. (Chatti et al., 2012, pages 5–6)

When comparing LA with conventional data analysis the main difference is that modern LA software compares a student's activity with others in class, previous classes, or in classes abroad, and this create a model to understand how each student is likely to fail. (Yu, 2015)

2.1 What-Who-Why-How approach

Chatti et al. (2012) described a reference model for LA based on four dimensions: *What? Who? Why? and How?* which can be perfectly applied to the definition cited before.

2.1.1 What?

data, context and environment **[Keywords in definition]**

Learning Analytics LA exists from educational data, which is feeded by two big educational sources: centralized educational systems and distributed learning environments. Centralized educational systems are represented by Learning Management Systems (LMS) like *Blackboard* or *Moodle*. Those systems are based on traditional teaching methods that are transformed from the classroom to an e-classroom. Distributed learning environments are independent on place and time and represented by the Personal Learning Environment (PLE). PLE helps students to manage their own learning way, encourages their skills to manage, monitors and reflects their learning progress. In a PLE students can also extract knowledge from different external sources, like YouTube videos, social media or forum posts.

Klašnja-Milićević, Ivanović, and Stantić (2020, page 237) organizes LA data as following:

- Learner characteristics like demographic data, prior knowledge, psychometric characteristics, learning strategies, academic performances and competencies.
- External data captured from social media platforms, networks or searches in online libraries.
- Data generated from online learning environments such as timing and frequency of activities.
- Curriculum information like exam paths or expected learning outcomes integrated into the analysis.

2.1.2 Who?

Learners **[Keywords in definition]**

The target group of LA are the persons involved in the learning process. The so-called stakeholders are build from students or learners, teachers or educators, educational institutions, researchers and administrators. All of them have different intensions:

Learners want to understand the learning subjects and extend their knowledge.

Educators try to evaluate and improve their education methods.

Educational institutions want to figure out non-mean students or apply tools for their decision making.

Researchers explore and share their knowledge and experiences with the other stakeholders.

2.1.3 Why?

understanding, optimizing **[Keywords in definition]**

The objectives and purpose of LA depend on the point of view of the stakeholders. Implicitly some of them were already mentioned in the *Who?* dimension. For teachers or educational institutions objectives are *monitoring and analyzing* student activities to support the decision-making process, *prediction, intervention and adaption* to lead the student during the learning process. For both teachers and students LA gives opportunities for *assessment, feedback, awareness and reflection*. In a distributed learning environment LA offers *personalization and recommendations* to guarantee learning success.

Klašnja-Milićević, Ivanović, and Stantić (2020, pages 236–237) abstracts the benefits of LA concerning its temporal use into three different viewpoints:

Descriptive - *Past*: LA benefits stakeholder by analyzing the learning activities, understanding and monitoring habits and progress of the learners as well as the way to achieve their objectives. It also compares learning sequences.

Actual time - *Present*: Refers to the facility of receiving the feedback in real-time, it provides support to automated interference, collaboration and appraisals.

Predictive - *Future*: By learning paths optimization, enhance engagement, capacity to follow proposed recommendations and boost achievement levels.

2.1.4 How?

measurement, collection, analysis and reporting **[Keywords in definition]**

There are mainly methods or techniques to detect hidden patterns. *Statistics* illustrate different quantities, such can be for example page visits, time online or accessed material. To visualize facts and numbers is the purpose of *Information Visualization (IV)*. Visualized information like charts, scatterplots or maps can be better used by people. *Data mining* is the fundamental process to discover hidden patterns. Bharati and Ramageri (2010, page 1) summaries data mining as a logical process to search large amount of data and therefore find useful data. It can also be called knowledge discovery in databases (KDD). Analyzing data, relations, linkages, networks of social networks are increasingly getting popular. Those *Social Network Analysis (SNA)* are modeled by a graph, which is represented by nodes and edges.

Chatti et al. (2012) illustrates their approach in 2.1. The inner part of the illustration shows the dimensions and some examples while the outer part lists each input quantities.

2.2 EDM vs LAK

Growing interest in this topic produced two research communities: **Educational Data Mining (EDM)** and **Learning Analytics and Knowledge (LAK)**. The two areas have a significant overlap. Both communities are based on a data-driven approach to education with the aim to improve education quality by understanding education problems better. As a result of this improved understanding, advanced solution approaches are generated and applied to stakeholders such as administrators, teachers and students. (Siemens and Baker, 2012, pages 252–253) But there are also various differences between the communities. While LAK tries to understand the whole learning experience holistically, EDM is more individual, inductive and reductionist. Siemens and Baker (2012, page 253) identify five distinctions:

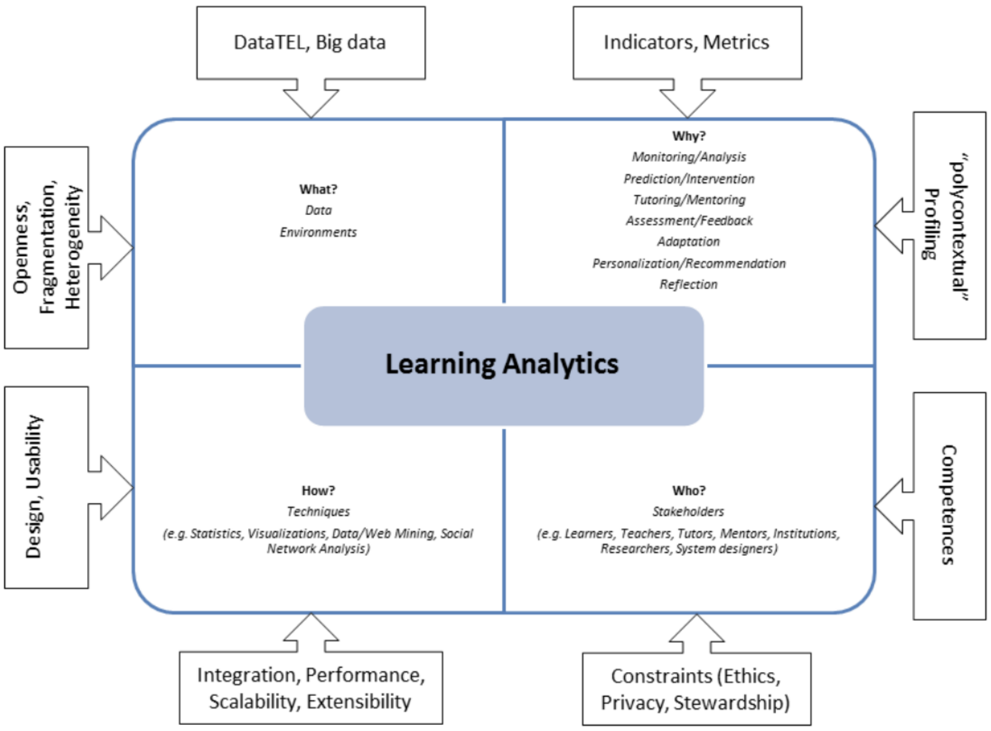

DataTEL, Big data
Indicators, Metrics
Openness, Fragmentation, Heterogeneity
"polycontextual" Profiling
What?
Data
Environments
Why?
Monitoring/Analysis
Prediction/Intervention
Tutoring/Mentoring
Assessment/Feedback
Adaptation
Personalization/Recommendation
Reflection
Learning Analytics
Design, Usability
Competences
How?
Techniques
(e.g. Statistics, Visualizations, Data/Web Mining, Social Network Analysis)
Who?
Stakeholders
(e.g. Learners, Teachers, Tutors, Mentors, Institutions, Researchers, System designers)
Integration, Performance, Scalability, Extensibility
Constraints (Ethics, Privacy, Stewardship)

Figure 2.1: Learning Analytics reference model [Chatti et al., 2012, page 7]

- **Discovery**
  EDM tries to automate discovery and uses human judgment as a tool to accomplish this. LAK on the opposite, automate discovery to achieve the goal of leveraging human judgement.
- **Reduction & Holism**
  EDM reduces systems to components and analyses them and their relationships. LAK wants to understand the whole systems.
- **Origins**
  EDM's origins are in educational software and student modelling while LAK is rooted in semantic web, "intelligent curricula", predicting outcomes and systemic interventions.
- **Adaption & Personalization**
  EDM focuses on automated adaption, whereas LAK informs and empowers instructors and learners.
- **Techniques & Methods**
  EDM works with visualization, clustering, classification, relationship mining, bayesian modeling and discovery with models while LAK focuses on different analysis and analytics.

The communities are not working against each other. It is more a friendly competition with a healthy rivalry. By learning from one another, both are striving to reach the maximum for education.

2.3 Learning analytics in language learning

LA can be applied to any domain of learning. This thesis focus on the domain of language learning. Applying learning analytics in language learning has a late start in research. Languages are complex. They are often using non-compositional patterns, which makes it hard to generate analyzable data. Data, as the fundamental good for LA, has to be generated by appropriate processes for each domain. In the language domain the processes are *corpus linguistics* and *Natural Language Processing* (*NLP*).

According to Brysbaert, Mandera, and Keuleers (2017) corpus linguistics refers *to the study of language through the empirical analysis of large databases of naturally occurring language*. In corpus linguistics the whole context, naturally occurring bodies of texts are observed and considered. The corpus itself is a huge collection of relevant language documents which are feeded to machine learning algorithms. The output of the learned system are tools (algorithms, databases) that solve problems. (Entrikin, 2012)
The tools generated from corpus linguistics are then applied in NLP to better understand patterns, structures and roles of language in daily life. NLP is usually used to transcribe oral language to text-based data. Based on this data LA can perform semantical analysis to start the learning analytics process. (ElAtia, Ipperciel, and Zaiane, 2016, page 256)

In language education, especially in foreign language teaching, Computer Assisted Language Learning (CALL) is often used as a teaching approach. LA is slowly getting more and more implemented in CALL applications. CALL systems are typically organized to perform Self-regulated Learning (SRL). SRL includes the cognitive, motivational, and emotional aspects of learning. This core conceptual framework has become a very important research area within educational psychology. (Panadero, 2017) Since SRL is very individual, it is hard to measure students' learning situation. Possibilities for guiding and instructing are limited for teachers. Applying LA can help to change this situation and offer tools to support both, teachers and students. (Li et al., 2017)

Youngs, Moss-Horwitz, and Snyder (2015, page 348) did a descriptive study in 2015 about analyzing data from a language CALL course, French online elementary 1, at their university. They prove that EDM can help to answer questions such as:

- How much time do students spend on lessons, sections of lessons, exercises for lessons?
- What do students do when they have questions?

- Do they continue without the answer, or do they return to an explanation and then retry the exercise?
- If the time is not 'equivalent' to the time students spend in a traditional course, does this mean that the online learner is disadvantaged in some way?

Collecting and analyzing such education data can be used to show students hidden behavior and later to improve a course design. Youngs, Moss-Horwitz, and Snyder (2015, pages 365–366) underlines that data mining can help figure out students behavior more directly than relying only on observations.

3 Software Concept

This chapter is not only about the software concept itself. The author has given a substantial priority to the way of how the final software concept has been elaborated. In order to understand how the software concepts arose, an introduction into the history, the involved persons and their roles are indispensable.

APPELL started in 2017 as a result of the cooperation between the *treffpunkt sprachen - Centre for Language, Plurilingualism and Didactics* of the Uni Graz and the staff unit *Lehr- und Lerntechnologien - Educational Technology* of the TU Graz. Representing the Center for Language, Carole Bourgadel, university lecturer for French as a foreign language at the Uni Graz, has started the initiative to implement a language learning app based on already existing language platforms. She was the driving force and has overtaken the initial conception in terms of customer requirements. She was involved in every development iteration and validaton. The Educational Technology staff unit under the head of Martin Ebner nominated Markus Ebner, PhD candidate and employee there, to supervise the technical development and to overtake a consulting, especially in project management and technical matters. His main focus was on verification - *implementing things correctly*, and the initial conception in terms of development requirements. For deploying and server infrastructure Josef Wachtler, also PhD candidate and employee at Educational Technology, is responsible for integrating APPELL into the Learning Lab platforms and hosting. The main tasks of the author of this thesis, was to help to develop a MVP and to implement the whole platform according to its requirements.

After initial meetings by all the involved persons and after estimating the project scope, in both customer and development requirements, the software concept and the MVP became more clear. Then the project aimed to an agile software project process oriented to the agile process framework Scrum. In retrospective the agile process covered especially UI/UX design decisions and evaluations, feature changes played a minor role.

In the following sections a more detailed overview is presented. The involved persons and roles, the requirements that had to be fulfilled, the elaborated MVP and how the MVP can be wrapped in a technical implementation and the system architecture are explained below.

3.1 Responsible Persons

As mentioned before, several persons were included during the software development process. The roles of the persons fits to the typical roles of the Scrum framework. Schwaber (2004, pages 5–6) enumerates and describes the basic three roles of the Scrum framework as:

- **Product Owner**
  The Product Owner represents the stakeholders and the customer. The Product Owner is responsible for the business results, to build the right product. To guarantee good results the Product Owner defines customer requirements, sets release plans and deadlines, negotiates priorities and ensures that the product backlog is visible, and clear. This role was overtaken by Carole Bourgadel.
  The typical stakeholders of APPELL are the students who are using the app, and the *treffpunkt sprachen* division plays the role of a customer.
- **Development Team**
  As its name suggests, Scrum's Development Team is responsible for the software development itself. It usually consists of a group of cross-functional people, self-managing and self-organizing. But in

APPELL's case only the author of this thesis was in charge of all this tak. It is important to highlight that tasks as deploying, hosting and integrating, were done by employees of the *Educational Technology* unit.

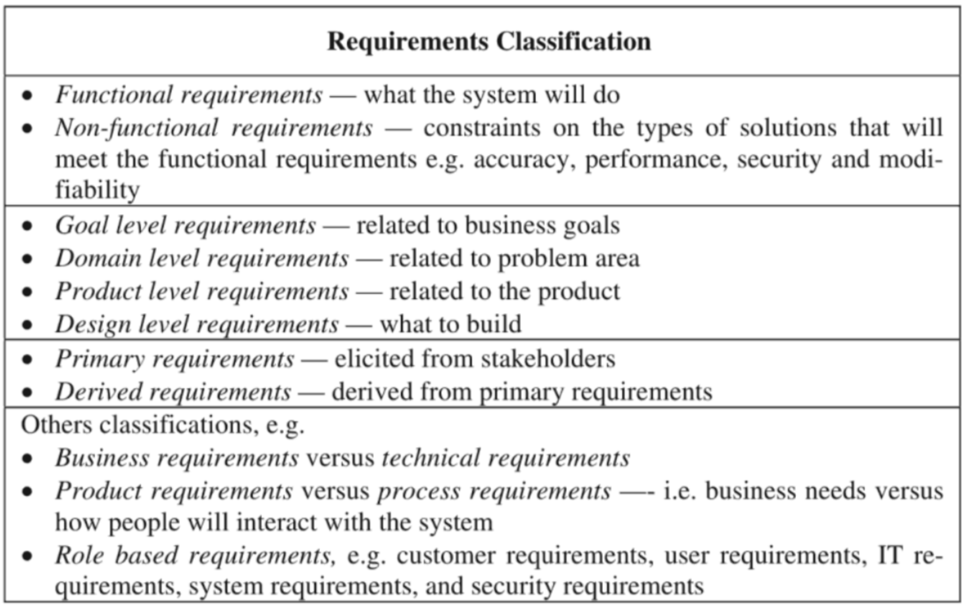
- **Scrum Master**
  The Scrum Master can be compared to a classical project manager. The Scrum Master ensure that the agile workflow of Scrum is complied and also supports the Product Owner and the Development team if any input or help is needed. The Scrum Master provides process leadership. (Rubin, 2012, pages 185–186)
  This role wos overtaken by Markus Ebner.

3.2 Requirements

There are a lot of different definitions of requirements in literature. As Aurum and Wohlin (2006, page 4) show in the figure 3.1 there is a big difference between functional *"what the system will do"* and non-functional *"system attributes, like accuracy, performance or security"* requirements. They can be classified depending on their relations in primary and derived requirements, and many other kinds. Due to its history, APPELL make a distinction between the origin of the requirements, customer requirements and development requirements. Those requirements classification are role based. The customer requirements were defined by the customer, in this case the Product Owner who represents the stakeholders. The development requirements were defined by the *Educational Technology* unit.

3.2.1 Customer requirements

The need to develop an own language learning platform has been evaluated by Carole Bourgadel. There is no language learning platform avail-



Requirements Classification

- *Functional requirements* — what the system will do
- *Non-functional requirements* — constraints on the types of solutions that will meet the functional requirements e.g. accuracy, performance, security and modifiability

- *Goal level requirements* — related to business goals
- *Domain level requirements* — related to problem area
- *Product level requirements* — related to the product
- *Design level requirements* — what to build

- *Primary requirements* — elicited from stakeholders
- *Derived requirements* — derived from primary requirements

Others classifications, e.g.

- *Business requirements* versus *technical requirements*
- *Product requirements* versus *process requirements* —- i.e. business needs versus how people will interact with the system
- *Role based requirements,* e.g. customer requirements, user requirements, IT requirements, system requirements, and security requirements

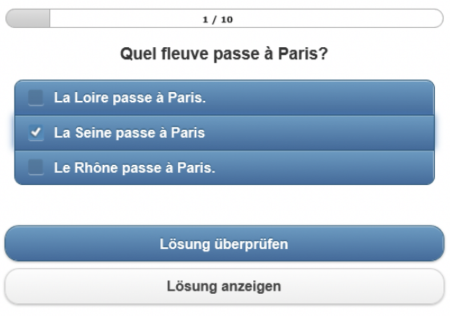
Figure 3.1: Types of requirements [Aurum and Wohlin, 2006, page 4]

able which fulfills the need to support students during language learning classes at the level B1 and B2.

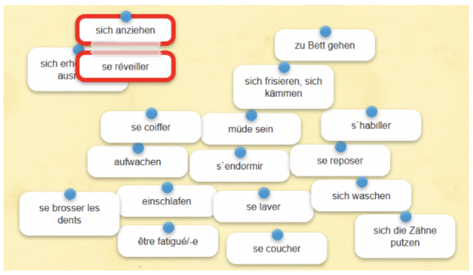
The original requirements communicated at the beginning of the project can be summarized with the following statement:

> *" A self-organizable, mobile application to support students in their French as foreign language (niveau B1 and B2) with main focus on lexical, grammatical and snytactical categories should be implemented. "*
>
> [Carole Bourgadel - *treffpunkt sprachen - Centre for Language, Plurilingualism and Didactics*]

A more detailed and unstructured definition of the requirements is listed below:



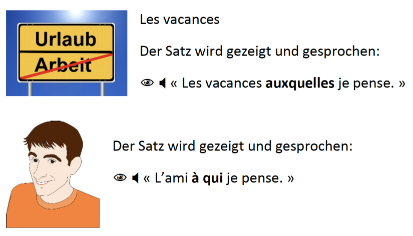
(a) Quiz as a fun feature.



(b) Memory to memorize vocabular more easily.

Figure 3.2: Different possible realizations were given. (a) shows how a quiz can look like. (b) shows a memory version to find correct translations. [Both images created by Carole Bourgadel.]

- **Self-organizable data source**
  All the different data, that need to ensure subsequent features, should be easily maintainable and extendable by administrators of the app. A web interface should offer a CMS software.
- **Multimedia**
  Various multimedia files should be provided and therefore be more descriptive and closer to the users.
  - **Images** to improve the app graphically.
  - **Audio** files to improve the listening comprehension.
  - **Videos** to improve the listening comprehension together with visual effects, to give simple explanations or to watch different mouth gestures.
- **Glossary**
  A collection of all the included words and phrases should be indexed and able to be easily lookedup within a glossary.
- **Explanations**
  Explanations have to be provided in form of documents and videos.
  - **Weblinks** that links to information in the web.
  - **Documents** in form of PDF able to download, share and print.



Les vacances

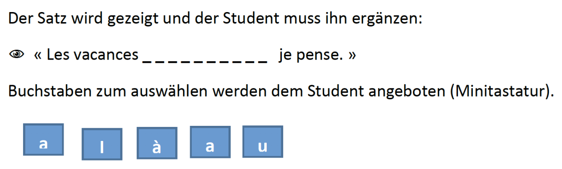
Der Satz wird gezeigt und gesprochen:

« Les vacances **auxquelles** je pense. »

Der Satz wird gezeigt und gesprochen:

« L'ami **à qui** je pense. »

(a) Listening and reading

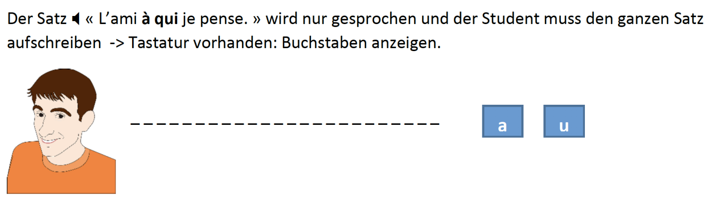


Der Satz wird gezeigt und der Student muss ihn ergänzen:

« Les vacances ___________ je pense. »

Buchstaben zum auswählen werden dem Student angeboten (Minitastatur).

(b) Writing



Der Satz « L'ami **à qui** je pense. » wird nur gesprochen und der Student muss den ganzen Satz aufschreiben -> Tastatur vorhanden: Buchstaben anzeigen.

(c) Listening and writing

Figure 3.3: The sequence of how exercises should be studied. It begins with listening and reading (a), and then continue with cloze tests. (b) The difficulty increase at the end of each exercise. (c) [All three images were created by Carole Bourgadel.]

  - **Videos** are already existing a private YouTube channel with short videos describing different grammars, exercises, etc. Those videos have to be included into the app.

- **Exercises**
  The core feature are the exercises. They have to be supported in a training and challenging mode. In training mode all available information should be consumable. The way how exercises should be designed and arranged is described afterwards.
- **Feedback**
  A feedback function has to be implemented to receive direct feedback from students. The feedback should then be listed within the CMS.
- **Quizzes**
  Quizzes with questions about general knowledge, both single and multiple choice, should introduce a fun factor into the app. A reference image is given in 3.2a explaining how the quiz can look like.

Depending on the the available information, exercises can offer images, videos, documents, external web information and audio sequences. With these information students can study exercises as it is shown in the figures 3.3.
In step 3.3a *Listening and reading* the sentence with the targeting phrase in bold is shown. The student can listen to the audio sequence and learn how to pronounce it properly. In the next step 3.3b *Writing* the students have to fill in displayed letters in the gaps, which are replacing the targeting phrase. In the last step 3.3c *Listening and writing* all the sentence has to be written. Every letter is now replaced by a gap.

3.2.2 Development requirements

Development requirements can be derived from customer requirements. Martin Ebner summarized and generalized the development requirements as following:

- store, modify, provide and delete data
- render and access web pages for a user friendly data management
- provide data by an interface to be accessible by mobile applications

In a more technical context it means that a web application has to be similar to a *CMS* to be accessible via the internet. That web application works with a database for the data management, which has then to provide this data via an *Application Programming Interface (API)* to separately developed mobile applications.

The web application part of the software platforms has to be embedded into the TU Graz *Learning Lab* environment. *Learning Lab* core framework builds a web application based on Zend Framework (ZF)[1], which is implemented in PHP and covers typical web applications requirements, like

[1] Zend framework has been renamed into *Laminas* in February 2020.

those listed above. To ensure typical software requirements, like maintainability and extendability, the responsible persons decided to implement all web-based software components for *Learning Lab* software projects with ZF.

Due to the fact that different *Learning Lab* software projects need a protected area, especially for data management, a user-management module is provided by the *Learning Lab* core framwork. This user-management module uses a MySQL database, which is the reason why the responsibles suggested to use MySQL database as well.

Finally the responsible defined to implement mobile applications natively. The characteristics of native mobile apps are distributed app stores, developed especially for their mobile operation system and not compatible with other mobile OS.

Three development requirements are summarized below, which later leads to three pre-conditions for the technology stack:

- Zend framework
- MySQL database
- Native mobile apps

As an addition to the features to meet the customer requirements, Martin Ebner requires analytics tools based on the results of the theoretical part of this thesis: to track the students' effort of studying, their app usage, the events of started and closed exercises and similar LA data should be anonymized collected, stored and be available for educators. Also the platform should be expandable to support multiple language courses and multiple administrators.

3.3 System architecture

Resulting from the Development requirements in 3.2.2 a system architecture has to be distributed. Systems are called distributed, when their resources used by parts of the system are deployed on different computers and these ones are connected by a network. Different services enables the communication between the distributed software applications. A distributed application is an application that needs at least two different software applications deployed on different computers within the distributed system. The well known server-client model is the most popular representative of distributed systems. Three-tier, n-tier and peer-to-peer systems are further examples. There are several essential benefits of a distributed application. They are highly scalable, reliable, very performable and have the ability to spread processing and resources based on their requirements (Anthony, 2015, page 8). Nevertheless, developing a distributed application is much more complex and correlates with more development effort, also it needs more experience and know-how on different software and system components.

As shown in 3.4 the concept for APPELL's system architecture is distributed regarding several software components on client side and server side connected by the internet as the network:

Client-side On client side is the frontend running by a web browser and the mobile applications to add, modify, delete and view data requested and responded via HTTP.

Server-side On server side are running different server applications, mainly the web server, application server to run the business logic (backend) and the database server to run the database.

Network The internet as the network or cloud connects client side to server side. The servers are then connected via a local network.

If a client application wants to get data from the database, it has to perform a HTTP request to the web server. The web-server is then performing database queries, serializing the fetched data and sending it back to the client. Received at the client, it can be transformed again, prepared graphically and presented to the user.

3.4 MVP

MVP stands for *Minimum Viable Product* and was firstly used in 2001. Nguyen-Duc et al. (2020, page 83) defines it as ”a proxy of the final product that requires the least effort to develop but obtain maximum learning.”

The MVP for APPELL is therefore a mixture out of Development requirements and Customer requirements:

- Distributed software platform consisting of a native iOS application, a CMS web application with a frontend and a backend, and an API.
- Possibility for delimited users to perform Create, Read, Update and Delete (CRUD) actions.
- Native downloadable iOS mobile app with following functionalities:
  - Register user to user manager
  - Structured exercises with cloze tests, supported by images, audio files, pdf documents and external links
  - MC and SC quizzes
  - Glossary with lookup functionality

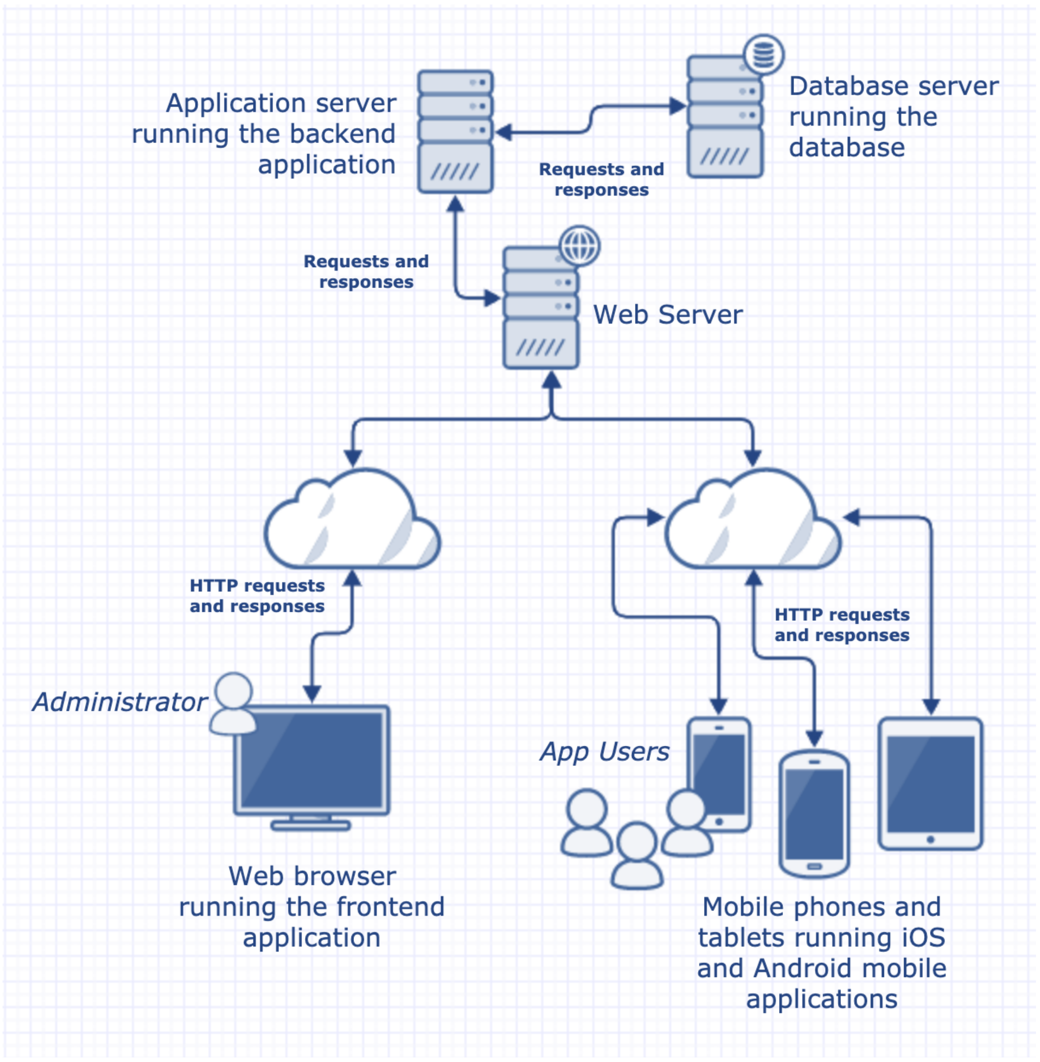

Application server running the backend application
Database server running the database
Requests and responses
Requests and responses
Web Server
HTTP requests and responses
HTTP requests and responses
Administrator
App Users
Web browser running the frontend application
Mobile phones and tablets running iOS and Android mobile applications

Figure 3.4: Distributed system architecture concept [Image created by the author of this thesis.]

4 Technology Stack

The technology stack describes the set of technologies that are used to build a software platform. Single applications usually have a very small tech stack, while distributed software applications use much more technologies. APPELL is a distributed software platform that has also dependencies to existing software components. Several mandatory and deducted technologies emerge from the Requirements in 3.2. Following the the Development requirements in 3.2.2 arise three pre-conditions for the technology stack:

- Zend framework
- MySQL database
- Native mobile apps

From these pre-conditions, dependent technologies can be derived: due to the fact Zend framework 3 is implementend in PHP, PHP 5.6 is used as the core web application programming language. As an object-relational mapping framework for MySQL and PHP Doctrine 1.1 is providing an easy database access. For communication between the PHP web application and the mobile apps, Apigility 1.4 offers a state of the art API builder for Zend projects. Native mobile apps are typically implemented with the most recent Swift version for iOS, or the most recent Java or Kotlin versions in Android. As mentioned before the scope of the present master thesis is to develop a web application and a native iOS mobile app for iOS. Swift 4.2 is used for developing the iOS App with a deployment target minimum iOS 11.0.

Originally planned to develop a short analyse tool within APPELL, the increasing project scope and the the time needed for that forced to a re-thinking about how to ensure this essential feature. A reliable, established, libre software has to be found. After a short research it turned out that Google's Firebase[1] seemed to be an appropriate candidate.

The total used technology stack consists of:

- Zend framework 3 with PHP 5.6
- MySQL 8.0 database
- Doctrine 1.1 ORM
- Apigility 1.4 API builder
- Swift 4.2 for iOS 11.0
- Firebase

The next sections and subsections give a brief information about concepts, fundamental functionalities, applications, versions and licensing of these technologies.

4.1 MySQL

MySQL is an open source SQL database management system from Oracle Corporation. It is relational, open sourced, very fast, reliable, scalable, and easy to use as its developer Oracle (2020) is applying it on their website. MySQL is written in C and C++ and works on many platforms. Under the terms of the GNU GPL MySQL is free and therefore very popular for small projects and open source projects. But also big companies like Facebook, Google or Youtube are based one MySQL databases, because of their nature under a proprietary licence. MySQL was developed in 1994 and is often used in combination with Apache or Nginx web servers, PHP as the programming language and Linux operation systems. This technology

[1] Visit: `https://firebase.google.com/` (accessed 24-July-2020)

stack is called LAMP (Apache web server) or LEMP (Nginx web server). (Tarr, 2011, page 239)

APPELL's MySQL database consists of 15 tables, 89 columns and 27 relations.

4.1.1 File storage

The requirement of storing audio files, images and documents demands a file storage. After a discussion with the *Learning Lab* administrators and an estimation of the required file storage in GB, files will be directly stored into the *Learning Lab* ZF project. Which means that data is not stored on a file server, but on the application server.

4.2 Zend framework

Zend Framework ZF is a web application framework consisting of multiple PHP packages and written in PHP. It is object-oriented, provides a large spectrum of language features and is licensed under the New BSD License - an open source licence. More than 570 millions installations makes ZF beside Laravel, Symfony, CodeIgniter to a widely used PHP web framework. Currently ZF stable release version is 3.0.1. and runs with both PHP 5 and PHP 7. Since February 2020 ZF is labeled under the name of *Laminas*. (Zend, 2020)

Zend Framework ZF implements software concepts, which are based on different design patterns. Those ensures to develope easy, fast, clean and structured code. Eggert (2017, pages 36–37) describes the concepts as following:

- **Dependency Injection**
  Dependencies of classes should be injected when needed and not

instantiated within a class. Factory method pattern helps to achieve this goal.

- **Event-driven Architecture**
  Class functionalities should not be called directly from other classes. The event should be triggered by a service in ZF the so-called event-manager, where different listeners are registered. The main advantage of those pattern is to handle asynchronous tasks and to reduce confusing class instances.
- **Module Manager**
  To encapsulate and reuse functionalities of an application a modular construction should be applied. Zend's module manager ensures to load and to configure modules.
- **MVC**
  Modern web application separates its program logic into interconnected elements: a model, view and controller - the so-called Model-View-Controller (MVC) software design pattern.

  Model contains the business logic of the application and implements the persistence of the data

  View presents the data and should only contains output logic; has read access to the model

  Controller ensures the interaction of the components; gives data to the model and also gets data from model to pass it to the view
- **Middleware concept**
  Middleware is a software used between frontend and backend applications. For example, it manages disparate requests to ensure a heterogeneous response.

All TU Graz *Learning Lab* projects are implemented in one great ZF 3 skeleton project. Each *Learning Lab* project like APPELL is then implemented in a encapsulated module. Global configuration such as database connection parameters, session configuration, used modules, etc. are maintained and administrated by *Learning Lab* administrators.

As a package and dependency manager ZF uses composer[2]. Composer can be compared with Node.js' npm, macOS' Homebrew or .NET's NuGet and runs from the command line.

Following packages are used in APPELL and managed by composer:

```
  "doctrine/doctrine-orm-module": "^1.1",
  "php": "^5.6",
  "zendframework/zend-component-installer": "^1.0 || ^0.7 ||
↪   ^1.0.0-dev@dev",
  "zendframework/zend-mvc": "^3.0.1",
  "zendframework/zend-mvc-form": "^1.0",
  "zendframework/zend-json": "^3.0",
  "zendframework/zend-soap": "^2.7",
  "zendframework/zend-session": "^2.8",
  "zendframework/zend-paginator": "^2.8",
  "zfcampus/zf-apigility": "^1.4",
  "zfcampus/zf-development-mode": "^3.0"
```

Code Snippet 4.2: Included PHP libraries by composer

The package names already tells a lot about the packages usages. Doctrine and PHP are self-explanatory. *zend-component-installer* is used for injecting modules and configuration providers into application configuration. *zend-mvc* and *zend-mvc-form* enables the event-driven MVC layer. *zend-json* provides methods for serializing PHP objects to JSON and to decode JSON again back to PHP objects which is necessary to communicate via the REST API. *zend-soap* allows to manage the SOAP protocol and therefor to communicate with the UserManager. *zend-session* takes responsibility for all aspects of classical session management like starting or destroying a session, setting the time-to-live and so on. *zend-paginator* ensures that only needed data to be displayed is transmitted. *zf-apigility* is used to build the REST API. *zf-development-mode* enables or disables development mode. In development mode for example caching is disabled, also the Apigility is only accessible in development mode.

[2] Visit: https://getcomposer.org/ (accessed 30-May-2020)

4.2.1 PHP

Zend FrameworkZend Framework is implemented in PHP: Hypertext Preprocessor (PHP) and therefore code has to be written in PHP. PHP is a scripting language suited for web development. It is open source, first appeared in 1994 and now is in its 7th version. (PHP, 2020b)

PHP code is executed on the server which allows HTML-files to be displayed by web browsers: clients send requests to a web server to GET back the needed web pages. The web server processes the requests and loads the required PHP source files from the storage and forwards them to the PHP interpreter. The PHP interpreter finally parses the source files and send the interpreted HTML-files back to the client.

It supports all major operating systems, like Linux, Microsoft Windows, macOS, etc. Moreover it can be used by most of the web servers, including Apache, IIS, Nginx. (PHP, 2020a)

Due to backward compatibility reasons of *Learning Lab*, APPELL had to reduce the recommended PHP version from 7 to 5.6.

4.2.2 Doctrine

Doctrine is a set of libraries to provide data persistance in PHP applications. It is focused on database storage and object mapping which is achieved by the Object Relational Mapper (ORM) and the Database Abstraction Layer (DBAL). Doctrine is open source, licensed under MIT license and supports all the popular PHP frameworks like Laravel, Symfony or Zend Framework on one hand and all the popular relational database management systems like MySQL, PostgreSQL, MS SQL or Oracle on the other hand. (Doctrine, 2020)

Dunglas (2013, pages 8–10) describes Doctrine's main concepts Data Mapper and Unit of Work as following:

Data Mapper synchronizes database data with their related object instances. Doctrine's Data Mapper is called Entity Manager and is characterized by:

- adding and modifying database rows held by object properties.
- deleting database rows if object properties are marked to be deleted .
- hydrating in-memory objects which means adding object properties by real data stored in a database.

Unit of Work keeps track of all database operations done during a business transaction and performs operations after the business transaction is done. It is increasing performance and data consistency by not syncing entities with the database every time.

4.2.3 Apigility

To fullfil the need of a modern API builder, Zend Technology developed the open source project Apigility around ZF 2. It also works with ZF 3. It provides different tools to build and document APIs REST endpoints easily. A GUI helps to handle content negotiation, authentication, documentation, versioning and a lot of other services.

APPELL created a REST API to transfer data from the ZF web application to the native mobile apps.

4.3 Swift

Swift is the successor of Objective-C, the standard language to implement native applications for iOS. Published in 2014 from Apple Inc. as open source licensed under the Apache License 2.0, Swift should combine the strengths of C and Objective-C without the constraints of C Compatibility with modern approaches to safety, performance and design patterns. It is intended to be used an all Apple platforms: iOS, macOS, watchOS and tvOS. (O. Swift, 2016; Swift, 2020)

Since Apple provides the standard IDE for Swift called Xcode, which is only running on macOS, Apple tries to force developers to use MacBooks or iMacs. Currently Swift is in the 5th version, but as APPELL was developed, the most recent version was Swift 4.2 with a set deployment target iOS 11.0.

Similar to PHP's package manager composer, Swifts uses Cocoapods[3] as its package and dependency manager.

```
# UI packages
pod 'Tabman'                        # easy tab view handling
pod 'Pageboy'                       # dependency of tabman
pod 'TransitionButton'              # button animations
pod 'CDAlertView'                   # alert modal
pod 'AvatarImageView'               # avatar image view

# Core/Helper packages
pod 'Alamofire'                     # http networking lib
pod 'CryptoSwift'                   # crypto lib
pod 'KeychainAccess'                # easy keychain access

# Analytics tools
pod 'Firebase/Core'                 # analytics tool
pod 'Fabric', '~> 1.9.0'            # analytics tool
pod 'Crashlytics', '~> 3.12.0'      # analytics tool

# Other
pod 'Kingfisher'                    # music file handling lib
```

Code Snippet 4.3: Included cocoapods in Swift for iOS

[3] Visit: https://cocoapods.org/ (accessed 30-May-2020)

TestFlight

To test swift applications, Apple offers and application called **TestFlight**. Developers can invite testers to test their application by sending out TestFlight testing emails. Those mails consist an invitation link. With the help of the TestFlight application and the invitation link, the not yet released application version can be downloaded and tested.

4.3.1 Firebase

Google's Firebase is a software platform that allows to monitor and analyze web or mobile applications. Embedding by adding Firebase to the podfile and configuring within the mobile application, specific data is sent to Firebase from the iOS application.

Firebase is described in detail in section 6.1.

5 Technical Implementation

The following technical description is about the final software product and summarizes the result of the agile software process. The implementation phase started directly after finishing the software concept. Discarded concepts are not considered.

The source code is stored at the TU Graz GitLab repository and can be cloned by authorized persons[1]. The specific projects for iOS, Android and the the web application[2] can be found in the corresponding directories.

To run APPELL's web application some steps and dev tools are needed. As a virtualization platform a Vagrantfile with the needed configurations is prepared. Therefore vagrant[3] has to be installed and the Vagrantfile executed by `$ vagrant up`. Vagrant offers now a *debian Operating System* and installs *apache2*, *php5* and *mysql*.

The database can be build and pre-filled with the SQL dump, which is also available in the repository.

To ensure a common appearance in all the different views of all software components, two colors are used: Blue (color code #6195FF) as the main color and grey (color code #6195FF) as a secondary color. Texts are shown in black or white.

[1] Visit GitLab: https://gitlab.tugraz.at/llt/learninglab_coop/schule/schule__learning_apps.git (accessed 21-September-2020)

[2] In the following also *data management application* will be used as a synonym.

[3] Visit: https://www.vagrantup.com/ (accessed 12-May-2020)

5.1 Environments

APPELL uses two environments: **"dev"** (develop) environment for development and testing purposes and the **"prod"** (productive) environment, which is the currently released version and actually used by the students. Both environments use an own database and an own user-manager key-appId key-pair (see subsection 5.3.2). The dev instance is intended to test or debug functionalities. Since it has own data instances, it has no cross-effects to the productive application. Each environment is accessible with its baseurl.

Indirectly the local environment used by developers with a local database can also count as an additional, third environment.

```
extension RESTManager {
    public static func getBaseUrl(for environment: Environment =
     ↪  RESTManager.actualEnvironment) -> String {
        switch environment {
        case .local:
            return "http://localhost:8080/api"
        case .dev:
            return "https://schule-dev.tugraz.at/Appell/api"
        case .prod:
            return "https://schule.learninglab.tugraz.at/Appell/api"
        }
    }
}
```

Code Snippet 5.1: Swift RESTManager environments

The snippet in 5.1 shows how the environments for the REST API are executed from the mobile applications.

5.2 Database

Database design is very important. The changes in the database in lasts stages are very difficult and tricky, even worse if it is already used in productive systems. Therefore a well thought concept at the beginning is inevitable. Based on the database concept all models in every development environment are derived.

The MySQL database consists of 15 tables and 27 relations which is quite complex for that amount of tables, as the EER diagram in Figure 5.1 illustrates. The database is designed to fit in different projects, which are represented by the project table. Most of the other tables are related to that one by a one-to-many relationship and hold the project reference by their *projectid* column. Around the project can be multiple glossar entries (*translation* and *phrases*), quizzes (*pool*, *type*, *question* and *choices*) and exercises (*pool*, *type*, *exercise* and *gap*). A project typically has two languages (*language*) entries, the native and foreign language. The access to a project is only allowed to administrators, whose id is stored in the *access* table.

A more detailed description of the tables, their aims and relations are given in the next subsection.

5.2.1 Database model

The design of the database can be split into five major parts related to their responsibilities: *Project*, *Glossar*, *Type*, *Quizzes* and *Exercises*.

- **Project**
  Project is responsible for handling multiple projects/class/courses and their functionalities.
  A project has a name, a responsible person with contact information to know who is responsible for the data and to restrict access to

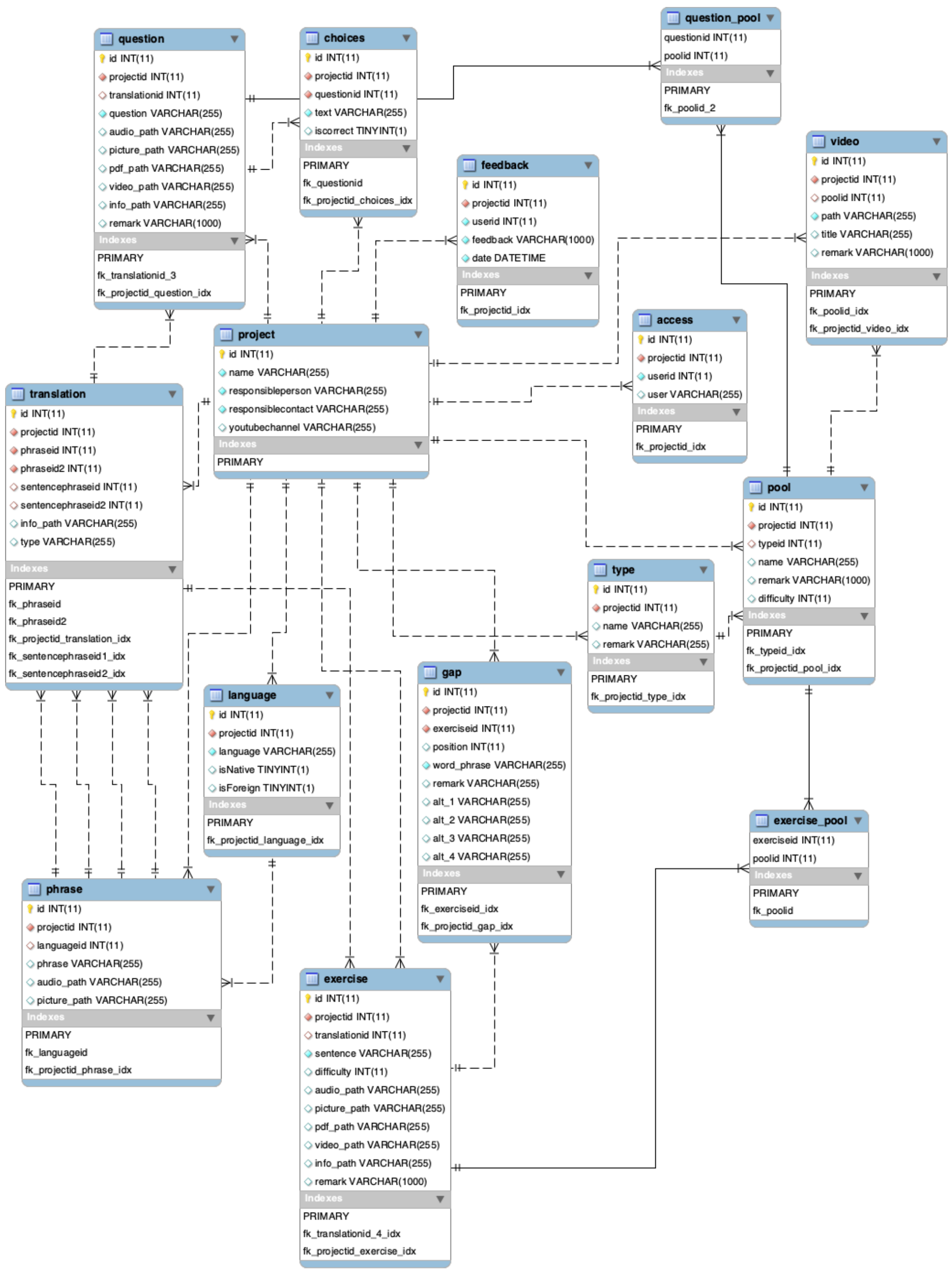


Figure 5.1: EER diagram [Image created by the author of this thesis.]

the CMS. It also has a YouTube channel that stores an URL link to YouTube within the mobile applications. Most of the tables are related to the project table, except for join tables that ensures to build m:n relationships. Tables only related to *project* are *feedback* and *access*. App-users can send feedback which is stored with the user id and the creation timestamp.

- **Glossar**
  The glossar is implemented with the help of the *language*, *phrase*, and *translation* table. Usually a project has a foreign and a native language. This information is stored in *language* table. A single word, phrase or sentence can be stored in the *phrase* table which is then assigned to a project, language and a translation. The paths to link an audio file or picture can be stored in the corresponding columns within the *phrase* table. For a valid translation, a pair of phrases is needed and therefore stored in the *translation* table. Additionally it can also be stored a whole sentence to show how a word or phrase is used in context. A cross-link to online sources is represented by the info path column.
- **Types**
  Types and pools are implemented to structure questions and exercises. A type can have n pools and each pool multiple questions and exercises ensured by *question pool* and *exercise pool* tables. The aim is to structure quizzes and exercises roughly by types and more precisely by pools. Both tables can store remarks, pool additionally stores a difficulty integer.
- **Exercises**
  An exercise holds relations to project and translation. The exercise itself builds the sentence property. Links to an audio file, .pdf file, video and picture can be stored in order to support the exercise. The targeting phrase of the sentence is represented by 1 to n gaps. Four alternative gaps can be used to show a wrong usage. The position attribute of gap is needed to target the right gap. Exercises can be assigned to multiple pools.

- **Questions**
  A question is implemented similarly as an exercise. The only difference is that a question does not store gaps, it stores multiple choices. Choices can be correct or incorrect. Questions can also be assigned to multiple pools to build a quiz.

5.3 Web application

The web application or data management application is the central element of the platform. Written in PHP and based on Zend Framework, it uses the MVC design pattern to ensure typical CMS actions to maintain the database and provides a Representational State Transfer (REST) Application Programming Interface (API) to manage data exchange with the mobile applications. The most important components are described below.

5.3.1 Doctrine & Models

One of the core functionalities of the web application is the database connection. As an Object Relational Mapper (ORM) Doctrine is very popular for PHP projects.
Derived from the database model, PHP classes are designed exactly according to the database table, and created automatically by using Doctrine's `orm:generate-entities` command. Those classes are now mapped according to their database table and represent the data structure and finally holding data. With the help of special PHP comments - called *annotations* - this so-called **entities** are getting described and offers Doctrine information about how to map the classes/entities with the corresponding database tables. The syntax and structure of such annotations are shown in snippet 5.3. Further information are provided at Doctrine's website[4].

[4] Visit: `https://www.doctrine-project.org/` (accessed 24-May-2020)

- *Line 4* maps the class to the database table by its exact name.
- *Line 5* links the class to its repository class.
- *Line 10* gives information about the property's data type in PHP.
- *Line 12* gives information about the property's name, data type and further attributes in the database.
- *Line 13* tells that the property is an identifier.
- *Line 14* ensures that the identifier property is getting auto incremented.

```
1  /**
2  * Project
3  *
4  * @ORM\Table(name="project")
5  * @ORM\Entity(repositoryClass="CMS\Repository\ProjectRepository")
6  */
7  class Project
8  {
9      /**
10     * @var int
11     *
12     * @ORM\Column(name="id", type="integer", nullable=false)
13     * @ORM\Id
14     * @ORM\GeneratedValue(strategy="IDENTITY")
15     */
16     private $id;
17     ...
18 }
```

Code Snippet 5.3: Project entity

Line 5 in 5.3 links an entity class to a Doctrine's repository class. A **repository** provides methods to retrieve data from its entity/database table. By extending from its base class *EntityRepository* customized queries can be provided.

The *EntityManager* is a Doctrine class, composed by `doctrine/doctrine-orm-module` and retrieved in the factory classes as follows: `$entityManager = $container->get('DoctrineORMEntityManager')`.

Method	Description
detach($entity)	Detach an entity from the entity manager that it is no longer managed.
persist($entity)	Adds new entity to the entity manager and manages it.
merge($entity)	Merges new entity into entity manager and manages it.
remove($entity)	Removes an entity from persistent storage.
createQuery($dql)	Creates a new Query object.
getRepository($entityName)	Gets the repository for an entity class.
flush()	Flushes all changes to objects and perform database commands.

Table 5.1: Doctrine EntityManager commands

With the help of this EntityManager, entities from their repositories using different search criterions can be retrieved, manipulated and stored back to the database. Krivstov (2019) summarizes Doctrine's main methods as listed in table 5.1.

A typical Doctrine flow, which writes to the database, is done within every controller as following:

1. get EntityManager
2. initialize representing entity PHP instance
3. initialize representing form PHP instance
4. check data if valid by using the form instance
5. set/update entity instance by valid data
6. persist or merge entity instance
7. flush changes

In the other hand to retrieve data with the EntityManager or the repository:

1. get EntityManager

2. get repository by the entity name
3. retrieve data by using standard EntityManager or repository methods or by customized methods

5.3.2 Authentication

The authentication of APPELL is managed externally. The *Learning Lab* core framework provides a Simple Object Access Protocol (SOAP) web service called *UserManager*. An application has to be registered at the UserManager by an *appId* and an *secret key* for both dev and prod instance as shown in 5.4. Those authenticates the requesting app. Within the web application, authentication is handled by the classes *AuthManager* and *AuthAdapter*. The authentication manager service is responsible for logging in/out a user and to check if a user is allowed to access a page.

The UserManager provides several methods to retrieve user information or to check if a user is registered. Those information can be accessed by SOAP. To check if a user is registered, some credentials has to be passed to the `isUserAllowed()` method according to *line* 2 in 5.3: *username* and the *appId* are self-explanatory, while the *password* is the hashed plain password with the sha256 algorithm `hash('sha256', $plainPassword)` and *$hash* represents the keyed hash value using the Uash-based Message Authentication Code (HMAC) method with the sha256 algorithm, the concatenated username, password hash and app id as the method's data attribute and the registered app key as the secret key. The return object holds different information like if the user is accepted, id, roles, etc. There are two types of users/administrators that get access. Both has to be accepted users (registered users) by the UserManager:

Project admins have a role set as everything else than *'admin'* but are stored in *access* database table. Project admins can only generated by global admins.

Global admins have a role set as *'admin'*.

```
$hash = hash_hmac('sha256',$this->username . $this->password .
  ↪  $this->appId, $this->key)
$result = $this->client->isUserAllowed($this->username, $this->password,
  ↪  $this->appId, $hash);
$result = $this->client->getName(intval($fullResult->idUser));
```

Code Snippet 5.3: UserManager SOAP methods

By calling the `getName()` method a user's name can be requested.

Since pages of the web application are only permitted to global admins an access filter is implemented. Every route except `/login` is restricted. Therefore and *Dispatch* event listener which is triggered after the *Route* event. If the user is not allowed to access a route, this route is stored in the URL, the user gets forwarded to the login page and after a successfully log in, again forwarded to the the originally requested route. After the login, a new session starts. By default a session lasts 60 minutes which can be prolonged to 30 days by selecting the *remember me* checkbox.

5.3.3 Views & Controllers

Almost every model has its own view and its own controller. Views are responsible to produce HTML output for web browsers. ZF uses *.phtml* files, so-called view templates. Those view templates can hold different PHP syntax, such as loops, conditions, variables, etc. In 5.3 for example, the sentence of an exercise is getting inset between a `<h3>` tag. `$this` and `$element` are passed by the controller. The `escapeHTML()` function is provided by Zend to ensure that no malicious code can be injected. The special Zend class *view resolver* resolves the code to produce plain HTML pages.

To fit the time-frame of the master thesis APPELL's web application views are very rudimentals and close to the data itself. Generally there are three kinds of views as shown in figure 5.2:

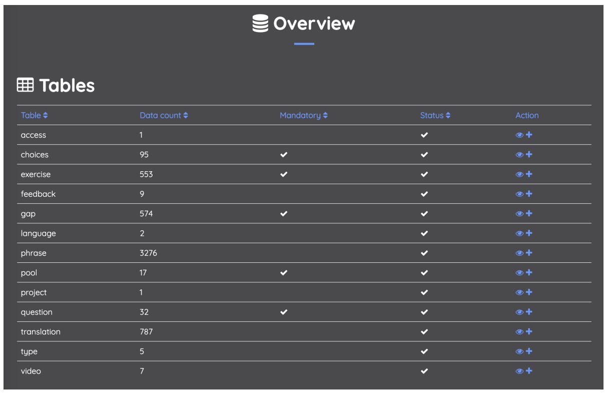
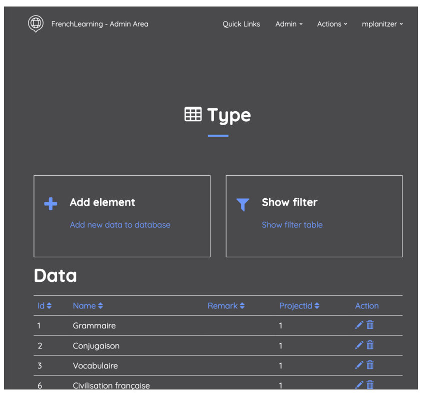


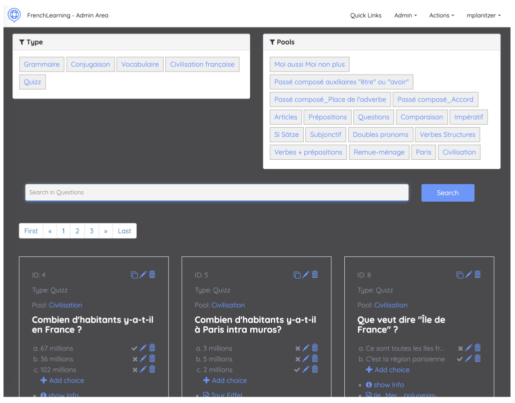
Table	Data count	Mandatory	Status	Action
access	1		✓	👁 +
choices	95	✓	✓	👁 +
exercise	553	✓	✓	👁 +
feedback	9		✓	👁 +
gap	574	✓	✓	👁 +
language	2		✓	👁 +
phrase	3276		✓	👁 +
pool	17	✓	✓	👁 +
project	1		✓	👁 +
question	32	✓	✓	👁 +
translation	787		✓	👁 +
type	5		✓	👁 +
video	7		✓	👁 +

(a) Project overview - table view



Id	Name	Remark	Projectid	Action
1	Grammaire		1	✎ 🗑
2	Conjugaison		1	✎ 🗑
3	Vocabulaire		1	✎ 🗑
6	Civilisation française		1	✎ 🗑

(b) Type - table view with possibility to add and filter elements

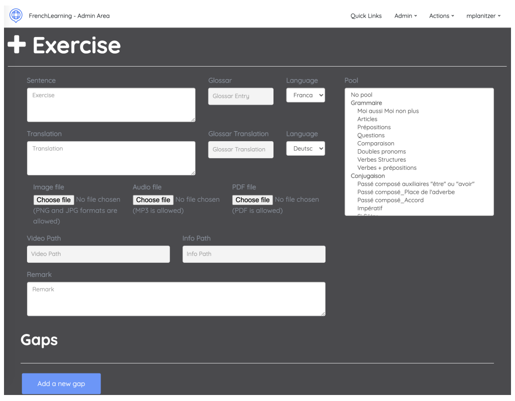


ID: 4
Type: Quizz
Pool: Civilisation
Combien d'habitants y-a-t-il en France ?
a. 67 millions ✓
b. 36 millions ✗
c. 102 millions ✗
Add choice
show Info

ID: 5
Type: Quizz
Pool: Civilisation
Combien d'habitants y-a-t-il à Paris intra muros?
a. 3 millions ✗
b. 5 millions ✗
c. 2 millions ✓
Add choice
Tour Eiffel

ID: 8
Type: Quizz
Pool: Civilisation
Que veut dire "Île de France" ?
a. Ce sont toutes les îles fr... ✗
b. C'est la région parisienne ✓
Add choice
show Info

(c) Questions overview with type and pool filter, search and paginator.



(d) Adding an exercise view

Figure 5.2: Different APPELL web views. (a) shows an overview about the whole platform. (b) shows a typical table view, while (c) shows the card view for exercises and questions and (d) shows a typical form mask to add content in this case exercises and questions. [All images created by the author of the thesis.]

```
<div><h3><?= $this->escapeHtml($element->getSentence()); ?></h3></div>
```

Code Snippet 5.3: phtml syntax

table views to show data very close to their model, the SQL table

card views are more complex views to show exercises and questions with their types, pools, gaps, choices, etc.

form views to add or modify data of different tables

Controllers are responsible for the communication between models and views. As mentioned in 5.3.1 Doctrine & Models, controllers therefore processes input from or for a HTTP request by processing data from or for the database.
Controllers in Zend recognizes actions to be executed by the **Action** suffix. APPELL for example uses `function addEditAction()` to perform add and edit actions. The function is triggered when submitting a form from client (browser) to the server (web application). The transmitted form can then be fetched by `$this->getRequest()` and processed.
`function indexAction()` is used to fetch data and represent that initially on the different views.

5.4 Interfaces

APPELL uses two different Application Programming Interface (API) technologies: Simple Object Access Protocol (SOAP) and Representational State Transfer (REST) APIs.

5.4.1 SOAP

SOAP is used in form of a backend-backend communication to authenticate users as described in subsection 5.3.2. It is hosted externally by the

UserManager and can be consumed according to the code snippet 5.4, provided an application is whitelisted. The UserManager's SOAP API provides two methods (see code snippet 5.3) which can be easily accessed by Zend's `Soap\Client` class.

```
'user_manager' => [
    'dev' => [
        'service' => "http://schule-dev.tugraz.at/usermanager/soap?wsdl",
        'key' => "********************************", // 32char hex key
        'appId' => 37
    ],
    'prod' => [
        'service' => "http://schule.learninglab.tugraz.at/usermanager/soap?wsdl",
        'key' => "********************************", // 32char hex key
        'appId' => 24
    ],
]
```

Code Snippet 5.4: UserManager SOAP connectors

5.4.2 Rest-API

The communication between the mobile applications and the backend (web application) is ensured by a REST API. The API can be accessed according to code snippet 5.1.

APPELL uses Apigility as an API builder provided also by Zend Framework (ZF). Apigility is applied by a GUI, which injects the endpoint settings in form of source code directly to the project. Those injections have then to be connected to Doctrine's repository classes to detach or persist the data. The code snippet 5.4 shows the `/projects` endpoint. If requested, the endpoint sends back all the projects with their languages in form of a JSON object to the requester. Line 8 and 9 assign the needed repositories `Project:class` and `Language::class`. The project repository fetch all projects, in line 11 and line 15 the languages of the projects are appended to the project objet. In line 18 the result is sent back.

```
1  /**
2  * Fetch all or a subset of resources
3  *
4  * @param  array $params
5  * @return ApiProblem|mixed
6  */
7  public function fetchAll($params = [])
8  {
9      $projectRepo = $this->entityManager->getRepository(Project::class);
10     $langRepo = $this->entityManager->getRepository(Language::class);
11     $projects = $projectRepo->findAllProjects();
12
13     foreach ($projects as &$project) {
14         $languages =
    ↪  $langRepo->findAllLanguages($project['project']['id']);
15         $project['project']['languages'] = $languages;
16     }
17
18     return $projects;
19 }
```

Code Snippet 5.4: REST API endpoint /projects

To pass parameters, REST uses four different types:

header parameters are included in the request header.
path parameters are located within the path and appears before query string parameters.
query string parameters are located after the path but still in the URL and delimitated by a '?'.
request body parameters are included in the request body.

APPELL uses three of those types. All the **GET** requests are using path parameter (PP) and query string parameters (QSP) to pass needed parameters to the backend which performs then the database lookups. The **POST** request sends its content as a JSON object in the request body.

There are all together 18 endpoints, 17 **GET** endpoints and one **POST** endpoint:

- **GET** /data/:path
  retrieves the data (image, pdf or mp3) stored in the given path
  PP: path: string
- **GET** /exercises
  retrieves all the exercises from a pool; randomly=true, if the order should be randomly
  QSP: poolid: int, randomly: boolean
- **GET** /exercises/:exerciseid
  retrieves a single question by id
  PP: exerciseid: int
- **GET** /languages
  retrieves all the languages from a project
  QSP: projectid: int, getNative: boolean
- **GET** /languages/:languageid
  retrieves a single language by id
  PP: languageid: int
- **GET** /phrases
  retrieves all the phrases by the given ids; orderSecond=quizzes, if sentence phrases should be given back first
  QSP: phrasesids: [int], orderSecond: boolean
- **GET** /phrases/:phraseid
  retrieves a single phrase by id
  PP: phraseid: int
- **GET** /projects
  retrieves all the projects
- **GET** /projects/:projectid
  retrieves a single project by id
  PP: projectid: int
- **GET** /questions
  retrieves all the questions from a pool; randomly=true, if the order should be randomly

QSP: poolid: int, randomly: boolean

- **GET** /questions/:questionid
  retrieves a single question by id
  PP: questionid: int
- **GET** /translations
  retrieves all the translations from a project; getNative=true, if native language should be the outer element
  QSP: projectid: int, getNative: boolean
- **GET** /translations/:translationid
  retrieves a single translation by id
  PP: translationid: int
- **GET** /types
  retrieves all the types from a project; fetchor=quizzes, if types for quizzes, fetchfor=exercises, if types for exercises
  QSP: projectid: int, fetchfor: enum
- **GET** /types/:typeid
  retrieves a single type by id
  PP: typeid: int
- **GET** /user
  authenticates a user with the hashed password; isDev gives information wether the user is registered in dev or prod environment
  QSP: isDev: boolean, hash: string, user: string
- **POST** /feedback
  post a feedback to a project with the userid
  QSP: projectid: int, userid: int
  body: message: string

5.5 iOS App

The native mobile applications are the most used software components of the whole APPELL software platform. As described in chapter 1 only the iOS app is part of the thesis and can be downloaded in the Apple Store[5].

The app requires iOS 11.0 or later and is universally compatible on both iPhone and iPad. Its age rating is 17+ because of the webview linking to YouTube. In YouTube users younger than 17 years, could watch not allowed videos that consists violence for example. APPELL is written in Swift 4.2 and uses *at.tugraz.appell* as the product bundle identifier.

5.5.1 UI & UX

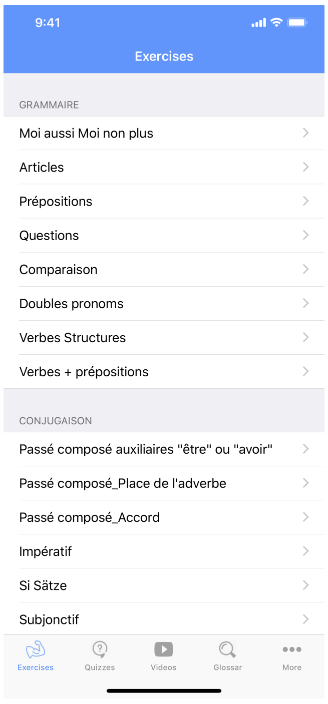
UI and UX has been given a high importance for the mobile applications. For iOS, the app has been developed strictly according to the iOS human interfaces guidelines[6]. Therefore as the main navigation pattern, the flat navigation pattern has been implemented. This so-called tab bar is set in style of a bottom navigation. From every item in the tab bar a hierarchical navigation by pushing views on the navigation controller is then be initialized. The items are structured according to their functionality, starting with *Exercises*, *Quizzes*, *Videos*, *Glossar* and a *More* section (see figure 5.3a).

According to the Customer requirements there are four major user stories and each is representing a bottom navigation item:

- doing exercises
- doing quizzes
- watching leaning videos
- looking up the glossary

[5] Visit: `https://apps.apple.com/at/app/appell/id1449511915?l=en` (accessed 25-May-2020)

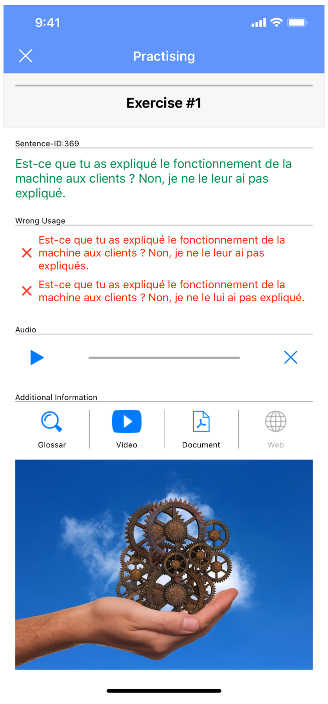
[6] Visit: `https://developer.apple.com/design/human-interface-guidelines/ios/overview/themes/` (accessed 25-May-2020)



GRAMMAIRE
- Moi aussi Moi non plus
- Articles
- Prépositions
- Questions
- Comparaison
- Doubles pronoms
- Verbes Structures
- Verbes + prépositions

CONJUGAISON
- Passé composé auxiliaires "être" ou "avoir"
- Passé composé_Place de l'adverbe
- Passé composé_Accord
- Impératif
- Si Sätze
- Subjonctif

(a) Exercises overview



Exercise #1

Sentence-ID:369

Est-ce que tu as expliqué le fonctionnement de la machine aux clients ? Non, je ne le leur ai pas expliqué.

Wrong Usage

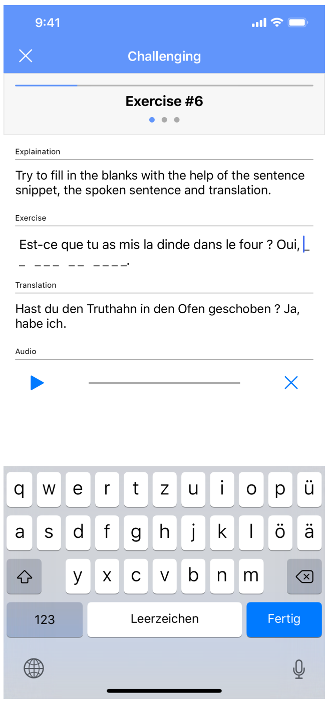
Est-ce que tu as expliqué le fonctionnement de la machine aux clients ? Non, je ne le leur ai pas expliqués.

Est-ce que tu as expliqué le fonctionnement de la machine aux clients ? Non, je ne le lui ai pas expliqué.

Audio

Additional Information

(b) Practising view



Exercise #6

Explaination

Try to fill in the blanks with the help of the sentence snippet, the spoken sentence and translation.

Exercise

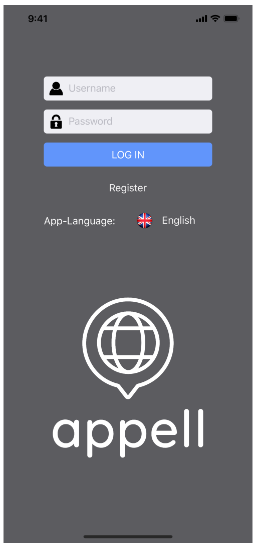
Est-ce que tu as mis la dinde dans le four ? Oui, _ ___ __ ____.

Translation

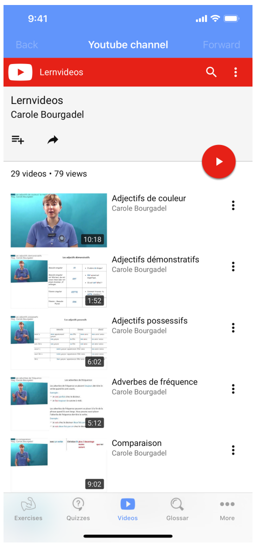
Hast du den Truthahn in den Ofen geschoben ? Ja, habe ich.

Audio

(c) Cloze text view



(d) Login view

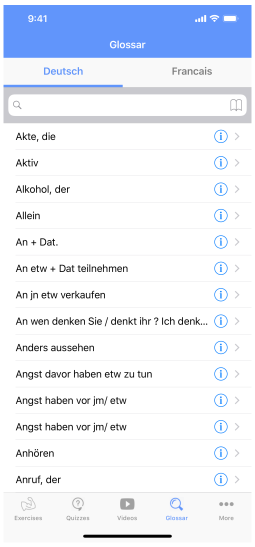


Lernvideos
Carole Bourgadel

29 videos • 79 views

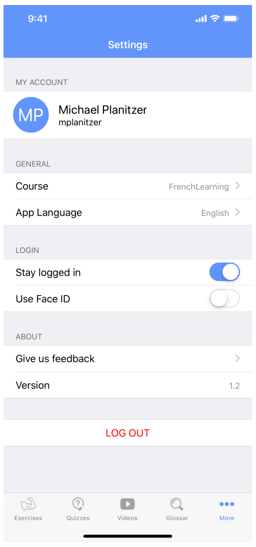
Adjectifs de couleur Carole Bourgadel	10:18
Adjectifs démonstratifs Carole Bourgadel	1:52
Adjectifs possessifs Carole Bourgadel	6:02
Adverbes de fréquence Carole Bourgadel	5:12
Comparaison Carole Bourgadel	9:02

(e) Video channel view



- Akte, die
- Aktiv
- Alkohol, der
- Allein
- An + Dat.
- An etw + Dat teilnehmen
- An jn etw verkaufen
- An wen denken Sie / denkt ihr ? Ich denk...
- Anders aussehen
- Angst davor haben etw zu tun
- Angst haben vor jm/ etw
- Angst haben vor jm/ etw
- Anhören
- Anruf, der

(f) Glossary views



MY ACCOUNT

MP Michael Planitzer
mplanitzer

GENERAL

Course	FrenchLearning
App Language	English

LOGIN

Stay logged in	✓
Use Face ID	

ABOUT

Give us feedback	
Version	1.2

LOG OUT

(g) More tab view

Figure 5.3: Different APPELL iOS app views.
[All images created by the author of the thesis.]

Exercises

If students want to practice exercises, they have to choose a pool of exercises, for example *Comparaison, Questions, etc.*. Pools of exercises are structured into topics (types) like *Grammaire, Conjugaison, etc.* (see figure 5.3a). After choosing a pool an overview about those pool is shown with different information, such are: the number of containing exercises, the last attempt for practice and challenge mode and the best attempt for practice and challenge mode.

The difference between practice mode and challenge mode is that practice mode shows all the available information of an exercise in one view (figure 5.3b). Right and wrong usages are shown, an audio can be listened to get the right pronunciation. A link to the glossary entry is given. If available video links, a pdf document and an external link to a website can be used. An image for visual learning types is displayed at the end.

In challenge mode students have to solve cloze text problems and MC problems. For each exercise there are two cloze text problems and one MC problem to solve:

1. At first students have to fill in a gap in a sentence. The student is getting the translation and the spoken sentence to listen (figure 5.3c).
2. Secondly students have to to choose the right sentence out of several.
3. Lastly students have to fill out the whole sentence, which means all the sentence is in cloze style.

In every step students can get to the practice view with all the information for the exercise. Also after each step students have to validate their answers to get to the next step. After finishing all three problems, the next exercise with the three sub-exercises has to be solved.

Quizzes

Quizzes are similar to the exercises. A pool of questions can to be chosen. Pools are structured into topics (types). After selecting a questions pool, students can answer them in form of SC or MC.

Videos

The videos section is a webview linking to a YouTube channel (figure 5.3e). All the learning videos are hosted on YouTube. The URL of the used video platform can be set in project table via the web application. The webview can be navigated forward and backward in the native navbar.

Glossary

The glossary is mainly a collection of all the words, phrases and sentences used for the exercises and questions. Also there can be added further, independent words and phrases. The glossary can be applied to both languages, the origin (native) language and the target (foreign) language. Sorted alphabetically, glossary entries can be easily looked up with the search bar at the top (figure 5.3f). Once an entry is chosen, it shows the word or phrase itself, the translation, an example sentence and the translation of the example sentence. If existing an audio track with the spoken sentence can be listened and a picture is shown.

Login & Registration

To use APPELL students have to be registered in the UserManager as described above. The entrance screen after downloading the mobile app is the login screen. Students can open a webview to register directly at the

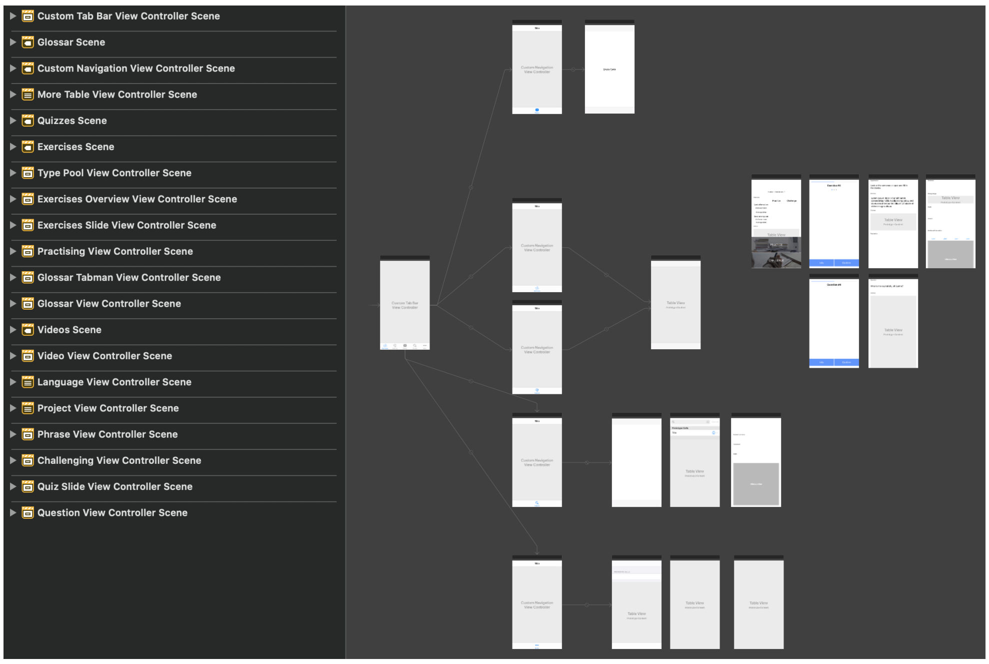

Custom Tab Bar View Controller Scene
Glossar Scene
Custom Navigation View Controller Scene
More Table View Controller Scene
Quizzes Scene
Exercises Scene
Type Pool View Controller Scene
Exercises Overview View Controller Scene
Exercises Slide View Controller Scene
Practising View Controller Scene
Glossar Tabman View Controller Scene
Glossar View Controller Scene
Videos Scene
Video View Controller Scene
Language View Controller Scene
Project View Controller Scene
Phrase View Controller Scene
Challenging View Controller Scene
Quiz Slide View Controller Scene
Question View Controller Scene

Figure 5.4: Main storyboard [Image created by the author of this thesis.]

UserManager web page, if already registered, users can log in to APPELL by entering the username and password (figure 5.3d).

More

The fifth element of the bottom navigation is the more section (figure 5.3g). In this section users find their credentials, they can select the course they want to exercise for, set up the app language (so far german, english and french is supported), can choose if they want to stag logged in or to use pin, touch or face id. Also they can give feedback which administrators can read in the web application, see the current app version or log out again.

5.5.2 Technical structure

The iOS project is implemented with Xcode and uses different file types:

plist files contain configuration settings and properties to run the application.
strings files contain key-value translations.
xib files are xml files containing UI settings and properties.
storyboard files are also xml files storing screens and screen interactions.
Podfile is a specification file that saves the used libraries.
swift files are the source code files written in Swift.

Those files are structured in model, view, cells and view-controller directories and into supporting directories, such as storage, helper, api and global directories. Models, view, cells and view-controller are overtaking the classical MVC pattern. The directories are named according to their responsibilities and therefore named after the database tables.

xib files and storyboards are saving user interfaces which can be styled by the interface builder of Xcode. While a xib file is representing a single view element, for example an audio player or a table cell, a storyboard applies different scenes which are consisting of multiple xib views and standard UI elements and handles transitions between those. APPELL uses the two standard storyboards, the storyboard for the launching screen and the main storyboard. The scenes shown in Figure 5.4 are representing more or less the whole application. Origin is the flat navigation on the left most scene, which leads to the five sections described in subsection 5.5.1.

Each scene of the storyboard has its own view-controller, where all the UI-elements are accessible. View-controllers have mainly four responsibilities (Apple, 2020):

- Updating view content
- Responding user interactions
- Managing layout and views of the interface

```swift
RESTManager.shared.getTranslations(language1.isNative) { result in
    switch result {
        case .success(let data):
            self.translations = data.translations
            self.glossarTableView.reloadData()
            self.state = .finished(self.translations)
            Logger.info("Successully fetched translations")
        case .failure(let error):
            Logger.error("Could not get translations")
            self.state = .failure(error)
    }
}
```

Code Snippet 5.5: Swift RESTManager

- Interacting with other view-controllers

To be able to build views, data have to be fetched from the server and parsed to the model structure. The `RESTManager` singleton class is responsible for every API request. It manages the environment handling, session handling, checks if internet and the API is reachable, offers methods for the different endpoints and deserialize and parses the JSON responses. After data is fetched, the delegates and data sources of the different view-controllers can be set. The code snippet in 5.5 for example fetches translations for the glossary. The `getTranslations` method parses the REST response to the translation model. On success the translations are set in the local scope and the the table-view with the data source and delegate reloaded.

5.6 Firebase

APPELL uses Firebase for two reasons: analytics and crash reporting. The standard features of Firebase are free and can be used after creating a project at its web application[7]. By adding users via the Firebase console, further administrators and developers can access Firebase.

```
private func logEvent(_ event: String) {
    guard let index = self.currentIndex, let exerciseMode = mode else {
     ↪  return }
    let totalEvent = exerciseMode == .practice ? "practice_\(event)" :
     ↪  "challenge_\(event)"
    let exercise = exercises[index / ExerciseModes.allCases.count]
    Analytics.logEvent(totalEvent, parameters: [
        "courseid": pool.projectid as NSObject,
        "type": poolType.name as NSObject,
        "typeid": poolType.id as NSObject,
        "pool": pool.name as NSObject,
        "poolid": pool.id as NSObject,
        "exercise": exercise.sentence as NSObject,
        "exerciseid": exercise.id as NSObject
    ])
}
```

Code Snippet 5.6: Logging Firebase event in swift

To connect an iOS application, the app has to be registered at Firebase by its bundle-id. Firebase provides then a .plist file, which has to be imported to the iOS project. By including and installing the Firebase SDK `pod 'Firebase/Analytics'` everything is ready. A simple `FirebaseApp.configure()` in the AppDelegate starts the SDK and communicates with the Firebase server whenever the application is started on a client device. Crash reports are automatically send out if there are unexpected app shutdowns. So far, there are no crashes registered. For analytics APPELL uses the standard functionalities to get information about daily active uses , visit durations, screen visits, audience information, etc. Subfigure 6 6.1a for example, shows the active users from September 2019 until September 2020, with a peak during the Austrian corona lock-down in April 2019.

[7] Visit: https://console.firebase.google.com/project/tug-appell/ (accessed 24-July-2020)

5.7 Updating software components

Updating third party components and own components to the most recent versions is very important to ensure their functionalities and should be done regularly. If there are major releases especially of the underlying core technologies, certain code lines might have to be adapted to the new version. In this section hints and update descriptions are shown.

The web applications can be updated by using the composer command `composer update`. Due to requirements the web application is running with PHP v5, which should be updated to PHP v7 if possible.

The pod dependencies in the iOS application can be updated by using the pod command `pod update`. The swift versions and the deployment target should also be updated if there is a new version of Xcode available. Non-maintained iOS applications are getting banned from the App store and therefore not downloadable anymore.

The import of the Firebase SDK may also become updated, which implies that the import in the mobile application has to be adapted. In this case Firebase would show a hint in its console.

Potential further software parts which can be updated and have effects on the running system are the external *UserManager*, the operating system and the web server which execute the web application, the MySQL database, and the links to the static files.

6 Applied and potential Learning Analytics methods

According to the theoretical aspects explained in chapter 2, some methods to generate Learning Analytics data were implemented. At specific points in the mobile application, data is forwarded to feed analytics databases. Due to the ongoing increasing workload during the sprints, the responsible technical persons of the project have been decided to use Firebase as the only LA instrument. Instead of sending LA data to an own database and of implementing own analytic charts, tables and other visualizations, individual Firebase events were introduced, captured, transmitted and measured.

6.1 Firebase Analytics

Firebase is a product of Google that offers different free and proprietary products for both, mobile and web applications. It helps to build better apps, to improve their quality and to grow software components.

Some notable products and features of firebase are *Firebase ML (Machine Learning)*, *Cloud storages and databases*, *Crashlytics*, *Google Analytics*, *Cloud Messaging FCM* or *Dynamic Links*.

APPELL uses two components, Crahslytics and Google Analytics. Both can be added by including their Software Development Kit (SDK).

Crashlytics

Crashlytics is a real-time crash report system. If crashes occur on client software parts, different crash-based information will be transmitted to Crahslytics and processed. All the information can be consumed on the Crahslytics dashboard, accessible in the Firebase console on its website.

Google Analytics

Google Analytics captures different events and user properties automatically and send them to Firebase. Similar to the crash reports, the analytic data can be accessed within the Firebase console. Typically standard information provided by Google Analytics are:

- Active users per time period
- Top screens and visit duration
- Audience information like location, devices, demographics and interests
- How many users are using which app version
- Retention cohorts

6.1.1 Individual measurements

Beside of that standard information, events and properties, developers can also capture and measure things only concerning the aim of the application. APPELL uses this feature to send LA data to Firebase for further usage. Subfigure 6 6.1c shows all the events captured and their occurrences.

challenge_exited is triggered when a challenge is exited before the last exercise has been done.

challenge_get_info is triggered when an user is requesting information in challenge mode.
challenge_started is triggered when a challenge starts.
practice_exited is triggered when the practice mode is exited before the last exercise has been done.
practice_started is triggered when a practice starts.
quiz_exited is triggered when a quiz is exited before the last question.
quiz_started is triggered when a quiz starts.
view_glossary_entry is triggered when a glossary entry is read.

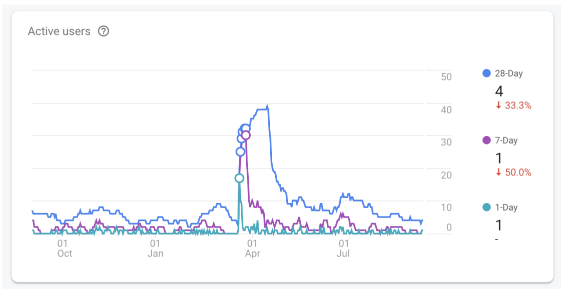
All the other listed events are automatically collected events by Firebase and are described on Google's support website[1]. The function `logEvent` in code snippet 5.6 is responsible for capturing practice and challenge events. Additionally there are further parameters like course, type, pool or the exercise transmitted as displayed in Subfigure 6 6.1d. Those could later be used for detailed analytics, if the data source is connected to Firebase's BigQuery project (see subsection 6.1.2).

The information derived from the individual properties might answer following educational questions:

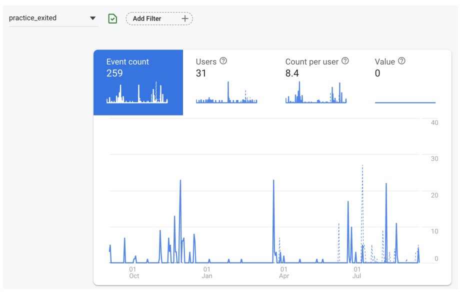
- How long do students need for which exercises?
- For which exercises do they need further information?
- When and on which exercise do they start and stop studying?
- Which glossary entries are how often and how long looked up?

If the individual properties are evaluated together with the standard properties, the previously listed questions can be extended by information of gender, used device, location, etc.

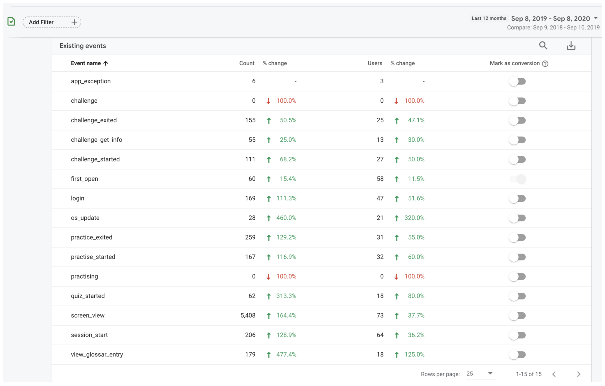
[1] Visit: `https://support.google.com/firebase/answer/6317485?hl=en` (accessed 04-September-2020)



(a) Active iOS users September 2019 - September 2020



(b) Event 'practice_exited' September 2019 - September 2020



Add Filter

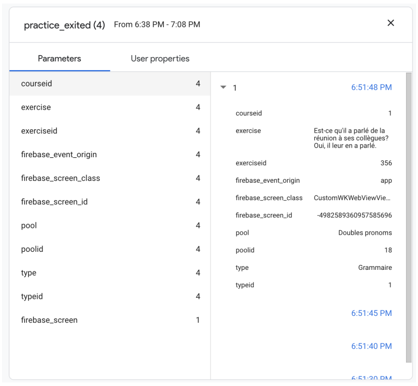
Last 12 months Sep 8, 2019 - Sep 8, 2020
Compare: Sep 9, 2018 - Sep 10, 2019

Existing events

Event name ↑	Count	% change	Users	% change	Mark as conversion
app_exception	6	-	3	-	
challenge	0	↓ 100.0%	0	↓ 100.0%	
challenge_exited	155	↑ 50.5%	25	↑ 47.1%	
challenge_get_info	55	↑ 25.0%	13	↑ 30.0%	
challenge_started	111	↑ 68.2%	27	↑ 50.0%	
first_open	60	↑ 15.4%	58	↑ 11.5%	
login	169	↑ 111.3%	47	↑ 51.6%	
os_update	28	↑ 460.0%	21	↑ 320.0%	
practice_exited	259	↑ 129.2%	31	↑ 55.0%	
practise_started	167	↑ 116.9%	32	↑ 60.0%	
practising	0	↓ 100.0%	0	↓ 100.0%	
quiz_started	62	↑ 313.3%	18	↑ 80.0%	
screen_view	5,408	↑ 164.4%	73	↑ 37.7%	
session_start	206	↑ 128.9%	64	↑ 36.2%	
view_glossar_entry	179	↑ 477.4%	18	↑ 125.0%	

Rows per page: 25 1-15 of 15

(c) App events linked to firebase



practice_exited (4) From 6:38 PM - 7:08 PM

Parameters User properties

courseid	4
exercise	4
exerciseid	4
firebase_event_origin	4
firebase_screen_class	4
firebase_screen_id	4
pool	4
poolid	4
type	4
typeid	4
firebase_screen	1

1 6:51:48 PM

courseid	1
exercise	Est-ce qu'il a parlé de la réunion à ses collègues? Oui, il leur en a parlé.
exerciseid	356
firebase_event_origin	app
firebase_screen_class	CustomWKWebViewVie...
firebase_screen_id	-4982589360957585696
pool	Doubles pronoms
poolid	18
type	Grammaire
typeid	1

6:51:45 PM

6:51:40 PM

(d) Detailed event 'practice_exited' view

Figure 6.1: Firebase offers a lot of different statistics. While (a) shows the active users within a year, (c) is listing up all the events which are transferred to firebase if triggered in the iOS application. (b) shows how often the event 'practice_exited' was triggered and (d) offers a detail evaluation of a single event. [All images created by the author of the thesis.]

6.1.2 BigQuery

As Firebase also BigQuery is a Software as a Service (SaaS) product of Google[2]. It is a fully-managed, serverless, scalable multi-cloud data warehouse designed for business purposes. BigQuery can be easily linked to Firebase to access the raw event data with all the parameters and user properties. The data can be queried in standard SQL and used for every kind of analytics purpose.

Due to the fact that the responsible persons from *Treffpunkt Sprachen* are currently not interested in deeper analytic reports at the moment and also they do not want to store such data permanently, the custom parameters can only used in audience filters and in the stream-view as shown in the sub-figures within Figure 6.1.

6.2 Potential usage of Learning Analytics

The possibilities to utilize learning analytics data are various. As described in the theoretical section What-Who-Why-How approach, there are different stakeholders gaining form learning analytics. In case of APPELL two stakeholders, the teacher and the student might profit from applied learning analytics methods. Following learning applications are imaginable:

Visualization To visualize not only the standard Firebase data, but also the individual measurements might show hidden patterns. Visualizations are important tools and starting point for interpreting data and to identify relationships between data sets. Projected to APPELL there might be hidden patterns between B1 and B2 classes, between Austrians and Non-Austrians or other properties.
Visualizations can also provided to the students to encourage self-reflection.

[2] Visit: `https://cloud.google.com/bigquery` (accessed 24-September-2020)

Monitoring The teacher can get information whether a student is performing differently compared to a relevant group. The teacher can then intervene to support under-performing students individually and more precisely. Over-performing students might be rewarded.
Results of monitoring can be part of the grade.

Recommender system A recommender system is predicting user preferences. Therefore past user's behaviors and behaviors of other users are analyzed and used to recommend exercises that may be interesting for a set of students. Tailored learning paths across the exercise types and questions can be provided.
Exercises that seems to be more difficult to a majority of students might become part of the final exam.

7 Evaluation

Evaluation tries to figure out the worth or metric of an object. It determines whether objectives have been achieved. (Stufflebeam and Coryn, 2014, page 6–7) This chapter evaluates different aspects of the developed software platform and its development process. There are many different types of evaluations used in every stage of the project. All of them can belong to one of the following categories: *formative* and *summative* evaluations. While formative evaluations are conducted proactive during the planning and developing phase. Summative evaluations are performed after the development in retrospective. Formative evaluations provide feedback for improvements and delivers information about how to increase quality. They are usually informally, periodically and interactively performed with everyone involved in the project. Summative evaluations offer information about an evaluand's value and provide an overall judgement. They are especially including stakeholder which are involved in the project as consumers and sponsors. (Stufflebeam and Coryn, 2014, page 23)

In the following sections results of formative and summative evaluations are presented. Those evaluations are periodically organized in pre-release and post-release evaluations because the transition from formative to summative in reality is very fluently and not always unambiguously.

The implementation of APPELL's software parts covered by this thesis has been finished in February 2019. The iOS application is available in store since April 2019. Since the Android application was first available in February 2020, the platform is used by students since the Summer Semester 2020.

7.1 Pre-release Evaluation

The need for developing an own language learning platform has been evaluated by Carole Bourgadel:

> *" Nowadays there is still no language learning platform available which fulfills the need to support students during language learning classes for the levels B1 and B2 in terms of lexical, grammatical and snytactical categories. "*
>
> [Carole Bourgadel - *treffpunkt sprachen - Centre for Language, Plurilingualism and Didactics*]

Based on this research Carole Bourgadel also defined the requirements that are needed for such platform. Those requirements are listed in the Customer requirements section.

During the software planning and developing phase the whole process was evaluated regularly. Before the software was released the outcome was evaluated as well. Those types of formative and summative evaluations are in the following measured by typical software quality criterions which are defined in different standards. The British Standard ISO/IEC 25010:2011, former ISO 9126. ISO/IEC (2013) for example, lists up eight characteristics:

- Functional suitability
- Performance efficiency
- Compatibility
- Usability
- Reliability
- Security
- Maintainability
- Portability

The time schedule for the evaluation of those characteristics, depends on the software development process and its philosophies. In traditional sequential models like the waterfall model, software is evaluated after the implementation phase. APPELL was developed using an agile approach. Functionality evaluations has been done many times, comparing the sprint results with the customer requirements. The technical characteristics compatibility, security and portability were precisely defined in the technical requirements by the appropriate persons mentioned in section 3.1 and supervised especially in the initiate software concept phase.

Reliability for the iOS application is ensured by Apple as long future iOS versions supports the environment target. Any necessary updates have to be overtaken by developers of the staff unit *Lehr- und Lerntechnologien*. The same applies to the ZF web application: if components get deprecated, someone has to update them.

Performance and maintainability were evaluated three times: after the database modelling, after finishing the web application and at the end, before the initial release in spring 2019. Those evaluations have been done in form of a detailed code review and beta testing by responsible persons of the staff unit.

The final usability evaluation was done together with some responsibilities of the *Treffpunkt Sprachen* and selected students in winter 2018/2019 in form of a thinking aloud test. Based on that, some minor changes in the mobile application has been done. After fulfilling the MVP of the data management application, this single part already got deployed on the live server to be able to fill the database with productive data. During this data filling process a lot of feedback was treated and used to improve the usability of the web application.

After all the characteristics were evaluated and approved, the whole software code was merged into the master repository, the Domain Name System (DNS) settings were set and the iOS application released in the Apple App Store.

7.2 Post-release Evaluation

More than one year has passed since the release of the software components and involvement on the language classes. The first time the platform was used by students was in Summer Semester 2020. In September 2020 the whole platform was evaluated retrospectively in form of an expert interview with the initiator of the whole project, Carole Bourgadel and some interviews with students who were using the application. Those summative evaluations aim to discover the impact of the project and its economic efficiency.

The interview with Carole Bourgadel reflected the whole software development process and especially the first experience with the operative usage in classes in Summer Semester 2020. In the Summer Semester 2020, altogether, 90 students worked with the application.

The most significant statements are structured into technical, organizational and educational factors. Technically the platform runs as intended. All features defined in the requirements has been implemented and are working smoothly. The mobile application is easy to use and has never crashed so far. The most problematical aspect is to fill the platform with data. It is a lot of work to collect all the multimedia files. Adequate images have to be found and checked if they are licensed openly, audio and video files have to be recorded and cut. At the end everything has to be uploaded and added to the platform. It took about 180 hours to organize and upload more that 350 exercises with images and audio records, 20 descriptions in form of PDF documents, and also 25 videos. Especially this amount of work discouraged colleagues of setting up the platform for their language classes. This time-consuming setup process decreases the cost-benefit relationship of the platform. It seems, it needs very motivated teaching staff spending their leisure time to get applied more widely. Also the fear of technology and the lack of experience using technology in teaching hinders participation. Apart from that, the colleagues and superiors feedback were favorable, and regarding to the ongoing digitalization, outstanding. Those feedback were given during a keynote at *Arqus*[1] and an internal presentation of APPELL.

The major impact of APPELL can be observed in teaching. As consequence of the introduction of the platform, other online tools and applications like *Moodle* are getting used more and more. The reason for this is that the digital content is created anyway. Sharing those in multiple channels is the logical consequence. In classes APPELL is getting introduced in the first lessons with a small live demo. Cross-references with exercises in the app and the current teaching content are pointed out continuously. Not least because of integrating exercises to the final exams makes it indispensable to use the application proactive for students. The educational effects are also topic for a research paper in progress, written by Carole Bourgadel with the title: *"Interlingual Interference Phenomena in French Courses"*[2].

Students have been interviewed for both classes, French B1 and French B2, to get some user feedback and insights about students' experience. In general the application was very welcomed and there was also a kind of *'wow effect'* - they have never seen classes supported by mobile applications. The structure of the application is very intuitive and the basics elements easy to use. The general design might be improved to point out a playful way of studying. To study independent of place and time, the application offers the possibility to do exercises occasionally. The greatest features from the perspective of the students are the multimedia contents: the videos and the audio records help to improve listening comprehension perfectly. In this way all learning types are covered. The mixture of phrases, grammar, descriptions and exercises are forming a contrast to other available language learning applications. Although those positive aspects, the students think that really big improvements cannot be achieved by using the application

[1] Visit: `https://european-university.uni-graz.at/de/neuigkeiten/detail/article/grenzenlos/` (accessed 26-September-2020)

[2] Visit: `https://treffpunktsprachen.uni-graz.at/en/research/didactics/projekte0/aktuelleprojekte/interlingual-interference-phenomena-in-french-courses/` (accessed 26-September-2020)

only but as a complement. Also the exercises are too reproductive, which means that the exercises repeat constantly.

8 Summary and Outlook

In this final chapter, important aspects, decisions, results and technical key facts addressed in the previous chapters are summarized. The thesis concludes by indicating possible further developments, potential improvements and a general outlook.

Learning Analytics and Educational Data Mining (EDM) hold a great potential for language learning. This thesis introduces Learning Analytics (LA) methods into a Computer Assisted Language Learning (CALL) platform. Resulting from the need to develop a self-organizable language learning platform by a language teaching institute, the platform APPELL has been designed and developed from scratch.

APPELL's main aim is to support students during their language courses, especially in terms of lexical, grammatical and syntactical categories, by providing a mobile application. The platform is embedded into the TU Graz *Learning Lab* environment and thus it employs its user management system called *UserManager*. Due to this pre-requirement, most of the applied technologies are derived from technologies already used by the Learning Lab environment. APPELL's architecture is designed distributed, which means there are more than one independent and viable software components that communicates among each other via interfaces. The data management is implemented as a protected CMS web application. It executes database requests to the platform-own MySQL database, providing the platform-own web service, connecting to third party interfaces and performing classical MVC processes. The mobile applications used by the students are implemented specifically and natively for iOS and Android.

Mobile applications and data management application communicate with the help of the REST API.
This structure forms the core framework of the distributed software platform and fulfills all the functionalities and tools documented in the Minimum Viable Product (MVP) requirements.

Based on that core framework, suitable LA data is captured and sent to the extern platform *Firebase*, which is embedded in the mobile applications. Firebase combines standard analysis like screen durations or audience information with individual measures - in this case learning analytics measures. Firebase is easy to set up and sent to its servers in real-time. For example, individual captured LA-based data provides information about the duration of spending on the element or on which exercise they are stopping to study. In the Firebase console the information can be retrieved and different evaluations done. Connected to Firebase's *BigQuery* project, the data can be stored permanently and typical database queries are possible.

A typical story sequence of applying APPELL might be following:

- An *administrator* logs in into the web application to administrate his 'French' class by selecting the class and filling in his credentials. The CMS authenticates the user by requesting the external *UserManager*. Freshly logged in, the administrator can perform different CRUD actions like adding a new grammar type and assigning new exercises to this type.
- A student who is attending to the real-world 'French' class can now download the mobile application, register and logging in to the platform. The student can apply to his 'French' within the application. The recently added grammar type and exercises are immediately available in the mobile application. All the data transfer is ensured and run by the RESTful web service.
- An Firebase administrator visits the Firebase console and sees different standard analytics of the application. If data is forwarded to BigQuery extensive queries and evaluations are possible.

The data sent to Firebase is only available in a reduced manner. Since the *'customer'* of the platform has never been interested in detailed analytics, the data has never been connected to advanced databases either. This implies that no reasonable evaluation and Learning Analytics are feasible - at least as the thesis was written. Should this situation change, third party databases can directly connected with Firebase. Also there is the opportunity to save LA data in the platform-own database and to prepare, analyze and display reports in the web application - changes in the mobile applications, the database and the data management application would be necessary. The possibilities for capturing analytics data and to utilize that data are diverse. The results of those analysis may serve to improve and extend features or procedures of the software itself.

Beside extending learning analytics tools and methods, UI/UX elements can always be improved. Especially the web appearance for the administrators can be enhanced. In this stage data is displayed very close to its relational nature. Usability tests may be an appropriate way to receive user feedback and in this way use the information to implement more intuitive user stories to Create, Read, Update and Delete data.

Sooner or later the file storage will also become a limiting factor. Due to the increasing usage of the platform, all the multimedia files stored directly into the web applications directory on the server will cause a memory issue. In this case the files have to be moved to an external file storage like Amazon's S3 storage and the paths to those files have to be updated in the database too.

The developed language learning platform builds a solid framework. If updated frequently and if some of the mentioned improvements are getting implemented, it might get an integral part of supporting students during their language classes. Extended LA methods might help to generate valuable information about the students learning behavior and learning progress, which again can be used to adapt learning contents and methods in the physical class and also to improve the software platform itself.

Bibliography

1st International Conference on Learning Analytics and Knowledge 2011 (2010). *About the Conference*. [Online; accessed 20-April-2020]. URL: https://tekri.athabascau.ca/analytics/ (cit. on p. 5).

Anthony, R. (2015). *Systems Programming: Designing and Developing Distributed Applications*. [Online; accessed 10-June-2020]. Elsevier Science. ISBN: 9780128008171. URL: https://books.google.at/books?id=BuCcBAAAQBAJ (cit. on p. 22).

Apple (2020). *UIViewController*. [Online; accessed 03-August-2020]. URL: https://developer.apple.com/documentation/uikit/uiviewcontroller (cit. on p. 55).

Aurum, A. and C. Wohlin (2006). *Engineering and Managing Software Requirements*. [Online; accessed 10-June-2020]. Springer Berlin Heidelberg. ISBN: 9783540282440. URL: https://books.google.nl/books?id=JRWrGLuWpLsC (cit. on pp. 16, 17).

Bharati, M. and Bharati Ramageri (2010). "Data mining techniques and applications." In: *Indian Journal of Computer Science and Engineering* 1 (cit. on p. 9).

Brysbaert, Marc, Pawel Mandera, and Emmanuel Keuleers (2017). "Corpus linguistics." In: (cit. on p. 12).

Chatti, Mohamed Amine et al. (2012). "A Reference Model for Learning Analytics." In: *Int. J. Technol. Enhanc. Learn.* 4.5/6. [Online; accessed 04-May-2020], pp. 318–331. ISSN: 1753-5255. DOI: 10.1504/IJTEL.2012.051815 (cit. on pp. 6, 9, 10).

Doctrine (2020). *About Doctrine*. [Online; accessed 03-May-2020]. URL: https://www.doctrine-project.org/ (cit. on p. 30).

Dunglas, K. (2013). *Persistence in PHP with the Doctrine ORM*. Community experience distilled. [Online; accessed 11-June-2020]. Packt Publishing. ISBN: 9781782164111. URL: https://books.google.at/books?id=0NtdAgAAQBAJ (cit. on p. 30).

Educational Technology (2020). *Educational Technology Website*. [Online; accessed 05-May-2020]. URL: https://www.tugraz.at/en/tu-graz/organisational-structure/service-departments-and-staff-units/educational-technology/ (cit. on p. 2).

Eggert, Ralf (2017). *Zend Framework 3 : das umfassende Handbuch*. Bonn: Rheinwerk Verlag GmbH. ISBN: 978-3-8362-3965-3 (cit. on p. 27).

ElAtia, S., D. Ipperciel, and O.R. Zaiane (2016). *Data Mining and Learning Analytics: Applications in Educational Research*. Wiley Series on Methods and Applications in Data Mining. [Online; accessed 07-June-2020]. Wiley. ISBN: 9781118998212. URL: https://books.google.at/books?id=01AeDQAAQBAJ (cit. on p. 12).

Entrikin, Russell (2012). "Applying Corpus Linguistics Techniques to Learning Analytics." In: (cit. on p. 12).

ISO/IEC (2013). "ISO / IEC 25010 : 2011 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models." In: (cit. on p. 66).

Klašnja-Milićević, Aleksandra, Mirjana Ivanović, and Bela Stantić (2020). "Designing Personalized Learning Environments — The Role of Learning Analytics." In: *Vietnam Journal of Computer Science* 07.03. [Online; accessed 12-May-2020], pp. 231–250. DOI: 10.1142/S219688882050013X (cit. on pp. 7, 8).

Krivstov, Oleg (2019). *Using Zend Framework 3*. [Online; accessed 05-June-2020]. URL: https://olegkrivtsov.github.io/using-zend-framework-3-book/html/en/Database_Management_with_Doctrine_ORM/About_Entity_Manager.html (cit. on p. 41).

Learning Lab (2020). *Learning Lab Website*. [Online; accessed 06-May-2020]. URL: https://learninglab.tugraz.at/ (cit. on p. 2).

Li, Huiyong et al. (2017). "Using Learning Analytics to Support Computer-Assisted Language Learning." In: (cit. on p. 12).

Nguyen-Duc, A. et al. (2020). *Fundamentals of Software Startups: Essential Engineering and Business Aspects*. [Online; accessed 10-June-2020]. Springer International Publishing. ISBN: 9783030359836. URL: https://books.google.at/books?id=rZHTDwAAQBAJ (cit. on p. 23).

Oracle (2020). *MySQL 8.0 Reference Manual*. [Online; accessed 01-May-2020]. URL: https://dev.mysql.com/doc/refman/8.0/en/what-is-mysql.html (cit. on p. 26).

Panadero, Ernesto (2017). "A Review of Self-regulated Learning: Six Models and Four Directions for Research." In: *Frontiers in Psychology* 8. [Online; accessed 06-June-2020]. DOI: 10.3389/fpsyg.2017.00422 (cit. on p. 12).

PHP (2020a). *PHP - What can PHP do?* [Online; accessed 03-May-2020]. URL: https://www.php.net/manual/en/intro-whatcando.php (cit. on p. 30).

PHP (2020b). *PHP - What is PHP?* [Online; accessed 03-May-2020]. URL: https://www.php.net/manual/en/intro-whatis.php (cit. on p. 30).

Rubin, K.S. (2012). *Essential Scrum: A Practical Guide to the Most Popular Agile Process*. Addison-Wesley signature series. [Online; accessed 11-June-2020]. Addison-Wesley. ISBN: 9780137043293. URL: https://books.google.at/books?id=HkXX65VCZU4C (cit. on p. 16).

Schwaber, K. (2004). *Agile Project Management with Scrum*. Developer Best Practices. [Online; accessed 11-June-2020]. Pearson Education. ISBN: 9780735637900. URL: https://books.google.at/books?id=6pZCAwAAQBAJ (cit. on p. 15).

Siemens, George and Ryan Baker (2012). "Learning analytics and educational data mining: Towards communication and collaboration." In: *ACM International Conference Proceeding Series*. [Online; accessed 06-May-2020]. DOI: 10.1145/2330601.2330661 (cit. on p. 9).

Stufflebeam, D.L. and C.L.S. Coryn (2014). *Evaluation Theory, Models, and Applications*. Research Methods for the Social Sciences. [Online; accessed 30-September-2020]. Wiley. ISBN: 9781118870228. URL: https://books.google.at/books?id=IZW1BAAAQBAJ (cit. on p. 65).

Swift (2020). *About Swift*. [Online; accessed 05-May-2020]. URL: https://swift.org/about/ (cit. on p. 32).

Swift, O. (2016). *Programming: Swift: Create A Fully Functioning App: Learn In A Day!* Lulu.com. ISBN: 9781329837409. URL: https://books.google.at/books?id=q6qZCwAAQBAJ (cit. on p. 32).

Tarr, A. (2011). *PHP and MySQL 24-Hour Trainer*. EBL-Schweitzer. [Online; accessed 12-June-2020]. Wiley. ISBN: 9781118172933. URL: https://books.google.at/books?id=-wdKNwFLH2oC (cit. on p. 27).

Youngs, Bonnie, Sarah Moss-Horwitz, and Elizabeth Snyder (2015). "Educational data mining for elementary French on-line: A descriptive study." In: *Researching language learner interactions online: From social media to MOOCs*, pp. 347–368 (cit. on pp. 12, 13).

Yu, Qinglan (2015). "Learning Analytics: The next frontier for computer assisted language learning in big data age." In: *SHS Web of Conferences* 17. [Online; accessed 06-June-2020], p. 02013. DOI: 10.1051/shsconf/20151702013 (cit. on p. 6).

Zend (2020). *About Zend*. [Online; accessed 02-May-2020]. URL: https://framework.zend.com/about (cit. on p. 27).