

**BABEȘ-BOLYAI UNIVERSITY CLUJ-NAPOCA  
FACULTY OF MATHEMATICS AND COMPUTER  
SCIENCE  
SPECIALIZATION ENGLISH**

## **DIPLOMA THESIS**

# **Asynchronous Practice-Based E-learning Software Development**

**Supervisor**

**Maria Camelia Chisăliță-Crețu,  
Lecturer PhD**

*Author  
Groza Ionuț*

2022

**UNIVERSITATEA BABEȘ-BOLYAI CLUJ-NAPOCA  
FACULTATEA DE MATEMATICĂ ȘI INFORMATICĂ  
SPECIALIZAREA ENGLEZĂ**

## **LUCRARE DE LICENȚĂ**

**Principii de dezvoltare a aplicat, iilor  
asincrone pentru e-learning**

**Conducător științific**

**Lector dr.**

**Maria Camelia Chisăliță-Crețu**

*Absolvent  
Groza Ionuț*

2022



---

## ABSTRACT

---

The following paper proposes and analyses guidelines for developing E-learning systems based on practice, and provides an example of such a product. The goal of the product is to provide an E-environment for learners from all over the world, embracing their diverse fields of study in a configurable, simple and friendly System-Repetition FlashCard application.

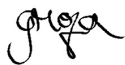
As students, language lovers, scientists keeping up with formulas, players who want to get better at chess techniques, soulful persons who memorize quotes for every situation, newbies in a musical instrument, for those who are preparing for a driving license, or a citizenship exam, there is a need for memorizing tools that both motivate and ease the process of achieving our goals.

The already existing tools either cluster in what concerns the subject learned (mostly language learning), or, on the other side, those who pretend to have a certain degree of customization, lack in making the process friendly or present inaccuracy in predicting the most appropriate time to memorize. Another major insufficiency is that not all of them are taking into considerations guidelines that were researched and proved to be efficient by world's specialists in psychology.

Subsequently, the main objective of the thesis is to offer guidelines for developing such systems, combining the two fields of psychology and computer science.

Both fields have met tremendous discoveries that were selected and compiled in the present paper in order to address the specific problems raised by asynchronous e-learning systems. Alongside the features that expedite learning, the scalability and the availability required by such systems also needs to be addressed at the most fitting trade off between efficiency and cost.

*This work is the result of my own activity. I have neither given nor received unauthorized assistance on this work.*



Groza Ionut

# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Motivation . . . . .	1
1.2	Objective and Chapter Description . . . . .	2
1.3	Original Contributions . . . . .	2
<b>2</b>	<b>E-learning Systems Problem Context</b>	<b>4</b>
2.1	Targeted User . . . . .	4
2.2	Functional and Non-Functional Requirements . . . . .	4
2.3	Technological Challenges . . . . .	9
2.4	Existing Solutions for the Technological Challenges . . . . .	10
<b>3</b>	<b>E-learning Systems Related Concepts</b>	<b>14</b>
3.1	Terminology . . . . .	14
3.2	Situating E-learning Environments . . . . .	14
3.3	General Attributes . . . . .	15
3.4	Specific Attributes . . . . .	17
<b>4</b>	<b>Client-Server Application Concepts</b>	<b>23</b>
4.1	Terminology . . . . .	23
4.2	Applicable Problems . . . . .	24
4.3	Technologies . . . . .	25
<b>5</b>	<b>E-learning System Development</b>	<b>35</b>
5.1	Requirements Gathering . . . . .	35
5.2	Analysis . . . . .	40
5.3	Architecture and Design . . . . .	42
5.4	Implementation . . . . .	54
5.5	Testing . . . . .	58
<b>6</b>	<b>Conclusions and Future Work</b>	<b>62</b>
	<b>Bibliography</b>	<b>63</b>

# Chapter 1

## Introduction

### 1.1 Motivation

This section describes the motivation of the present paper which reflects in the advantage of reusing previous knowledge and combining fields of study to understand the needs of humanity together with the possible solutions that could be implemented to fulfill those needs.

Recently, digital technology continued to evolve in a rapid cadence. Once with the implementation of Web 2.0 the traditional model of unidirectional instruction is shattered by the support of online multilateral exchanges that include visuals, audio, and text within and outside of the learning community. Social media and search engines allow learners to be knowledge receivers, producers, and distributors. Further, 3D worlds made popular by gaming applications offer learners such environments in which they can assume an avatar persona which can move around and interact with other participants or the immersive world. At the same pace, platforms have shrunk and diversified into range of mobile devices used for learning.

Nevertheless, the benefits one can gain from the new technologies can depend on the extent to which those are being used, specifically in ways that are compatible with human processes for cognitive learning. When tech enthusiasts become too excited about cutting-edge technology in such a manner that they human mental limitations are ignored, the true advantages of technology could not be used in ways that support learning. Having instructional methods who can support and not defeat human learning processes make for an essential ingredient of all course-ware related to effective e-learning. The most appropriate methods depend on the training goals (that be to inform or to perform); the related skills of the learner (whether they are familiar or new to the learned skills); and various environmental factors, including pragmatic constraints along with technological and cultural ones.

In these regards, stating, then following guidelines becomes more than advisory.

## 1.2 Objective and Chapter Description

The main objective of this paper is to highlight the baseline elements that compose into an e-learning application such that it answers to all three question that define e-learning. To show the effectiveness of the stated guidelines, the paper will present a multi-platform application and all the steps that were considered in the engineering process.

Reaching the objective is explained at large as follows:

*Chapter 1* provides an overview of the present paper, stating its motivation, objective, a short description of the chapters and a presentation of original contributions.

*Chapter 2* presents the general attributes of e-learning with respect to it's users. The key concept in this chapter is the user, and it presents how technology could meet psychology to the benefit of the learner. Further on, it analyses and compares different asynchronous e-learning software that have known popularity on the market.

*Chapter 3* analyzes guidelines and elements that should be contained in an asynchronous, practice based, e-learning software. It covers the following topics: multimedia, virtual coach, practice in the form of question-answer cards or of simulation games, gamification and social media, constant feedback and visual statistics.

*Chapter 4* focuses on the development process of e-learning applications. It presents different technologies that modern world offers and that could be used together to smooth out the developing process.

*Chapter 5* presents an example of how a real application that uses concepts from psychology can be implemented for the real use of learners. The steps from the development life-cycle of an application are put together and documented.

*Chapter 6* aims to conclude the subject discussed in this paper and to canvass possible future work.

## 1.3 Original Contributions

Original contribution consists in gathering the most influential and effective guidelines and elements for engineering an asynchronous, practice based, e-learning software. These guidelines are further used in the practical development of such a software as described above, manipulating modern technology in forms of frameworks, platforms, architectures and best practices. The original contributions are restated below:

- study of the scientific literature on the e-learning systems attributes.
- analysis of particular LMS's and E-learning systems.

- asynchronous e-learning application development
- study of modern technologies that fit in the development of an asynchronous e-learning application which can be further detailed:
  - study and implementation of a serverless architecture for image handling to reduce the overhead of using blobs.
  - study and implementation of a cache mechanism to reduce the cost of the above mentioned solution.
  - study and integration with an intelligent system for quiz scheduling that favors learning.



# Chapter 2

## E-learning Systems Problem Context

The following chapter discusses the problem of E-learning. Systems that are developed to address E-learning should improve or even replace some of the processes that appear in a traditional learning system. Since the concept of E-learning is not brand-new, the society does not lack in providing solutions. At the end of this chapter, two such systems are analyzed from the perspective of the challenges they try to solve regarding humans learning process.

### 2.1 Targeted User

The following chapter describes the characteristics of an e-learning system user. Most such systems have two principle authors: the learner and the teacher.

Learners benefit from the e-learning systems from various ways: in what concerns the content they are learning, the techniques that are available to ease the process of learning, the interaction with teachers and schooling requirements.

As much as the Teacher is concerned, interactivity with the application should permit selecting, organizing, providing content as well as the access of learners to the content. Certain types of e-learning systems could also be used to automate some schooling system, allowing teachers to contain access to some functions of managing learners performance.

As managing users access in an e-learning system is discussed, both types of users may need to authenticate for certain roles.

### 2.2 Functional and Non-Functional Requirements

The following section analyzes some of the functional and non-functional requirements of an e-learning systems from the perspective of both users: Learner and Teacher.

As stated in the previous section, both types of users may need to authenticate themselves to the system. Authentication may enable some features that could not be implemented otherwise. For example, Moodle offers a grading system. Assessing a certain grade to a specific student implies the specific student to be uniquely identifiable. Further, E-learning systems that are focused on content, may demand some courses to be available only to users that accomplish some requirements (for example, users that have previously bought access to the course content, or users that have previously completed a course that is required in order to take another course that demands knowledge gained from the previous one).

The most important features that E-learning systems provide to a user of type `teacher` could be summarized to managing the system's content and communicating with learners.

Vice-versa, the most important features that E-learning systems provide to a user of type `learner` could be summarized to using the system's content and communicating with teachers.

Among non-functional requirements of e-learning systems one could find the degree at which the system could be used by non IT experts, the degree at which the system can be made ready-to-use without the intervention of experts or the degree at which users can reach out for support.

Nowadays, there are some alternatives available concerning software that provide E-learning systems or (Virtual Learning Environments), whether Open Source or commercial. Further, the paper will focus on the comparison between several e-learning systems to state their strengths and limitations, according to [AAZ08].

Features that concern the learner could be categorized as follows of tools, which are Communication Features, Productivity Features and Student Involvement Features. The products analyzed over these categories can be seen in 2.1.

Features categorized by their support capabilities could be seen in 2.2, according to [AAZ08].

Another comparison degree can be the Technical Specification of various e-learning systems, which are described in 2.3.

The final score of the described systems can be seen in 2.4. As can be seen, there is no perfect system, and the difficulty of implementing system that address all users requirements and features is considerable. The analysis in [AAZ08] proves that there are no products to meet all the criteria that were stated and obtaining an ideal system in terms of interface, technical, functional, or cost reasons is non-viable.

No	1	2	3	4	5	6	7	8	9	10
Product Name	Desire2Learn 8.1	KEWL	ANGEL Learning Management Suite (7.1)	eCollege	The Blackboard Learning System	Moodle 1.8	Claroline 1.6	Dokeos 2.1.1	OLAT	Sakai 2.3.1
Tools										
<b>1. Learner Features</b>										
<b>1.1. Communication Features</b>										
Discussion Forums	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Discussion Management	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
File Exchange	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Internal Email	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
On-line Journal	Y	Y	Y	N	Y	Y	N	Y	Y	Y
Real-time Chat	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Video Services	N	N	N	N	N	Y	N	N	N	N
Whiteboard	Y	N	Y	Y	Y	Y	N	N	Y	Y
<b>1.2. Productivity Features</b>										
Bookmarks	Y	Y	Y	Y	N	N	Y	N	N	Y
Calendar	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Orientation	Y	Y	Y	Y	Y	Y	N	Y	Y	Y
Searching Course	Y	Y	Y	Y	Y	Y	Y	N	Y	Y
Work Off-line	Y	Y	Y	Y	Y	Y	N	Y	Y	Y
<b>1.3. Student Involvement Features</b>										
Group work	Y	Y	Y	Y	Y	Y	Y	N	Y	Y
Community	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Student Portfolios	Y	Y	Y	Y	Y	Y	Y	Y	N	Y
<b>Total Features</b>	<b>16</b>	<b>16</b>	<b>16</b>	<b>16</b>	<b>16</b>	<b>16</b>	<b>16</b>	<b>16</b>	<b>16</b>	<b>16</b>
<b>Total Available</b>	<b>15</b>	<b>14</b>	<b>15</b>	<b>14</b>	<b>14</b>	<b>15</b>	<b>11</b>	<b>12</b>	<b>13</b>	<b>15</b>
<b>Total Missing</b>	<b>1</b>	<b>2</b>	<b>1</b>	<b>2</b>	<b>2</b>	<b>1</b>	<b>5</b>	<b>4</b>	<b>3</b>	<b>1</b>

Figure 2.1: Learner Features [AAZ08]

<b>2. Support Features</b>										
<b>2.1. Administration Features</b>										
Authentication	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Authorization	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
File Exchange	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Registration Integration	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
<b>2.2. Course Delivery Features</b>										
Test Types	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Automated Management	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Automated Support	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Course Management	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
On-line Grading	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Student Tracking	Y	Y	Y	Y	Y	Y	Y	N	Y	Y
<b>2.3. Content Development Features</b>										
Accessibility	Y	Y	Y	N	Y	Y	Y	Y	Y	Y
Content Sharing	Y	Y	Y	Y	N	Y	Y	Y	Y	Y
Course Templates	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Look and Feel	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Design	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Instructional Standards	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
<b>Total Features</b>	<b>16</b>	<b>16</b>	<b>16</b>	<b>16</b>	<b>16</b>	<b>16</b>	<b>16</b>	<b>16</b>	<b>16</b>	<b>16</b>
<b>Total available</b>	<b>16</b>	<b>16</b>	<b>16</b>	<b>15</b>	<b>15</b>	<b>16</b>	<b>16</b>	<b>15</b>	<b>16</b>	<b>16</b>
<b>Total Missing</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>1</b>	<b>1</b>	<b>0</b>	<b>0</b>	<b>1</b>	<b>0</b>	<b>0</b>

Figure 2.2: Support Features [AAZ08]

<b>3. Technical Specifications</b>										
<b>3.1. Hardware/Software Tools</b>										
Client Required	Y	Y	Y	Y	Y	Y	N	Y	N	Y
Database Requirements	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Unix Server	N	N	N	N	Y	Y	Y	Y	Y	Y
Windows	Y	Y	Y	N	Y	Y	Y	Y	Y	Y
<b>3.2. Pricing/Licensing Tools</b>										
Company Profile	Y	Y	Y	Y	Y	N	N	N	Y	N
Costs	N	N	N	N	N	Y	Y	Y	Y	Y
Open Source	N	N	N	N	N	Y	Y	N	Y	Y
Optional Extras	Y	Y	Y	Y	Y	Y	N	Y	Y	Y
<b>Total Features</b>	<b>8</b>	<b>8</b>	<b>8</b>	<b>8</b>	<b>8</b>	<b>8</b>	<b>8</b>	<b>8</b>	<b>8</b>	<b>8</b>
<b>Total available</b>	<b>6</b>	<b>5</b>	<b>5</b>	<b>4</b>	<b>6</b>	<b>7</b>	<b>5</b>	<b>6</b>	<b>7</b>	<b>7</b>
<b>Total Missing</b>	<b>2</b>	<b>3</b>	<b>3</b>	<b>4</b>	<b>2</b>	<b>1</b>	<b>3</b>	<b>2</b>	<b>1</b>	<b>1</b>

Figure 2.3: Technical Specifications [AAZ08]

No	1	2	3	4	5	6	7	8	9	10
Product Name Tools	Desire2Learn 8.1	KEWL	ANGEL Learning Management Suite (7.1)	eCollege	The Blackboard Learning System	Moodle 1.8	Claroline 1.6	Dokeos 2.1.1	OLAT	Sakai 2.3.1
<b>Total Features</b>	<b>40</b>	<b>40</b>	<b>40</b>	<b>40</b>	<b>40</b>	<b>40</b>	<b>40</b>	<b>40</b>	<b>40</b>	<b>40</b>
<b>Total Available</b>	<b>37</b>	<b>35</b>	<b>37</b>	<b>33</b>	<b>35</b>	<b>38</b>	<b>32</b>	<b>33</b>	<b>35</b>	<b>38</b>
<b>Total Missing</b>	<b>3</b>	<b>5</b>	<b>3</b>	<b>7</b>	<b>5</b>	<b>2</b>	<b>8</b>	<b>7</b>	<b>5</b>	<b>2</b>

Figure 2.4: Final Results [AAZ08]

## **2.3 Technological Challenges**

This section details several types of challenges that could be arising in the development of virtual learning environments and includes technological challenges, elements of creativity and a section of future possible research.

According to [RRL14], in the research areas related to technology, it is addressed the concept of developing e-learning systems that meet the users requirements. In this context two challenges that can not be neglected when it comes to e-learning are raising and are going to be discussed in the rest of this section.

### **Learning Communities and Interactive Learning**

Learning receives a lot of support when integrated with environments that facilitate community, interaction and cooperation. The developments in the area of e-learning environments provide new forms of interaction for learning experience. It generates new relationships between learner and computer and also form a new learning community. Key issues include [RRL14]:

- New forms of interacting with the system to support learning.
- New techniques to facilitate learning communities.
- The development mobile adaptable systems.
- Techniques for customizing the learning process to the learner's personal needs.
- Techniques to facilitate synchronous learning.
- Discovery of new learning communities.
- Systems for evaluating learner's performance.

### **Developing New Knowledge Facilities for E-learning**

E-learning environment needs to support the rapid increase in the size and variety of data by appropriate semantic services. The semantic services generate a surrounding semantic context for learning support. Research that needs to work on [RRL14]:

- Handle uncertain and incomplete knowledge with the help of new reasoning theories for learning.
- Systems adaptable to large-scale learning facilities.
- Handle dynamic processes that involve learning.

- Manipulation of information across different learning environments.
- Data gathering techniques for the analysis of humans learning process.
- Systems that handle different domains and users requirements

## 2.4 Existing Solutions for the Technological Challenges

This section studies two models of e-learning software in the light of the challenges described in the previous section. One such system relies on teaching languages using psychological techniques for retaining words and phrases over a long period of time, while allowing the user to customize the intensity of his study program. The other system focuses on promoting learning using creative ways that involve learning communities.

### Mosalingua

Mosalingua is an asynchronous E-learning software and is focused on "independent language learning". The environment is used by 11 million users around the world that can benefit from the specialized sub-modules for teaching 10 different languages.

MosaLingua's [Mos] effective learning method which is based on several cognitive science and psychology concepts, that will be presented below:

**Spaced repetition systems** rely on the popular opinion that the more a certain concept is being reviewed, the more likely is the one reviewing it is to remember it later. While this can be proved to be right, popular beliefs does not also solve the problem of what would be the best way to perform these review sessions. The least efficient way of reviewing is to learn the same concept multiple times right before a test. This implies that the information will be forgotten in only a few days later. One way to efficiently organize these review sessions such that the information will be stored to the learner's long-term memory is spacing out the review sessions. The review schedule will be different depending on the learner as well as on the concept to be learned. One example could take the form of learning a word for the first time, then review it 5 minutes later, then again 7 hours later, then 3 days later, then 10 days later, then 1 month later, until the word has been stored in the long-term memory. Such a scheduling technique is based on the forgetting curve 2.5 and the cause-effect element is represented by spacing the review sessions apart so that the information will be reviewed just before it is likely to be forgotten, slowly propagating the information in the long term memory [BBBB93].

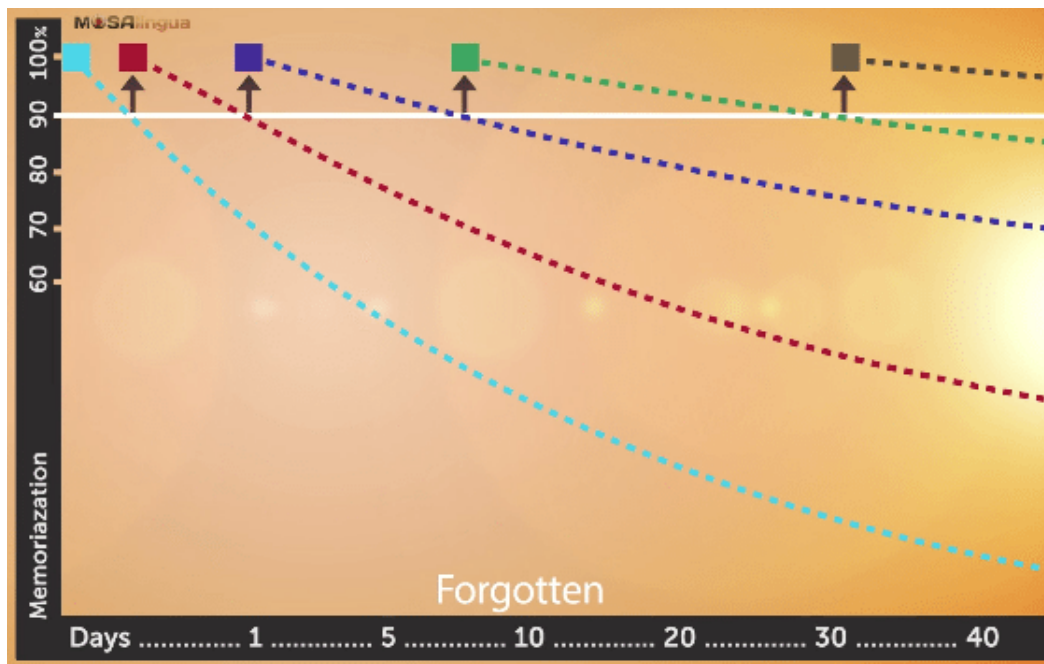


Figure 2.5: The SRS curve: Note that at each spaced review session, it takes increasingly more time for you to forget the information. [Mos]

**Active Recall.** Another way to implement new techniques for supporting fast life-long learning is Active Recall.

The effectiveness of multiple choice questions is put to doubt whether they make for efficient memorization techniques when they only require from learners to recognize the correct answer from a list of possible answers.

The efficiency could be improved if the learner is compelled to extract the information from his memory, without assistance or a clue.

MosaLingua's active recall principle is based on using a system of flashcards that requires more thinking effort but is proven to be much more effective due to the act of regularly extracting information from the memory. This technique reinforces memorization [PBB<sup>+</sup>07]. In other words, the learner's neurological network that routes to a certain piece of information becomes faster and more reliable.

Apart from its cognitive advantage, the method also implies some kind of amusement that comes with the process of recalling a concept by drawing from deep within the learner's memory, then flipping the card to get the feedback.

**Metacognition.** The simple act of thinking of an answer then revealing it would be incomplete without the third and very important component of this learning system: metacognition. It represents the act by which one reflects on his own thoughts. The process is implemented in Mosalingua by asking the users to evaluate their memorizations (on a scale of 1 to 4) before revealing the answer. This process of reinforcing one's memory is highly effective, as stated in [SG06].

To address the challenge of lacking motivation from users, which is also the pri-



mary cause of failing to learn new skills. Mosalingua developed ways to overcome the lack of motivation and transform learning into an enjoyable habit. By using them, the system proposes to help learners with managing their motivation (which varies with time).

Another challenge that arise in understanding personal needs is roadblocks in some specific skills (often formed during learner's education). This affects the learner because of the false ideas about the specific skill to be learnt that they have subconsciously.

Subsequently, to address learner personal needs, Mosalingua also offers free learning help via email, presenting the Web's best resources, as well as tips through bonus material or the learning community on the MosaLingua blog.

### **Kahoot!**

Founded in 2012 Kahhot! is based on research conducted by its co-founder Morten Versvik, a student of Professor Wang's at the time, for his Master's degree at the Norwegian University of Science and Technology (NTNU) [SM07].

**Learn by playing.** The mission of Kahoot! as they describe it is to provide life-long learning skills through curiosity and play. By combining the two, in a fun and social way, one can unlock the learning potential no matter the subject, age or ability.

Their solution consists in implementing an online platform with access to the following games:

- Users can use a video conference to host a live kahoot that connect students from their homes or via a big screen in class! Questions along with some answers will be displayed on screen so that everyone can see them, while students answer separately on their devices.
- Distance learning can be leveraged by assigning kahoots that students can play at home, at their own pace. Questions and answers are displayed on players' screens.
- Assign self-paced kahoots that students can play anywhere, anytime on their own devices – another feature for distance learning. Questions and answers will be displayed on their screens.

Though both platform met real success, being accessed by millions of users. Since they are different, each system addresses different technological challenges (see 2.1). Mosalingua focuses on learning techniques, while Kahoot! focuses on playing and learning communities.

While learning should be the purpose, playing and socializing should not be set aside.

Challenges	MosaLingua	Kahoot!
multimodal interface and personalisation techniques	provides multimedia enriched content for cards. Cards can contain images, text and sound. It also provides dialogues with transcripts. However, the most important concept when it comes to multimodal interface is the hands free mode and they intend to implement a voice recognition system to complete the feature	Multimedia rich Kahoots and video conferencing
support learning communities and promote and support interaction	Mosalingua offers free learning help via email, presenting the Web's best resources, as well as tips through bonus material or the learning community on the MosaLingua blog	This should be the domain of excellence for Kahoot!. They provide communities for teachers as well as for learners. Kahoot! Academy is a global community and knowledge platform for all creators, learners and learning providers. Each month they host more than 40 million participating players globally who engage with the ready-to-use content. Moreover, Kahoot! is supposed to represent a gamified multiplayer learning experience, so learners can also socialize during the games
support mobile communities of learners	Both systems support multiplatform access	
promote and support interaction	Limited, MosaLingua Blog	A time-lapse animation of seed germination
dynamic learning, lifelong learning, large-scale learning	MosaLingua learning system has proven to be very efficient through its methods described above: spaced-repetition, metacognition and active recall	Kahoot! touches billions of non-unique participants in their games. The techniques used are based upon gamification techniques that are described in [SM07] master thesis

Table 2.1: Challenges to which MosaLingua and Kahoot! respond

# Chapter 3

## E-learning Systems Related Concepts

The following chapter aims at defining E-learning Systems with highest consideration on Asynchronous E-learning Systems. Towards the end of the chapter are described some of the main attributes that can improve the efficacy of such systems.

### 3.1 Terminology

In the following paragraphs it will be stated the definition of E-learning, considering three main aspects that contribute in contouring its meaning.

The definition of E-learning as stated in [CM11] can be probed in three ways: probing what is e-learning, probing how e-learning is unfolded and probing why e-learning is conducted.

*What.* The information being presented and the techniques used to catalyze the process of acquiring it, both form the what component of e-learning systems.

*How.* The information presented in E-learning systems is communicated by the use of digital devices (as computers, smartphones, tablets etc.) that can convert the traditional way of sharing information -by words- into multimedia elements.

*Why.* Whether for skill building in companies or for personal learning, E-learning lessons are designed to help learners reach their learning objectives or perform their jobs in ways that improve the outputs of the company.

### 3.2 Situating E-learning Environments

This section aims to analyze the current state of e-learning.

Although it has been a deep studied subject, in today's technology-supported and technology-enabled educational, professional, and social situations, there is a continuing need for research and knowledge of learning behaviors.

Although research has made significant progress, it has also taken numerous new avenues, each grappling with how to study and portray learning in an era of rapid change in technologies, learning methods, and knowledge distribution.

The deployment of institutional learning management systems (LMS), has been linked with the phrase e-learning at times (VLE). E-learning, on the other hand, covers significantly more than just technology and educational institutions.

The focus of key e-learning challenges is on how to teach online, how to bring institutional resources into the service of distributed learning, and how to study and practice using different interfaces.

The increased usage of video-based resources for teaching and learning as well as games and gamification of learning, dashboards that highlight progress or effort in comparison to other learners, adaptive learning systems that identify future steps based on learner progress and types of errors, have all gotten a lot of attention in the e-learning world.

Further effects, such as context, beliefs, design choices, adoption patterns, and/or devices, are now being included in research and views, with an increased focus in how these affect learning opportunities and practices. Throughout this work, these concepts and their application to e-learning systems are discussed and analyzed with the purpose of providing a set of guidelines that will lead to more qualitative e-learning software, built with more ease.

Some of the general as well as specific attributes of e-learning systems are going to be discussed in the next section.

### 3.3 General Attributes

This section describes the two main division of E-learning, asynchronous and synchronous E-learning, as described by [Hra08].

**Asynchronous e-learning** supports productiveness among learners and teachers, even when contributors cannot be available at the same time, thanks to media such as e-mail and discussion boards. As a result, it is an essential component of adaptable e-learning. Indeed, one of the most prominent reasons humans take on-line courses is due to their asynchronous nature. This way, they can balance academic achievement with work, family, and other responsibilities. Asynchronous e-learning allows students to access an e-learning environment at any time and download documents or communicate with teachers and fellow students. When compared to synchronous communication, students may spend more time refining their contributions, which are generally regarded as more perceptive.

**Synchronous e-learning** has the potential to assist e-learners in the construction of learning communities, which is widely backed by media such as videocon-

ferencing and chat. By asking and answering questions in real time, learners and teachers perceive synchronous e-learning as more social. Also, having answers to their questions in real time can also help to avoid frustration. Another aspect is that synchronous classes make e-learners feel more like participants rather than isolated individuals.

The two divisions of E-learning could be provided by a single LMS (Learning Mechanism System), or multiple LMS could work together to allow the learners and instructors to interact efficiently. A definition of LMS is stated by [KK16] in the following manner.

**Learning Management Systems** are platforms that include various learning systems, course and content management systems as well as portals and instructional management systems. Learning Management Systems evolved from the processes and systems adopted by specific institutions to register students for specific courses and keep an ongoing track of their efforts. Various learning options have been established to allowing learners to take online classes, sometimes as part of a structured curriculum and other times due to the requirement for institutional certification. Course instructions, uploading assignments and downloading grades, tools allowing students and teachers actively interact with each other, facilitating the communication of students with their peers, easing the usage of other learning tools, allowing knowledge sharing, and conducting online evaluations are all working together a LMS.

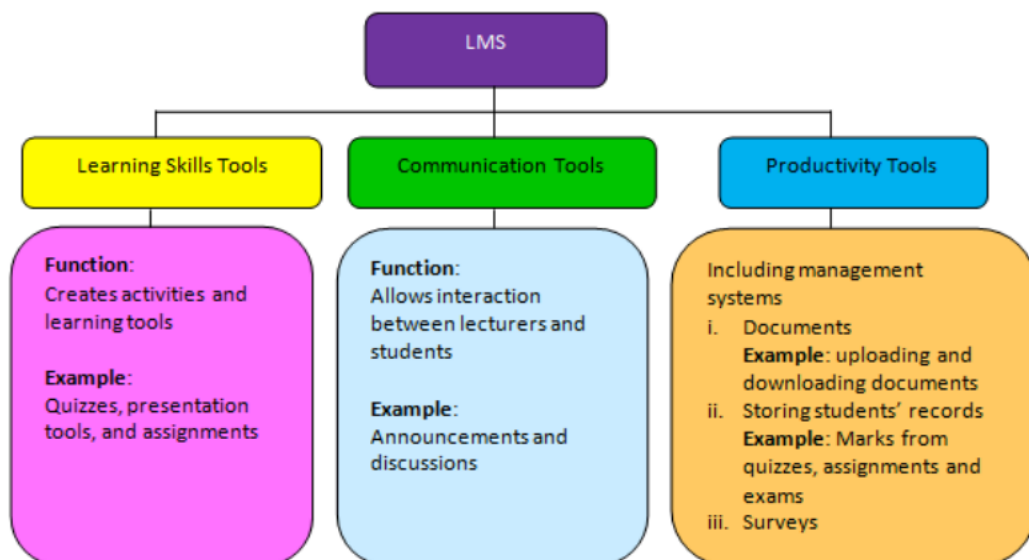


Figure 3.1: LMS Tools [KK16]

Based on [KK16], LMS can be grouped into three main types: tools for learning skills, tools facilitating communication, and tools for boosting productivity.

Tools for learning skills can be composed of learning module that generates certain tasks and activities for learners. Quizzes, online presentation tools, and assignments are all included in such tools:

- A quiz module would contain a number of features, including a database for questions together with their possible answers, an evaluation procedure, and a way to help students improve themselves.
- The online presentation tool allows users to upload presentations to the LMS or connect to them from other websites such as YouTube.
- The assignments are first uploaded by the lecturer to the LMS, students will complete the assignment online, and they will be able to revise or send the assignment at any moment until the deadline.

The second category promotes tools facilitating communication. These tools facilitate contact between professors and students, as well as student-to-student interaction. The most popular communication medium is announcements, which are used to present any information about the course to all students, including the most recent news and forthcoming activities. Furthermore, discussions are incorporated as communication tools, allowing both students and lecturers to publish and reply to messages, as well as view other users' remarks.

The last category of LMS tools, productivity boosting tools, include systems for document management, calendars or surveys, among others. Lecturers and students can upload and download files from any computer with an internet connection using document management solutions. Other LMS management solutions collect data on how often students use the LMS and how well they do. Other LMSs allow students to get reports of their total performance, including grades for each assignment, quiz, and test. The summary of LMS tools is represented in diagram 3.1.

## **3.4 Specific Attributes**

### **Multimedia**

Both cognitive theory and empirical results [CM11] recommend that e-learning courses should include both words and pictures avoiding relying only on words for communicating the information, where words could mean either written text (that is, words that people read on a screen) or spoken text (-i.e., words communicated through speech on digital devices). Static illustrations such as charts, drawings, maps, photos or graphs, as well as dynamic graphics such as animations or videos, are referred

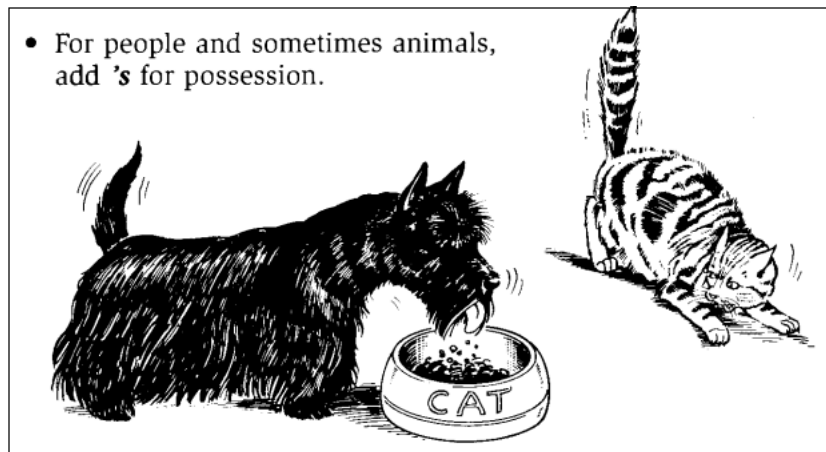


Figure 3.2: Example of Decorative Graphic [WE00]

to simply as graphics. Any presentation that includes both words and graphics is referred to as a multimedia presentation.

One multimedia principle stated in [CM11] is promoted in the following way: instead of picking visuals after the words have been composed, instructional designers should think about how words and pictures cooperate to provide the intended meaning to the learner. As a result, when conducting a job analysis and designing a course, pictures and words should be coordinated simultaneously.

The above guideline is based on the fact that people are more likely to understand the proposed content if they can participate in active learning. -that is, when they Active learning is the process by which learners focus on the important materials in the lessons, they organize the it into a coherent cognitive picture, and then consolidate the new acquired knowledge with their prior comprehension of elements around it.

By mentally picturing the subject in words and graphics and then mentally connecting the information so portrayed, multimedia presentations can inspire learners to engage in active learning. Presenting only information as words, on the other hand, may induce the learners with limited knowledge in certain area into participating superficially in the learning process, failing to connect the words to other knowledge.

Below, in the table 3.1, are analyzed different types of graphics, according to their functionality, as described by [CM11].

### Easiness of Access

Developing learning environments that fully use the relative strengths and weaknesses of the various platforms used to accomplish learning objectives is an essential aspect of multi-platform or blended learning. The desktop works in favour of

Graphic Type	Function	Example
Decorative	Illustrations that improve the overall aesthetics or are intended to provoke cheerfulness	A dog eating from cat's bowl in a lesson on possessives (an example can be seen in diagram 3.2)
Representational	Pictorial representations of an object's look and characteristics	A picture of an anatomic heart in a book on human's anatomy
Organizational	Visuals that depict qualitative correlations between the presented knowledge	A tree diagram
Relational	Visuals that depict quantitative correlations between the presented knowledge	charts and graphs
Transformational	Visuals that depict state transitions in time or space	A landscape depicted in all four seasons. An example can be seen in 3.3
Interpretive	Visuals that make intangible phenomena visible and concrete	Drawings of molecular structures

Table 3.1: Organization of Graphics by Functionality [CM11]

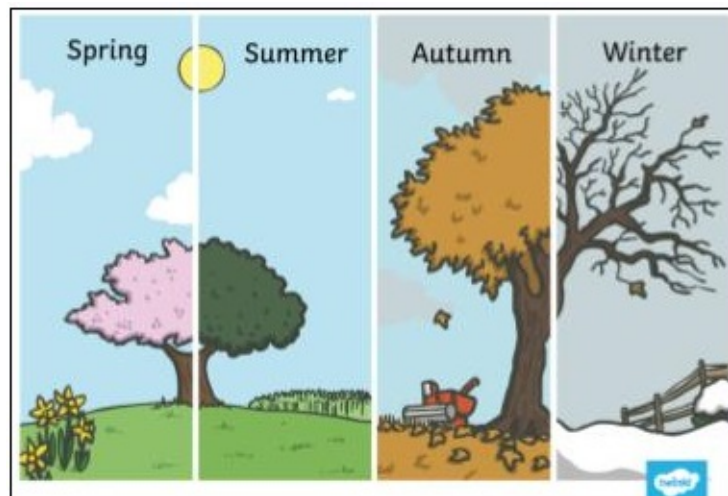


Figure 3.3: Example of Transformational Graphic [Twi]



immersive learning environments. Thriving of mobile devices, on the other hands, happens in contextual learning and performance support roles.

The use of mobile technology in education is a unique instrument that reaches its full potential when employed as part of a multiplatform strategy. Depending on the nature of the educational purpose, the advantages of a mobile device, such as accessibility and portability, might also be disadvantages. However, huge advances in efficiency and efficacy are conceivable if classroom, desktop, and mobile approaches can be linked to complement one another by leveraging on each's capabilities as it is further explained in [MR].

One of the most important features of mobile devices is regarded as its relative independence from specific locations at any time for accessing the learning material. According to a recent study [PKH07], many urban users reported that their daily travel times were one of the most favorable times to study instructional content on their mobile device. This unprecedented accessibility lowers the obstacles to short learning and encourages regular encounters. In contrast to typical classroom training, which requires a great deal of preparation and planning to implement, a mobile learning session can be relatively spontaneous.

### **Practicing Mechanism**

Effective e-learning involves learners in ways that encourage the selection, organization, integration, and retrieval of new information. First and foremost, the relevant material in the training must be highlighted. The instructive words and visuals must then be merged with one another, as well as with past knowledge. Finally, following the training, the new knowledge and abilities that have been stored in long-term memory must be so retrieved when demanded. All of these cognitive states are supported by effective practice routines. The rest of this section aims to analyze research and principles for getting the most out of online practice.

By practice one can obtain an improvement in his performance on a given topic, although at a decreasing rate. As it can be seen in 3.4, Even after a sufficiently large number of practice sessions, one can still see progress in its learning process. However, the biggest proficiency increases come during early trials as students become better at completing the tasks in creative ways. Similarly, in later practice sessions as students can become more efficient in completing the given tasks, but this is attributed to automaticity rather than creativity.

### **Gamification**

Even in traditional learning, the idea of making use of games to facilitate learning is common practice. It may be difficult, however, to adapt games into a digital form of

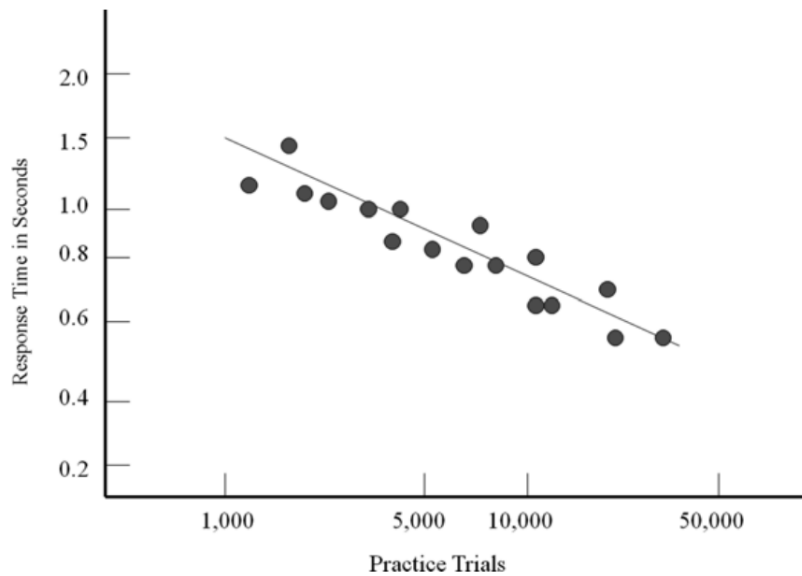


Figure 3.4: The Power Law of Practice: Speed Increases with Practice But at a Diminishing Rate [CM11]

learning, gamify the e-learning process so to say, which will be discussed below.

Gamification can be defined by the process of making use of some well designed systems called game-play mechanics which one can further use to obtain certain results in non-game applications, processes, different contexts or different tasks. The game-play mechanics techniques such as scoreboards and personalized fast feedback can be so put together in order to make users more engaged with tasks.

From a pedagogical standpoint, e-learning has some practical limits in that it cannot express emotion or engage students in the same way that a teacher can. An e-learning system must compensate for this lack of affective interaction by attempting to motivate learners through other means [Mun11].

Gamification is a relatively new concept, both on the market and in research, but it has a lot of prospect, developing its popularity by yielding effective results in certain websites that use it as a technique to enhance loyalty, advertising and effective marketing engagement.

Two concepts that inevitably appear when talking about Gamification are intrinsic and extrinsic motivation. Intrinsic motivations come from within, the user/actor has the ability to enact from his own will, giving birth to competition, cooperation, altruism or sense of belonging. Extrinsic motivations, on the other hand, occur when the user acts as a result of him being manipulated by external factors, for example: levels, points, classifications, missions, badges, awards [Vio11]. These can be put together to increase users motivation.

When employed in e-learning contexts, however, there are certain possible drawbacks to gamification. One such example could be educating pupils that they should

only learn by extrinsic motivations rather than inspiring them to participate and offer help to teachers, which can happen if the gamification design does not serve its objective [Mun11].

# Chapter 4

## Client-Server Application Concepts

The following chapter details the development of an E-learning application as a client-server application as well as how different technologies could be leveraged to address the specific features of an e-learning system.

### 4.1 Terminology

This section provides the definition for the concept of client-server architecture and details the linkage with another type of architecture, namely peer-to-peer.

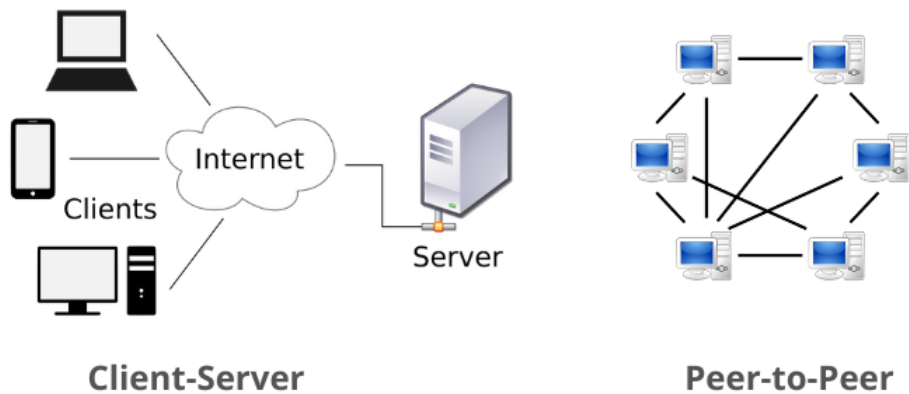
Client-server architecture is a computing approach in which the server hosts, provides, and controls the majority of resources and services that are to be consumed by the client. One or more client computers are connected to a central server through a network over the internet. Because all requests and services are supplied across a network, client-server architecture is also known as a networking computer paradigm or client-server network.

In addition to the client-server model, peer-to-peer (P2P) software architecture is also frequently used in distributed computing applications.

In the client-server model, the server is frequently built to function as a centralized system that serves a large number of clients. Because of the possibility of multiple clients to be accessing the server simultaneously, a server's processing power, memory, and storage requirements must be adjusted to the predicted workload. As a solution, client-server architecture proposes the use of a Load-balancing systems that is able to scale the server such that it runs on multiple physical devices.

Subsequently, load balancing is defined as the logical and efficient distribution of network or application traffic among numerous servers in a server farm. Each load balancer lies between client devices and backend servers, accepting and then distributing incoming traffic to any server that can handle them.

A peer-to-peer network consists of two or more computers that share their re-



---

Figure 4.1: Client-Server vs P2P architecture [Net]

sources and communicate in a non-centralized manner. Each computer is called a Peer. Peers function by exchanging information directly with one another. As opposed to a client-server architecture, where clients are only requesting information that was previously processed by the server, peers are all occupying the same hierarchical level in the network. A suggestive diagram can be seen in 4.1;

A load-balancing algorithm in the peer-to-peer paradigm means balancing the load in such a manner that all peers, even those limited in resources, can contribute to share the load. If a node goes down, its shared resources are still available as long as other peers are willing to supply them. This way, high availability is not necessarily required from a peer because other peers may compensate for any resource shortages. Because the peers may vary in their availability and load capacity it may be needed that certain requests to be rerouted.

## 4.2 Applicable Problems

Students from Waterloo University of California have stated a list [Nom] of applicable problems of the Client-Server Architecture, which will be discussed in the following list:

- The client-server architecture is best suited for systems that require a separation of concerns between the client and the server; it is designed for software that relies heavily on exchanging and making use of information. The client-server architectural style aids programs in scalability and performance.
- The client-server architectural concept is ideal for applications that require functional separation. The client might handle request validation and input,

while the load balancer directs the request to the server for proper processing. The server is in charge of handling the client's request and returning the result via the appropriate protocol. Because of their capability of handling tasks independently the client and the server are a handy tool when there appears the need for abstracting functionality; The client, for example, does not need to understand how the server performs user authentication or request validation.

- The The system need to be able to perform more efficiently at large scale. Within the client-server architecture, modern solutions have been developed to address scalability issues such as load balancing, sharding, and partitioning. On the server side of the architecture, these strategies increase performance for numerous requests and will be valuable for software programs that deal with multiple requests per users..

### 4.3 Technologies

From the invention of client-server architecture, there have been studied and developed a wide range of technologies that facilitate the development of this kind of applications. This section discusses some of these technologies.

#### RESTful APIs

A modern UI framework and an API are the two primary components of a client-server application. APIs are the modern method of retrieving data.

A specific kind of modern API is represented by RESTful APIs [Rai19]. REST (representational state transfer) was initially proposed by Roy Fielding in [Fie00]. The most dominant software platform that uses REST architectural style is the World Wide Web itself. The way REST generally works is by using verbs like GET, POST, PUT, DELETE, etc. to communicate over HTTP. Browsers, for example, retrieve web pages and transfer data to remote servers in the same way.

A common design can use `@RestController` annotation from Spring web framework MVC and create a REST API, with endpoints that act by publishing and consuming JSON in relation to clients. Different devices that impersonate clients (iOS, Android, IoT devices, etc.) can be provided with data, since RESTful facilitate the decoupling of business logic and the database from the client. This separation also makes it possible to integrate an app with third-party and partner capabilities.

## Server with Spring Boot

Spring simplifies the development of Java enterprise applications. It includes the settings required to use the Java programming language in a business setting. It allows the construction of a variety of architectures based on the application requirements.

Spring Framework is broken down into modules that, can further be selected by specific applications, for specific purposes. The most important modules, which are also fundamental, include a configuration model and a dependency injection mechanism. In addition, the Spring Framework includes core support for several application architectures, such as messaging, transactional data and persistence, and web applications, along with a Servlet-based Spring MVC web framework. One disadvantage of Spring, however, is the amount of configurations that need to be tackled in order to obtain what it really offers.

A solution to this problem is Spring Boot. It can be used in the development of stand-alone or industrial Spring-based applications with the promise of needing very little Spring configuration. As a result, developers may take advantage of Spring Framework modules with little configuration, allowing them to focus more on business logic.

The diagram 4.2 (from <https://spring.io>) shows how Spring Boot is the entry point to the rest of the Spring ecosystem.

`DemoApp.java` can be regarded as the core of a Spring Boot app as long as the class contain the `@SpringBootApplication` annotation and the `main` method.

Further, the application could contain the entities, a REST controller (as pointed in the above subsection) or a JPA repository which is a Spring Boot short way of instantiating a Spring Data JPA. This is a method to create a CRUD repository for an entity and it has the advantage of providing (implementing) the means for communicating with the underlying database.

Spring has made its name as the most popular tool for Java application development around the world. Along with a set of starter dependencies that can be used in both, Maven and Gradle environments to simplify build configuration, it also include a list of features (modules) that facilitate application development:

- **External configuration.** Spring Boot may be configured externally, allowing the use of the same application code in multiple settings, with configurations that can be exported through one or more of the following techniques: properties files, ENVARS, YAML files and command-line arguments.
- **Automatic configuration.** Spring Boot can configure Spring automatically whenever feasible. One can use this feature that involves a `META-INF/spring.factories`

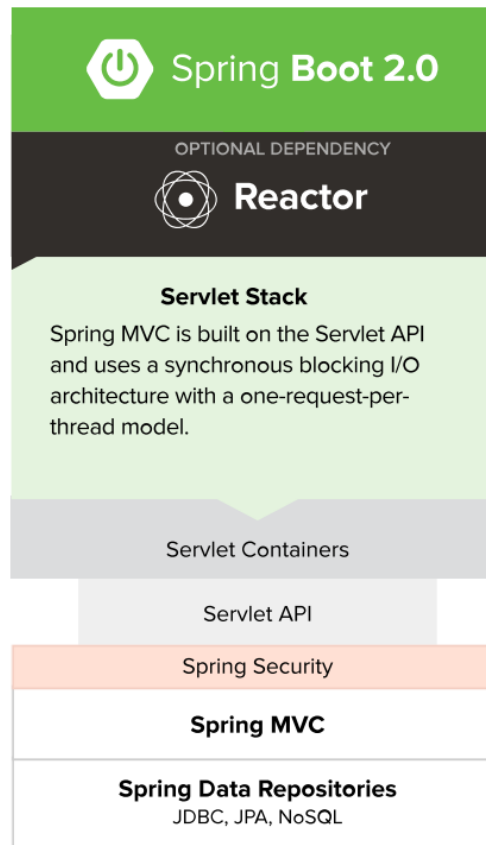


Figure 4.2: Spring Boot [Spr]

file beneath the `EnableAutoConfiguration` key where configuration classes are specified.

- **IDE support: Running, debugging, and profiling.** The term “integrated development environment” (IDE) refers to a software development environment. It is essential for a coder who enjoys using keyboard shortcuts and typing quickly. Code completion is included in most excellent IDEs, allowing users to write a few characters, hit tab, and have the desired code produced automatically. They also allow for simple formatting, easy access to documentation and debugging. In case Java or other statically typed languages are used, IDEs may be able to generate a lot of useful code code, such as getters and setters on POJOs, constructors, test classes, etc. Method references are also readily available.
- **Security.** Because of its connection with Spring Security, Spring Boot provides great security features by using the `spring-boot-starter-security` dependency. This provides HTTP Basic authentication out of the box.
- **JPA.** The ACID features of a typical relational database management system (RDBMS) ensure that transactions are completed reliably: atomicity, consis-



tency, isolation, and durability. RDBMS support is provided by databases like MySQL and PostgreSQL, which have significantly reduced database prices.

- **Liquibase.** Liquibase is a database source control system. It's an open-source project (Apache 2.0) that allows altering the database during the development or runtime process. It allows comparing the entities to the database tables and constructs migration scripts. It also allows entering default data that is separated by commas.
- **Deployment.** A Spring Boot client-server application may be installed on any device permitting the execution of Java applications. Spring Boot runs an embedded web server using a public static void main entry point. Spring Boot applications are packaged as a JAR that contains all of the required dependencies, such as the web server and start/stop scripts. The JAR is shareable and the application can be easily started because there is no need for a build tool, no setup, no web-server configuration, and so on.

### Client with Ionic/Angular

Ionic [Ion] is an open source UI toolkit for creating high-quality mobile and desktop apps using web technologies such as HTML, CSS, and JavaScript, as well as integrations for popular frameworks like as Angular, React, and Vue..

Ionic is a framework that focuses on an app's frontend UX and UI interaction — UI controls, interactions, gestures, and animations. It's simple to pick up and connects with other libraries and frameworks like Angular, React, and Vue. Alternatively, with a simple script inclusion, it can be utilized without any frontend framework.

Ionic is the first mobile app framework that allows web developers to create apps for all major app stores as well as the mobile web using a single codebase. Ionic apps also look and feel like at home on every device thanks to Adaptive Styling.

Ionic is designed to function smoothly on all of the latest mobile devices. Create lightning-fast apps with a minimal footprint and built-in best practices including as hardware-accelerated transitions, touch-optimized gestures, pre-rendering, and AOT compiling. Figure 4.3 shows a diagram of the performance score of ionic applications.

Ionic is built to operate and look great on all of today's mobile devices and platforms. One can create elegant applications using the ready-made components, typography, and an attractive (but extendable) base theme that adjusts to each platform.

Ionic is aiming to accomplish the following goals, stated in 4.1.

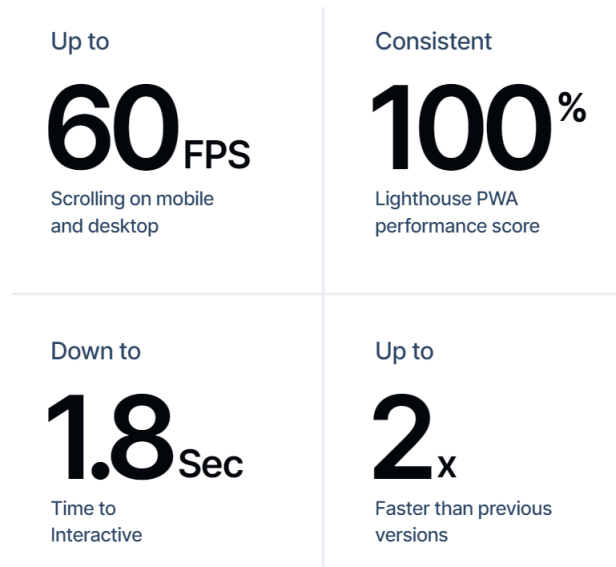


Figure 4.3: Ionic Performance Score [Ion]

Goal	Description
Cross-platform compatibility	Creating and deploying apps that run on many platforms, including native iOS, Android, desktop, and the web as a Progressive Web App, all from a same code base that can leverage a single code base and run anywhere.
Web Standards-based	Ionic is built on top of proven, standardized web technologies such as HTML, CSS, and JavaScript, as well as contemporary Web APIs like Custom Elements and Shadow DOM. As a result, Ionic components have a consistent API and are not subject to the whims of a single platform provider.
Beautiful design	Ionic means clean, straightforward, and practical functionalities. Ionic is built to work and look great on any platform right out of the box. Begin with pre-made components, typography, interactive paradigms, and a beautiful (yet extendable) basic theme.
Simplicity	Ionic is designed to be simple, making app development pleasant, simple to learn, and accessible to almost anyone with web development experience.

Table 4.1: Ionic Goals

While previous versions of Ionic were strongly tied with Angular, version 4.x was re-engineered to serve as an independent Web Component library with integrations for the most recent JavaScript frameworks, such as Angular. Most frontend frameworks, including React and Vue, can successfully use Ionic, albeit some frameworks require a shim for complete Web Component support.

The core of what makes Ionic amazing has always been angular. While the main components were written to serve as a stand-alone Web Component library, the `@ionic/angular` package makes Angular integration a simple. `@ionic/angular` interacts with essential Angular libraries, such as the Angular router, and provides all of the functionality that Angular developers would expect from Ionic 2/3.

Ionic is actively developed and maintained by a core team full-time, and its ecosystem is fueled by an international community of developers and contributors. Ionic is used by developers and businesses of all sizes to create and release fantastic apps that work on any device.

### **Leveraging APIs in application development**

API libraries provide thousands of APIs, and are essential in daily programming tasks [ZM19]. The following paragraphs analyze different APIs that can be used for handling specific tasks that can be included as needed in client-server applications providing features similar in concept to the tasks described by those APIs.

#### **Amazon S3 API for handling multimedia files**

Amazon Simple Storage Service (Amazon S3) [Ama] is a cloud-based object storage service with unrivaled scalability, data availability, security, and performance. Amazon S3 allows customers of all sizes and sectors to store and safeguard any amount of data for a variety of use cases, including data lakes, websites, mobile applications, backup and restore, archive, business applications, IoT devices, and big data analytics. You can use Amazon S3's administration tools to optimize, organize, and configure data access to meet your specific business, organizational, and compliance needs.

Since Amazon S3 service provides easy access to its data (through a URL as much as the client is involved), applications relying heavy on displaying multimedia files can benefit from the features that the service is providing, namely storing data, managing data stored, accessing data stored, logging and monitoring.

A common usage of Amazon S3 is to keep the S3 URLs in the database. Having the URL, it is easy to access the data that is needed. For updating or uploading data, Amazon S3 can be accessed by the means of Amazon S3 REST API, using a PUT Object.

A PutObject request adds an object to a bucket. One must have WRITE permis-

sions on a bucket to add an object to it.

Amazon S3, instead of adding fragmented items to buckets, if a success response is obtained, Amazon S3 adds the whole object.

### **Push API for handling notifications**

The Push API [Bev] allows a push message to be sent to a web application using a push service. Even if a web application or user agent is dormant, an application server can send a push message at any time. The push service ensures that information is delivered to the user agent in a timely and reliable manner. Push messages are sent to a Service Worker in the web application's origin, which can use the data in the message to alter local state or display a notification to the user.

When there isn't already an active communication channel between the user agent and the web application, push messaging is the ideal option. When compared to more direct ways of communication like Fetch Standard or websockets, sending push messages consumes significantly more resources. Push messages have a longer latency than direct communications and may be subject to use restrictions. The size and amount of push messages that can be transmitted are usually limited by most push systems.

The following presents how a push message is delivered from an application server to a web application:

- The application server instructs the push service to send a push message in accordance with [RFC8030]. The push endpoint specified in the push subscription is used in this request.
- The message is delivered to a specified user agent via the push service, which identifies the push destination in the message.
- The user agent detects and activates the intended Service Worker, then sends the push message to the Service Worker.

This framework enables application servers to activate a Service Worker in response to application server events. Information about those events can be included in the push message, allowing the web application to respond appropriately to them without having to make network calls.

The following and diagram 4.4 illustrate a hypothetical use of the push API.

### **Service specific API for handling business specific tasks**

A service can be available to its clients through an API. In this case, the clients would probably be developers, manipulating the specific service to further provide more robust functionalities (and yet simpler to use) to the final user.

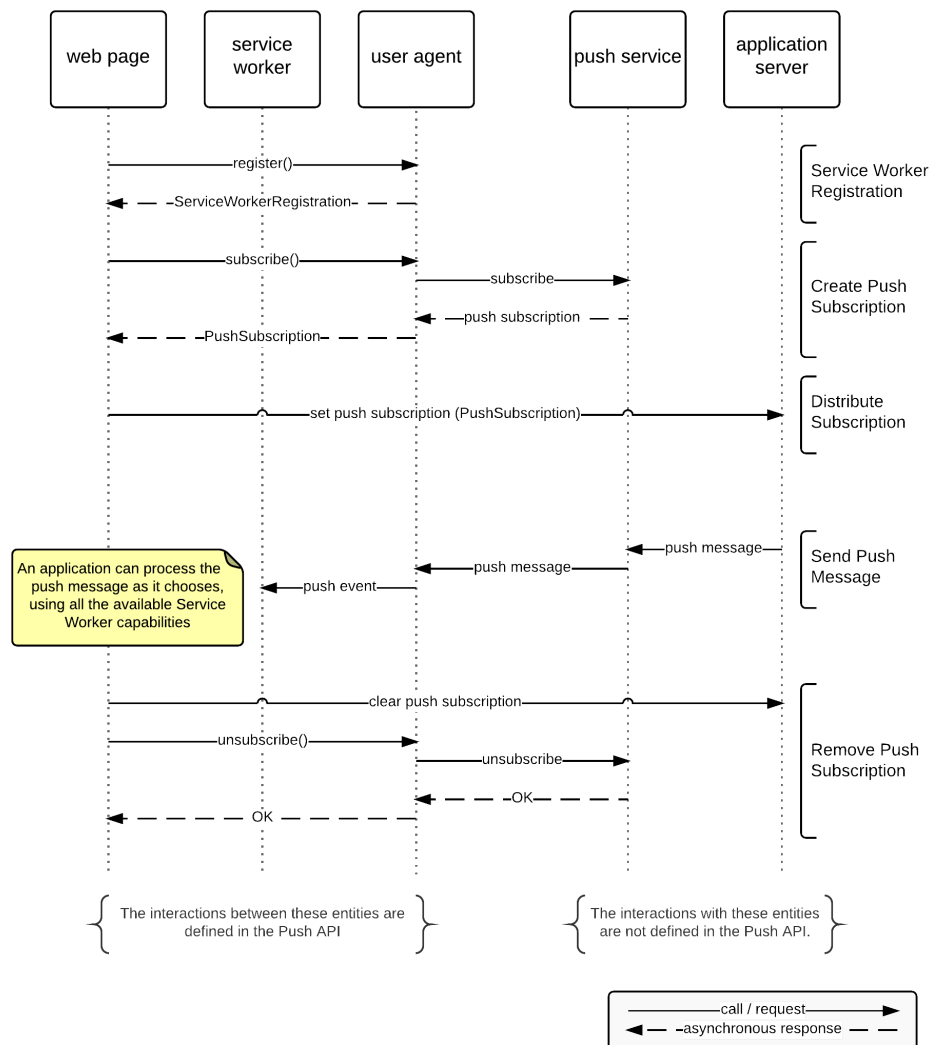


Figure 4.4: Example flow of events for subscription, push message delivery, and unsubscription [Bev]

One such example can be the Ebisu API, that allows clients to access the Ebisu service. Ebisu [Fas] is a public-domain library that answers two questions in the domain of memorizing facts:

- Which facts should be reviewed?
- What effect does a person's performance on a review have on the fact's future review schedule?

It's designed for software developers who want to make quiz apps, and it has a straightforward API for dealing with these two aspects of quiz scheduling:

- **predictRecall** gives the current probability of recalling a certain fact.
- **updateRecall** adjusts the chance of future recollection based on a quiz result.

The API provides access to the above stated functionalities of the system, acting as a communication channel between the user and the service, allowing the service to do more complex computation while providing the user with a simpler way of benefiting from the insides of the service.

This particular service provides access for multiple programming languages. For example, the Java port of the original Python implementation of Ebisu can simply be installed as a dependency using one of the two major building tools: Gradle or Maven. A code example for Maven can be seen below:

```
<repositories>
  <repository>
    <id>jitpack.io</id>
    <url>https://jitpack.io</url>
  </repository>
</repositories>
...
<dependency>
  <groupId>me.aldebrn</groupId>
  <artifactId>ebisu-java</artifactId>
  <version>-----Tag-----</version>
</dependency>
```

An example of how the ebisu API functions could be used is shown in figure 4.5.

```
EbisuModel previousModel = new EbisuModel(  
    ebisuCardModel.getAlpha(),  
    ebisuCardModel.getBeta(),  
    ebisuCardModel.getHalflife()  
);  
  
EbisuInterface updatedModel = Ebisu.updateRecall(previousModel,  
    successes: evaluation % (totalTrials + 1),  
    totalTrials,  
    hours);
```

Figure 4.5: Ebisu API functions

# Chapter 5

## E-learning System Development

This chapter presents the stages associated to the development of the proposed e-learning system. The process starts with the requirements gathering. These are further processed into usecases and analyzed from the point of view of software development. The next phases of software development are described, containing architecture and design, implementation and testing.

### 5.1 Requirements Gathering

This section describes the overall flow of the application from the users perspective. Alistair Cockburn [Coc00] is talking about this phase as if a bird would watch the users' interaction with the application, meaning that as long as the user is concerned, the internals of the application should be hidden.

#### Main Usage Narative

Below can be find a typical interaction with the application.

User wants to practice. He searches for a portal. He wants to practice a certain deck from that Portal. He can choose not to learn some cards in that Deck. The cards he is practicing in that Deck are added to his ToPractice Deck of that Portal. Whenever a card from that Portal's ToPractice Deck is about to be forgotten, the user is notified that he has cards to practice in that Portal. The user can learn Decks of multiple Portals on the same application. Whenever he has cards that he doesn't want to learn anymore, he can mark cards as SetAside. Those cards are removed from the ToPractice Deck of that Portal. He can also remove a whole deck from the ToPractice Deck. Whenever the user is Practicing a card, a new recallModel of that card is computed so that he can be informed about the next time he has to practice the card based on the previous times he had practiced it. Each Portal keeps a history of the cards learned by each user from where statistics can be performed.



A practice session is opened each time a user opens the ToPractice Deck of a Portal. Developers should be able to add statistics as time goes. Users can add cards in a Portal, but they are going to be visible by them only. A Teacher can create multiple portals. A teacher can create multiple decks in a portal and multiple cards in a deck. A teacher can share access to his portal through a link, or make it public. A User could start a game with other users in a portal.

### In/Out List

Scope, as Alistair Cockburn defines it in [Coc00] is the word we use for the extent of what we consider to be designed by us, as opposed to already existing or someone else's design job. Because Keeping track of the scope of the project, or even just the scope of a discussion can be difficult, one helpful tool for tracking and managing scope discussions is the In/Out List. It can be used to control scope discussions for ordinary meetings as well as project requirements. The way In/Out list was used for this particular project was to keep track of the scope of the application for its first version. Finally, the list contains the all the features, but it marks as In only those that will appear in the first version of the application. Below 5.1 is the diagram of the In/Out list.

Topic	In	Out
Registration		
Log In		
G-mail/Facebook Log In		
Learners Managing		
Portal Managing		
Decks Managing		
Cards Managing		
Review Cards		
Practice Session		
Add custom cards by non-teacher users		
Remove cards from practicing		
Add entire deck for practicing		
Remove entire deck from practicing		
Statistics Generation		
Review Reminder		
Game		
Eliminate cards from reviewing		

Table 5.1: In/Out List

### Actor-Goal List

The Actor-Goal list concept as found in [Coc00] names all the user goals that the system supports, showing the functional content of the system. Unlike the in/out list, which shows items that are both in and out of scope, the actor-goal list includes only the services that actually will be supported by the system. Below 5.2 is the project's actor-goal list:

Actor	Task Level Goal	Priority
Any	Authenticates to the system	
Teacher	Manages portal's content	
Teacher	Manages portal's access	
Teacher	Manages deck's content	
Teacher	Manages cards	
Learner	Reviews cards	
Learner	Manage custom cards	
Learner	Access statistics about learning performance	
Learner	Benefits from intelligent quiz review scheduling	
Learner	Plays game with other learners of same portal	
Learner	Eliminates cards from reviewing	

Table 5.2: Actor-Goal List (Priority increases as color gets closer to red)

### Use Cases

This subsection describes the use-cases obtained after taking into consideration the actor-goal and the in-out list, transforming the main narrative into features that can be applied to a minimum viable product.

#### Use Case 1. Authentication

**Narative.** User is opening the app (on smartphone or web). If he is not registered to the system, He can Register to the system with an email address. If he is registered but not authenticated, He can LogIn to the system. He does not have to Login every time he enters the app. A JWT Token will be used to authenticate the user instead.

**Success Guarantees** User has appropriate permissions within the application.

**Main Success Scenario:**

1. User authenticates to the application

**Extensions:**

1. User is not find within the application. User is redirected to registration page.

2. If the user has previously logged in, he will not be prompted to login again if the JWT token can be accessed.

### Use Case 2. Manage Portals

**Narative.** A Teacher can create, update, remove a portals. A Portal has a title, description, visibility, and a background image.

When a new Portal is created, a new entry will be added in the Portals Table in the database, containing the fields specified by the user and the owner id. If one of the Portal's fields is not valid, the portal will not be created and the user will be notified about the invalid fields.

**Success Guarantees.** Changes to portals persists in the application.

**Main Success Scenario:**

1. Teacher creates a new portal. Teacher sets title, description, background image and accessibility for the Portal to be created.
2. Teacher modifies or deletes an existing portal he owns. Teachers can modify the title, description, background image and accessibility

**Extensions:**

1. Teacher receives an error message if the creation was unsuccessful. The portal is not created.
2. Teacher receives an error message if the modification was unsuccessful. The changes will not persist in the database.

### Use Case 3. Manage Decks

**Narative.** A Teacher can create decks in a portal with a title, description, visibility, and a background image.

**Context.** User manages the decks of the portals he owns. User can create, modify, delete decks.

**Success Guarantees.** Changes to decks persist in the application.

**Main Success Scenario:**

1. Teacher creates a new deck in a portal.
2. Teacher sets the title, description and background image for the Deck to be created.
3. Teacher modifies or deletes an existing deck of a portal. Teachers can modify the title, description and background image.

**Extensions:**

1. Teacher receives an error message if the creation was unsuccessful. The deck is not created.
2. Teacher receives an error message if the modification was unsuccessful. The changes will not persist in the database.

**Use Case 4. Manage Cards**

**Narative.** A Teacher can add cards to a deck. To create a card, the teacher fills the front and back image the front and back text, the thumbnail and the title of the card

**Context.** Teacher manages the cards of a certain decks. Teachers can create, modify, and delete cards.

**Success Guarantees.** Changes to cards persist in the application.

**Main Success Scenario:**

1. Teacher creates a new card in a deck. Teacher sets the title and the resources of the card to be created.
2. Teacher modifies or deletes an existing card of a deck.

**Extensions:**

1. Teacher receives an error message if the creation was unsuccessful. The card is not created.
2. Teacher receives an error message if the modification was unsuccessful. The changes will not persist in the database.

**Use Case 5. Practice Cards**

**Narative.** A Learner can search portals and decks and choose to practice a deck. The learner can practice the cards one by one. As he practices cards, based on the evaluation, a recall model is computed for every card. Based on the recall model, the user can further tell when it is most likely to forget the card so he can practice it again.

**Context.** Learner chooses deck to practice.

**Success Guarantees.** Learner makes progress in learning a card.

**Main Success Scenario:**

1. Learner opens a deck he wants to learn.
2. Learner sees cards one by one and reviews them.
3. Learner access statistics.

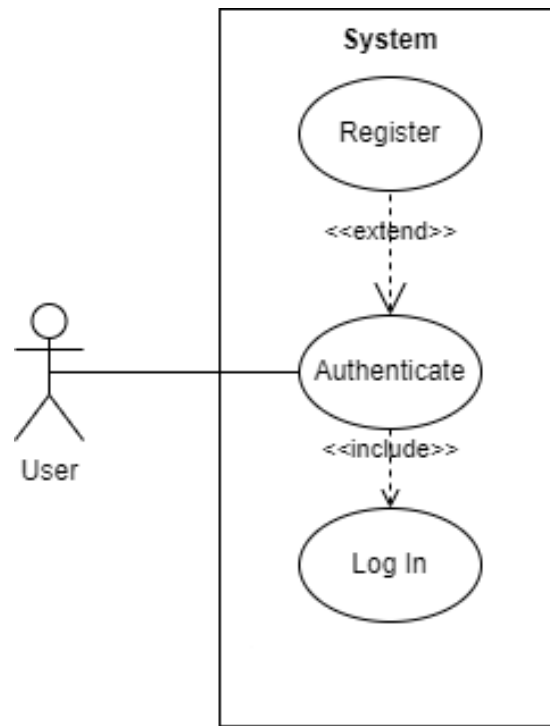


Figure 5.1: Regular User Use Case Diagram

## Use Case Diagrams

Below can be seen the use case diagrams for each type of users.

The use case diagram that includes both types of users, Teacher and Learner represents the Authentication, use case 1 5.1.

The use case diagram corresponding to Teacher type of user shows use cases related to managing the resources of the application and can be seen in 5.2.

The Learner's use case diagram that can be seen in 5.3 shows a visual representation of the Practice Cards Use case.

## 5.2 Analysis

This section explains the process of mapping the features stated in natural language to an actual system of interrelated classes.

Each class with the exception of the Enumeration classes also have an id besides all the attributes specified.

The first use case involves users Authentication.

User class is representing the security part 5.4 that involves a user: login, password, email address, and authorities. A User has a one-to-one relationship with an AppUser. This abstraction is added to separate a security user from an application user that can be so manipulated to add features related to the business of the appli-

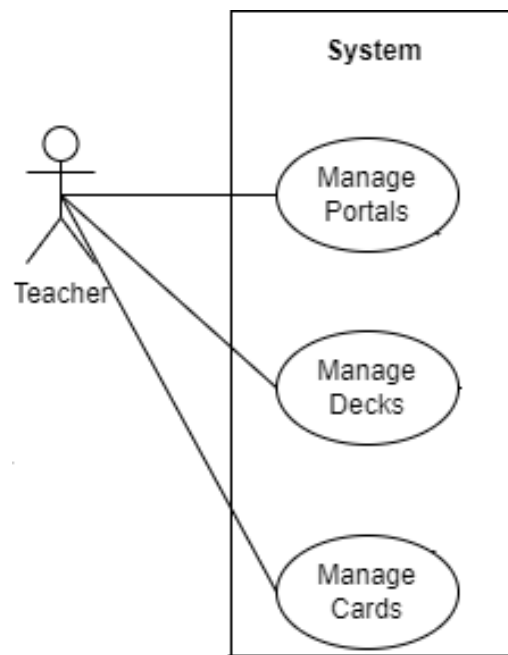


Figure 5.2: Teacher Use Case Diagram

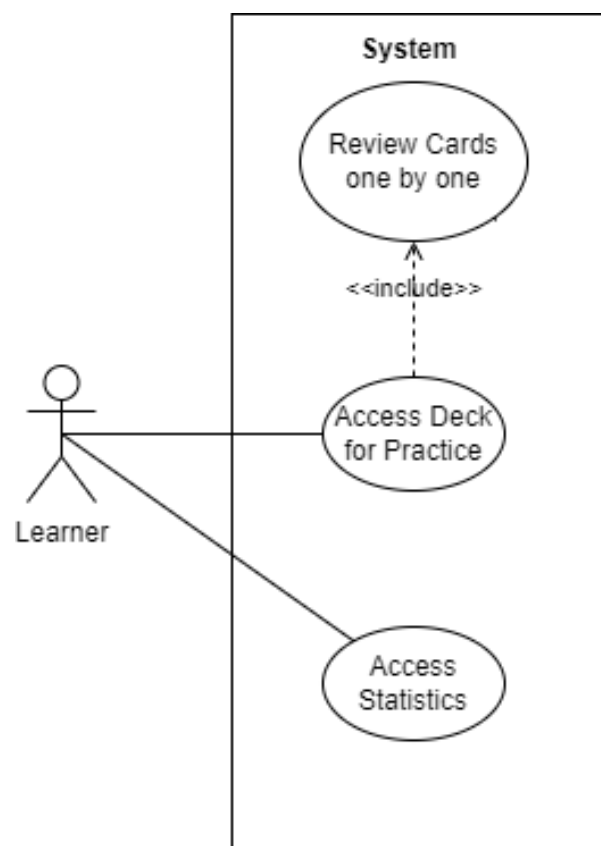


Figure 5.3: Learner Use Case Diagram

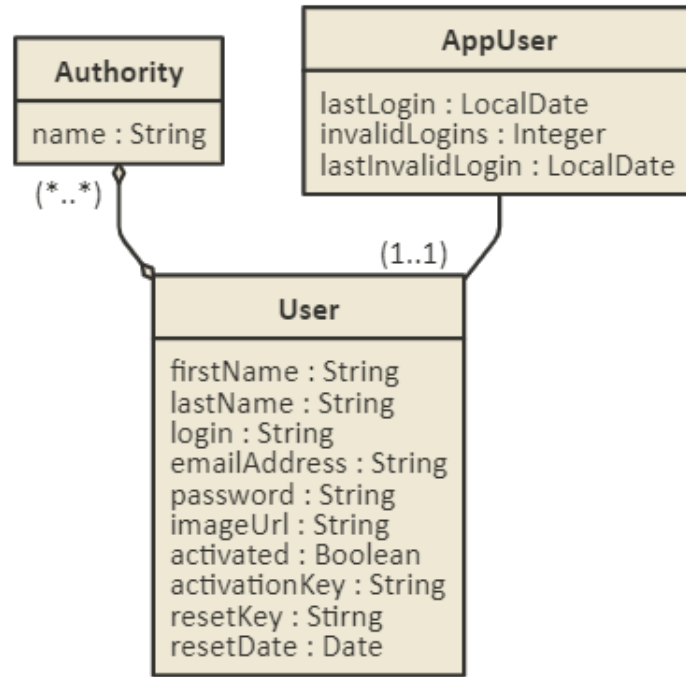


Figure 5.4: Security Analysis

cation avoiding the danger that could arise by implying the Security user directly. One such new feature could be adding badges or characters to a user.

Further, is presented the content managing analysis 5.5. The AppUser is linked to Portals with a one-to-many relationship representing a Teacher owning multiple portals, and with a many to many relationship representing multiple learners that can follow multiple portals.

Between Portal and Deck there is a one-to-many relationship involving a portal could contain multiple decks. A similar relationship maps Deck and Card.

Reviewing Process is visually described in 5.6.A UserCard can have multiple CardModelHistory elements that correspond to it. A CardModelHistory element represents a review of a certain card by a certain user. It contains the elements that an intelligent quiz scheduling system is needing for providing accurate results. By keeping a history of card reviewing, generating statistics is also tackled.

To give the possibility of using different quiz scheduling services, each Card-ModelHistory has a CardModelType that links to a class holding the elements necessary to a particular implementation of a quiz scheduling service (ex. Ebisu).

### 5.3 Architecture and Design

The following section describes the architecture of the system and ways of how it can be improved by means of organization, scalability and availability.

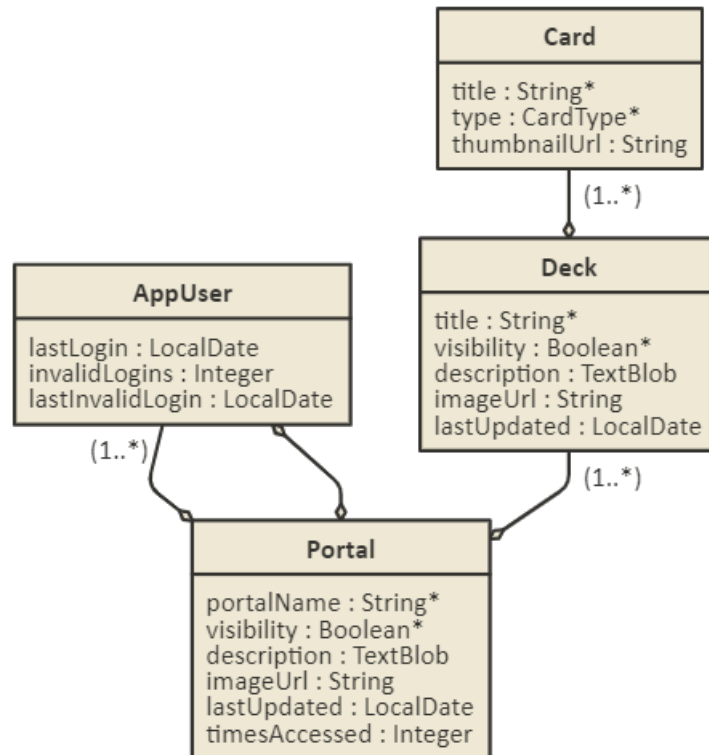


Figure 5.5: Content Analysis

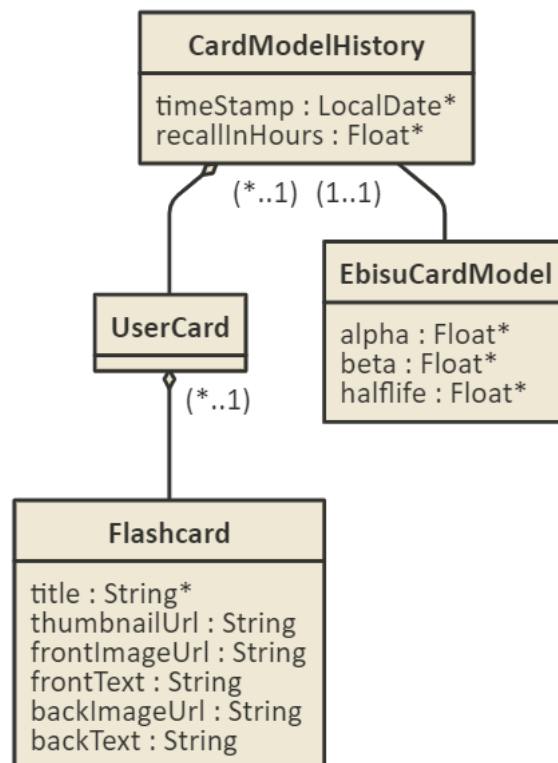


Figure 5.6: Reviewing Process Analysis



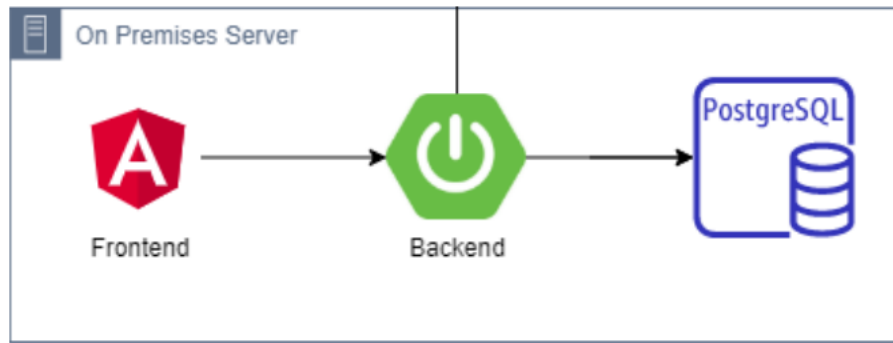


Figure 5.7: On Premises Server Architecture (Development Stage)

The initial architecture 5.7 of the system is represented by an on-premises server that runs locally and contains all the computational elements: Front-End, Back-End and Database.

The Front-end application is running on a development server which is booted up on the localhost by ionic through ionic CLI command `ionic serve`.

The Back-End component is a Spring application. Spring Boot uses a public static void main entry point that launches an embedded web server. Spring Boot applications are embedded in a JAR, which includes all necessary dependencies including the web server and start/stop scripts. The `.jar` can be distributed to any machine that can run java applications without the overhead of build tools or setup or specific web-server configuration. The simplicity should reduce to running the following command `java -jar appName.jar`.

The database component, which is a local PostgreSQL instance, can be accessed only locally and is exclusively used for development purposes. The motivation behind running a local Postgres instance in the development stage is the reduce management complexity, low cost, no network configuration.

While the simplicity of the above architecture is convenient in the development stage because of the no-cost tiers and the minimum amount of configuration needed, it is not suitable for hosting a real-life application accessible through internet from all over the world.

Another proposed approach is to use Amazon AWS Cloud Services to deploy the application to the cloud. An application deployed on cloud, can run in a public subnet of a virtual private cloud. AWS is providing the application running environment by means of an EC2 instance. The advantages are that the hardware is provided by the cloud provider and cloud services can be further used to improve security, availability and the scalability of the application. The new architecture can be seen in diagram 5.8.

In terms of security, the Database instance could be hosted in a private subnet that accepts traffic only from the Beck-End Servers. By doing so, scalability can

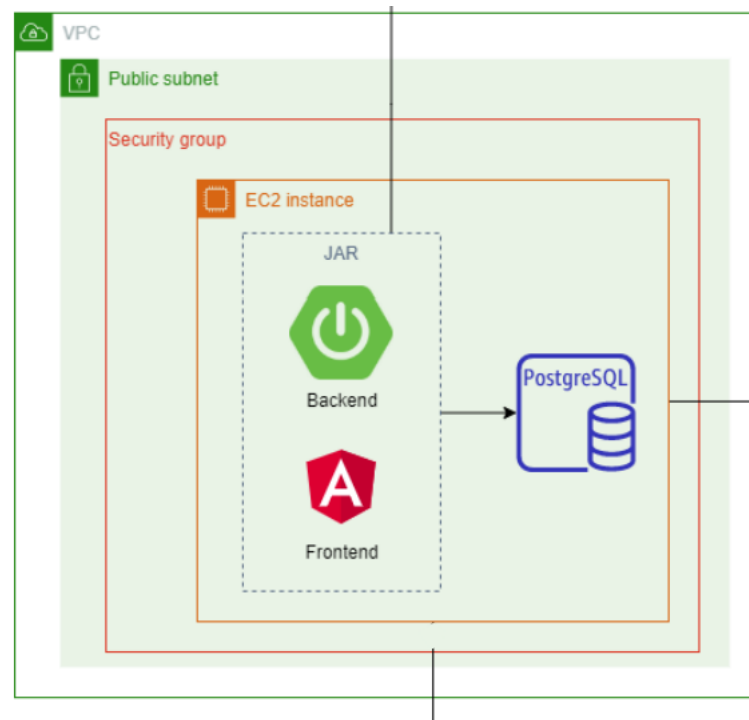


Figure 5.8: AWS Cloud Application Architecture

be improved by increasing the number of EC2 instances that run the application, each pointing to the same database instance. But rather than manually managing the number of active instances, AWS services as Auto-Scaling Group and Load Balancer could be used to automate the process. The new architecture is described in diagram 5.9.

The application tier serves both, Back-End and Front-End components. While this works fine for many cases, it adds unnecessary load on all components of the system, which also leads to unnecessary charges (due to increased traffic and resource usage). Since the apparition of UI frameworks of today (react, angular, vue) that generate static files (HTML, CSS, JS) and load the data from the back end (usually through HTTPS requests) at runtime, they do not require high computational resources, and they can be separated, using suitable AWS services such as S3, to concentrate the computing resources on processing data (Back-End). An illustration of this concept can be seen in diagram 5.10.

The diagrams 5.8, 5.9, 5.10 and the steps taken to improve the security, availability and accessibility of a traditional application using AWS are explained in more details in [Gre].

## Class Diagrams

The class diagrams are presented on sections that tie together classes that are used together to provide certain features. The main sections are Security, Managing Con-

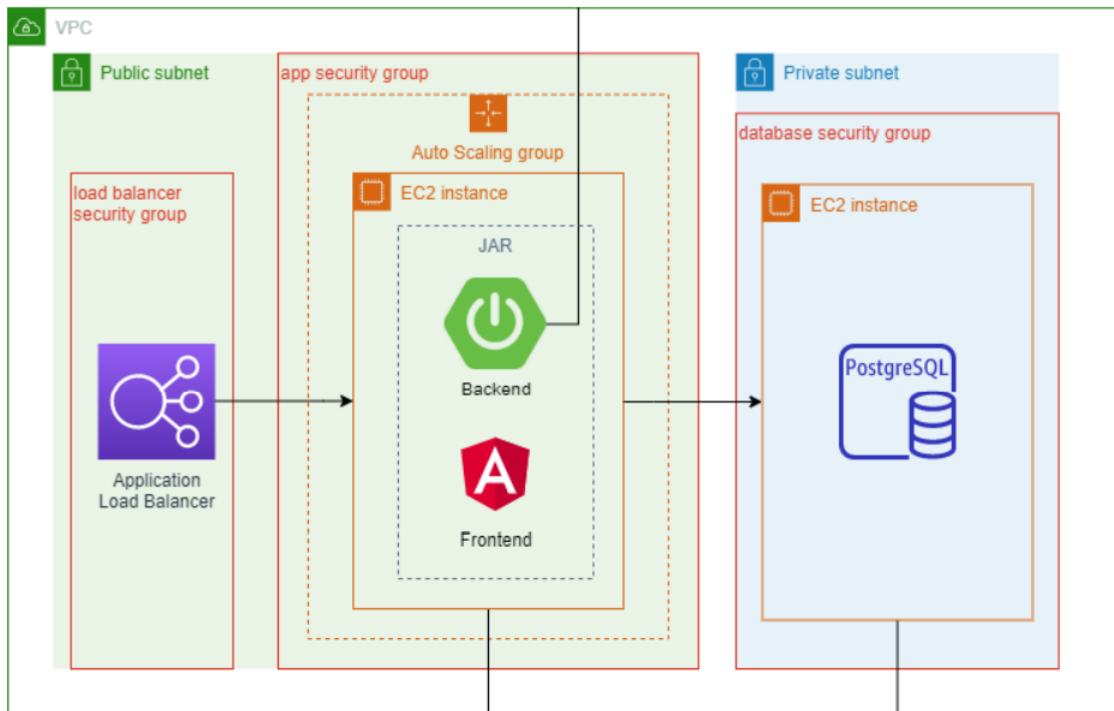


Figure 5.9: AWS Architecture with Separate Instance for Database

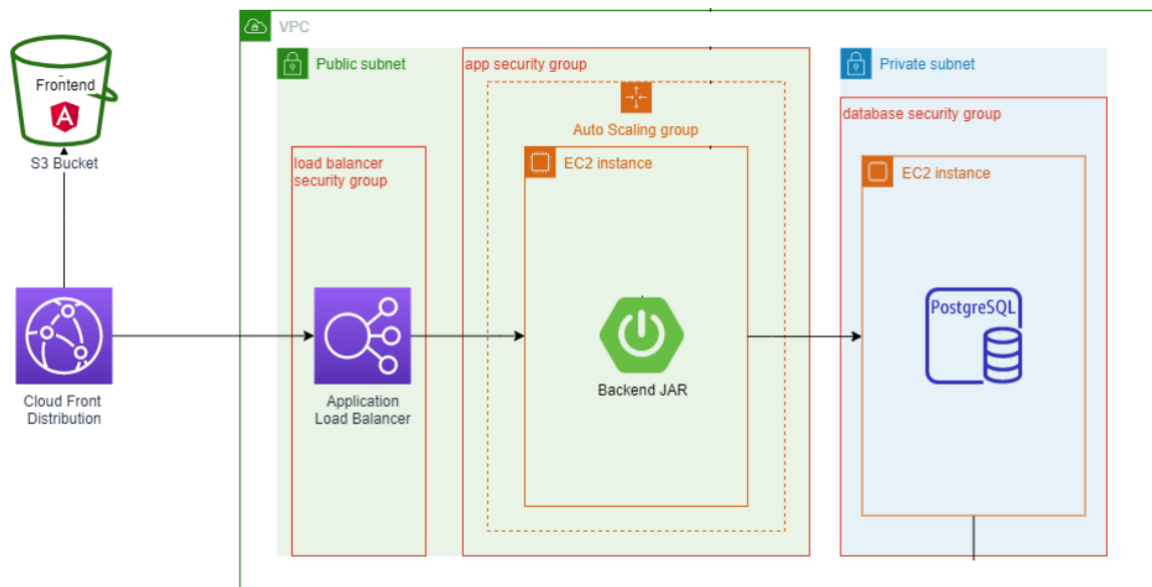


Figure 5.10: AWS Architecture with Separate Front End

tent, Organizing Cards and Reviewing Process.

The class diagrams for the Security section can be found in diagram 5.11. It states the relationship between a Security User (`User Class`) and an Application User (`AppUser Class`) and the relationship between a Security User and the `Authority Class`, used to specify the accessibility (authority) of a certain user over certain resources.

The class diagrams for the Content Managing section can be found in diagram 5.12. It represents the usability of the `AppUser Class` in separating the Security User from the Application user to increase security and decrease coupling. The relationships that can be observed are Many-to-One relationships (ex: `Deck` has a `Portal`). The motivation behind the choice of an unidirectional Many-to-One association instead of a One-to-Many association is that Hibernate might create unexpected tables and execute more SQL statements in the latter case [Tho].

The class diagrams for the reviewing process section can be found in diagram 5.13. The architecture facilitates adding new types of Quiz Scheduling Models as well as the generating statistics based on user's review history.

### Sequence Diagrams

This section presents the sequence diagrams for some of the application's features.

The sequence diagram for the Authentication feature is represented by diagram 5.14 and it includes the Log In and Register phases. In the Log In phase, the system checks the credentials by consulting the database. If the credentials are valid, the user receives a JWT token that can be used so that to spare the user from the necessity of Logging in to the application every time is accessed. If the credentials are not valid, the user is redirected to the Registration page where he can register new credentials to use for subsequent access to the application.

Below, in diagram 5.15, is described the feature of practicing cards. It consists of the user selecting a Deck to practice. The System interrogates the Database for the `UserCards` of a certain Deck and retrieve the response to the Learner. The Learner reviews the cards one by one. For each card, The system is recomputing the IS (Intelligent Scheduling) Model.

In the diagram below 5.16 is described the process of creating Portals. A similar process is involved in creating Decks and Cards. A Teacher initiates the creation of a Portal. The system validates the portal. A valid portal is saved to the database, and the Teacher receives a message whether the portal was saved or not to the database. If the portal is invalid, the Teacher receives a message that the portal is invalid.

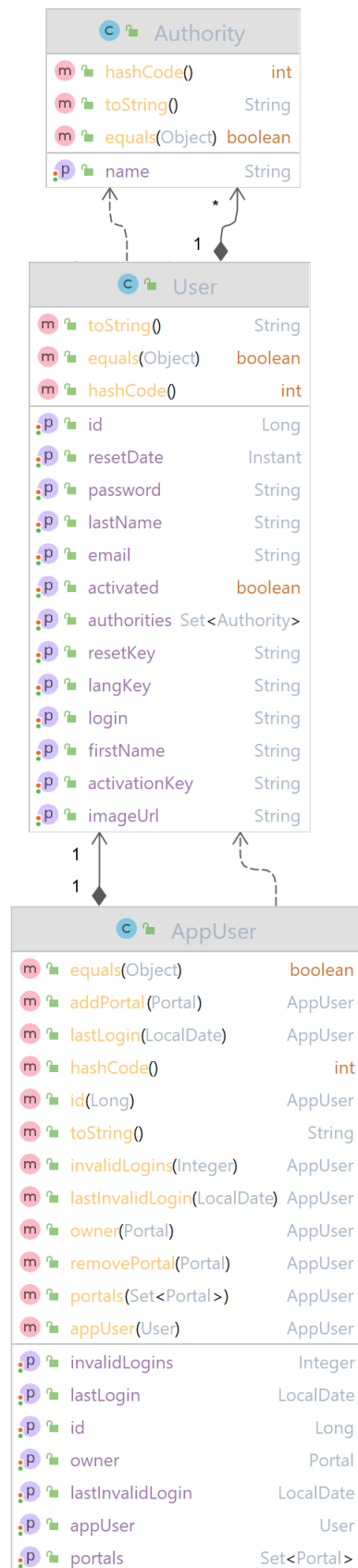


Figure 5.11: Security Class Diagram

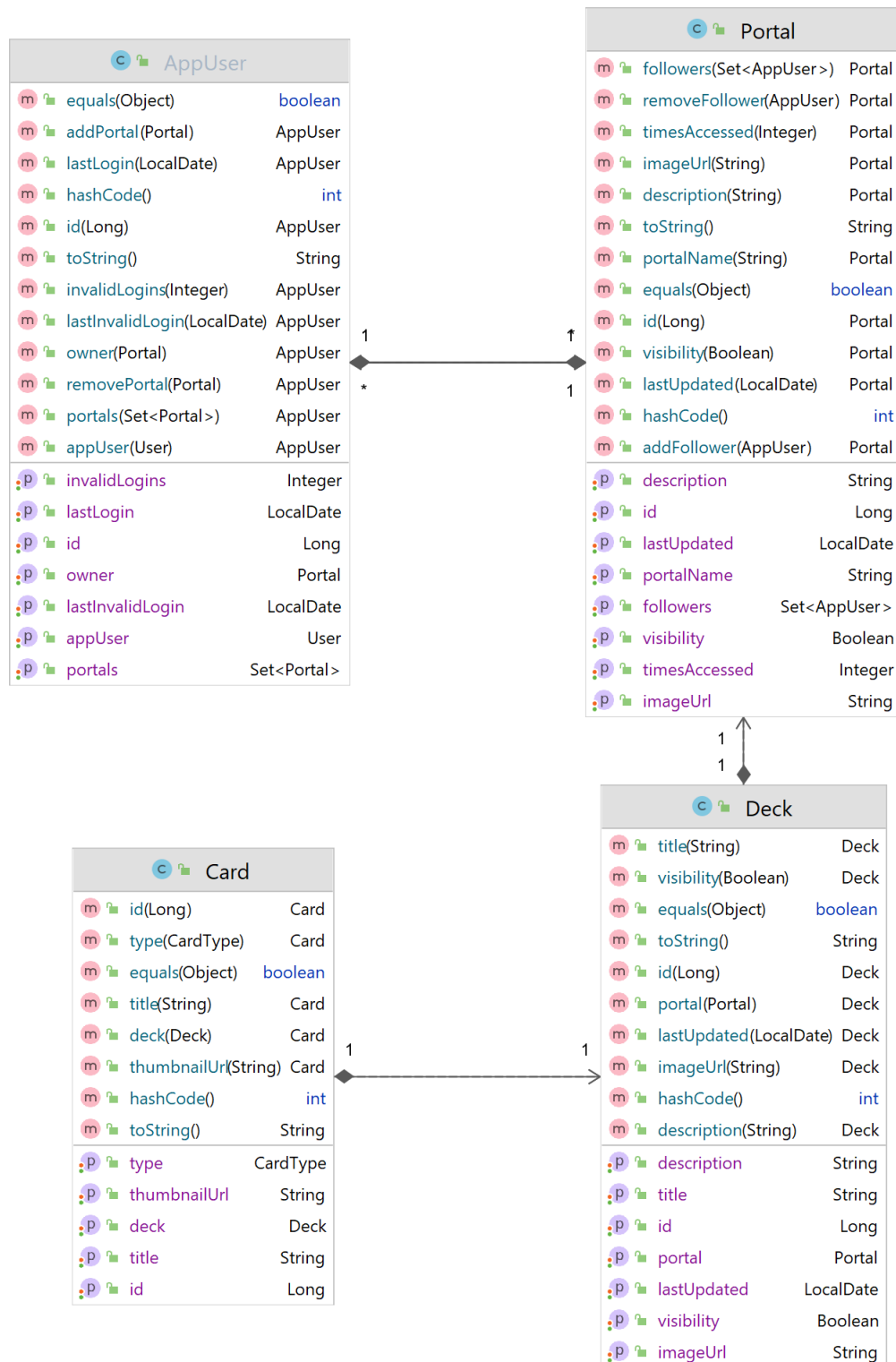


Figure 5.12: Content Managing Class Diagram

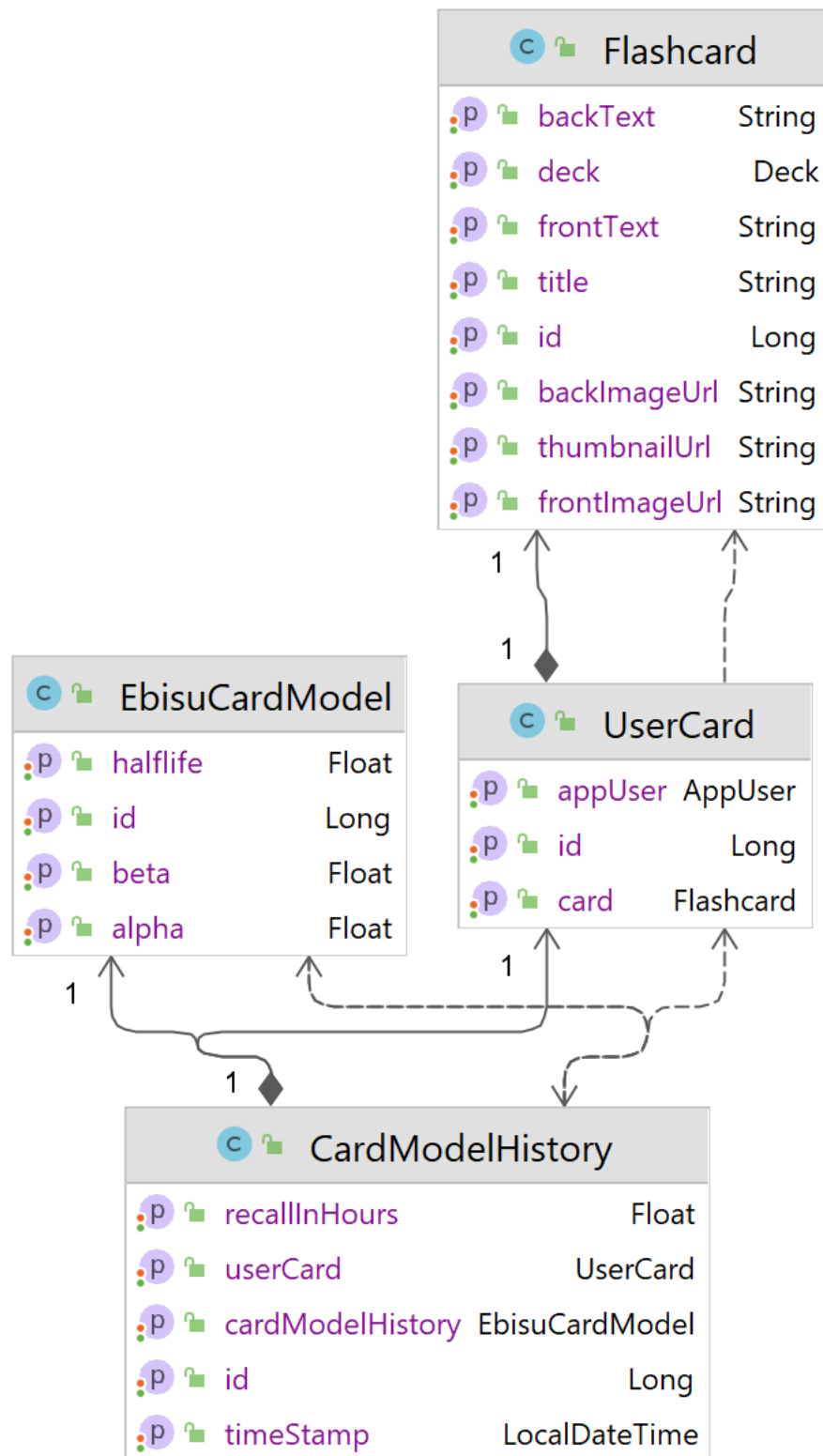


Figure 5.13: Reviewing Process Class Diagram

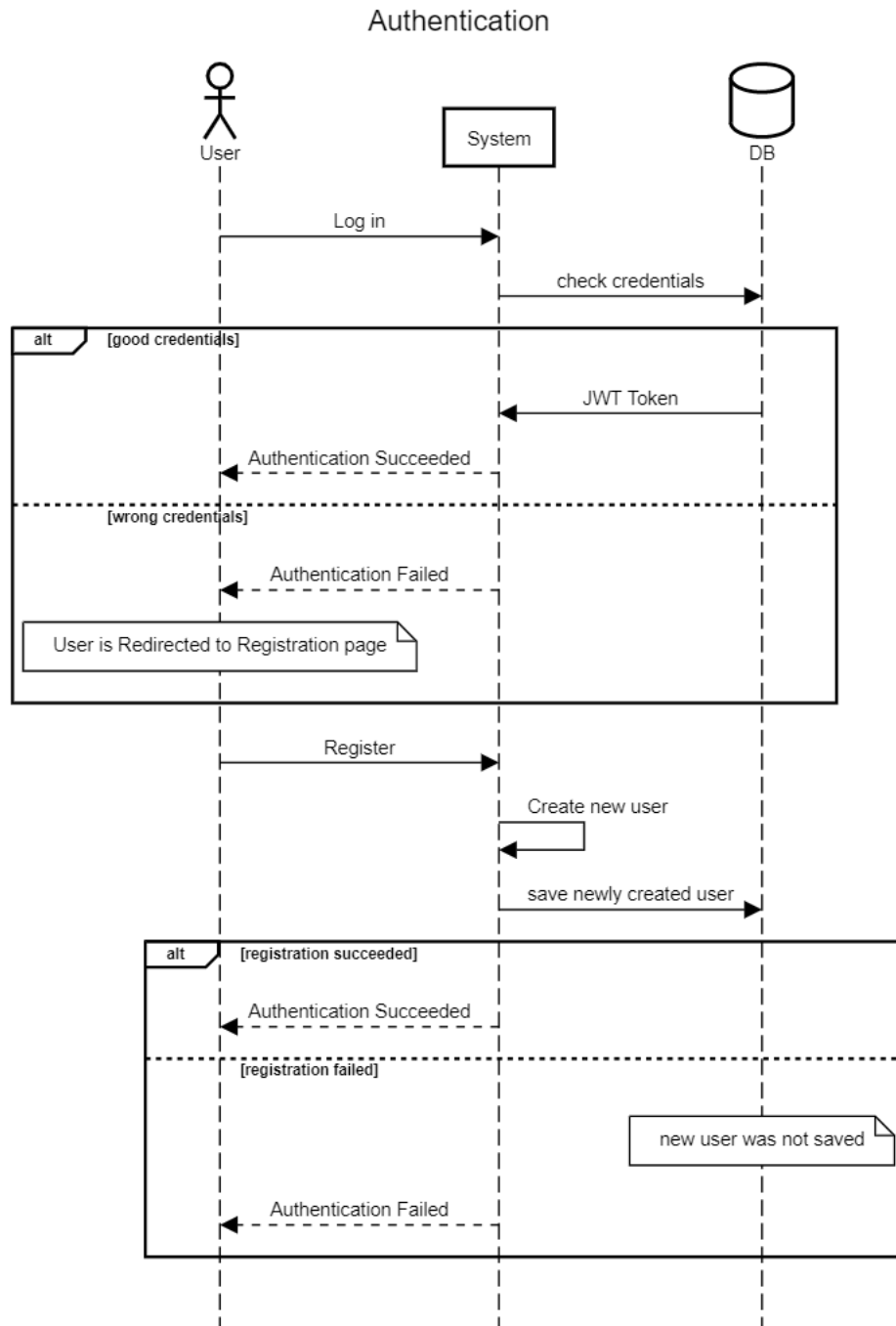


Figure 5.14: Authentication Sequence Diagram



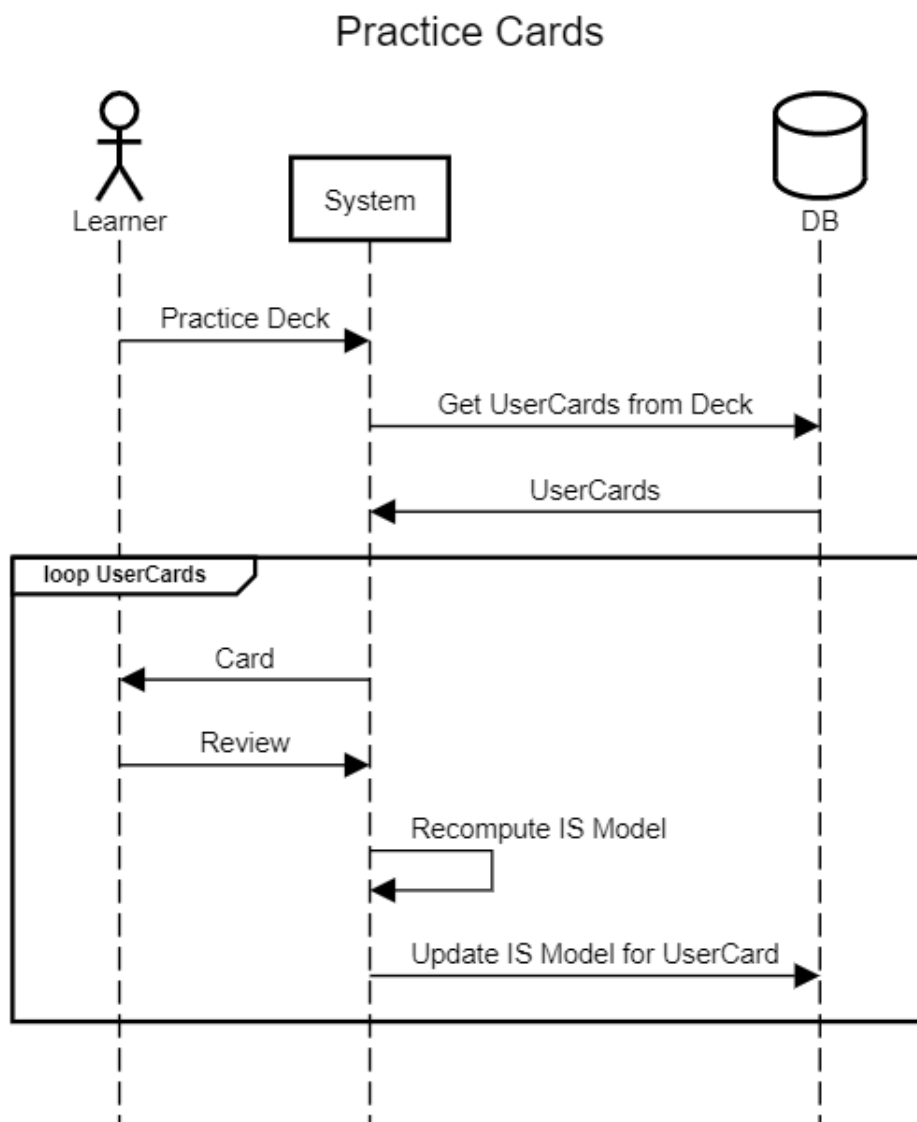


Figure 5.15: Practice Cards Sequence Diagram

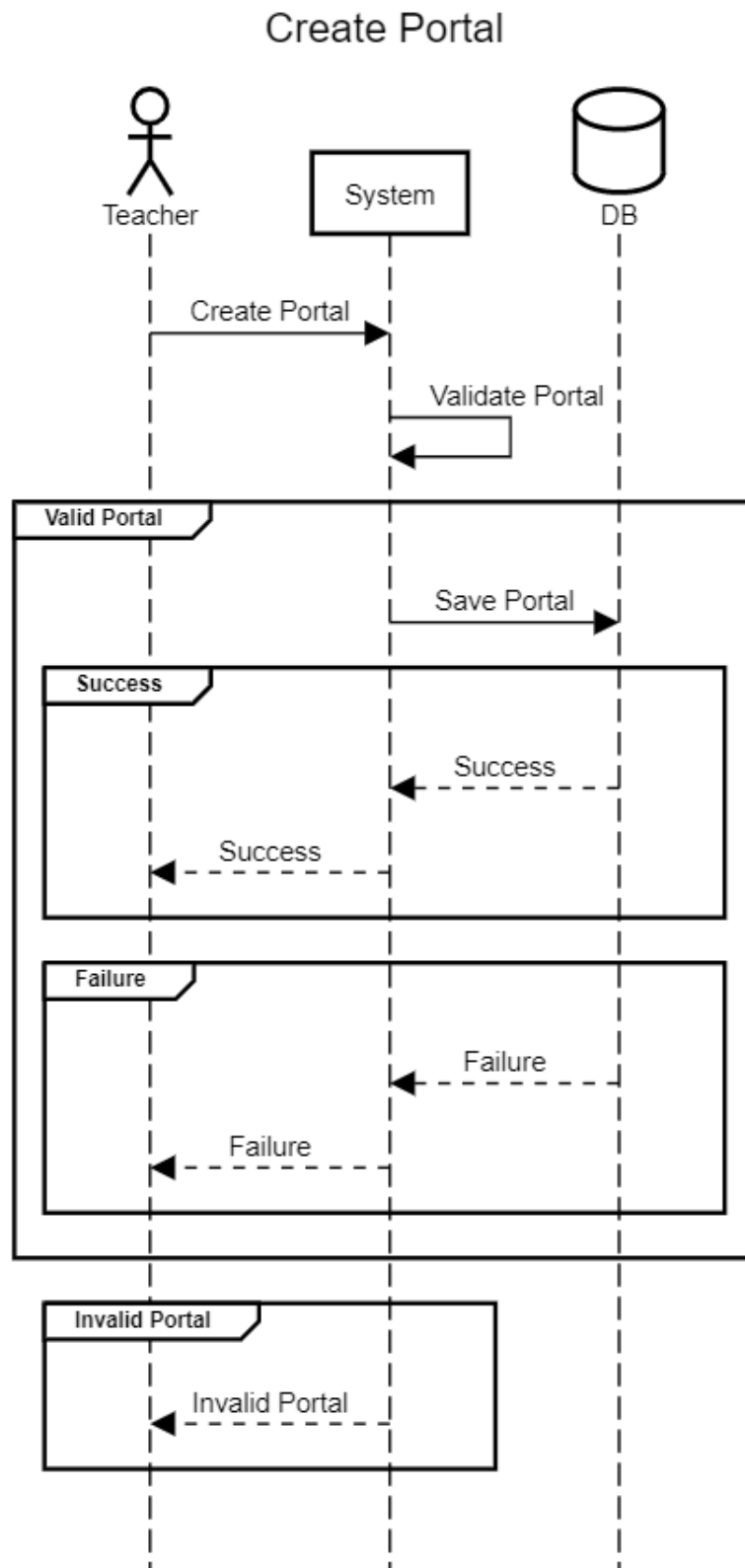


Figure 5.16: Create Portal Sequence Diagram

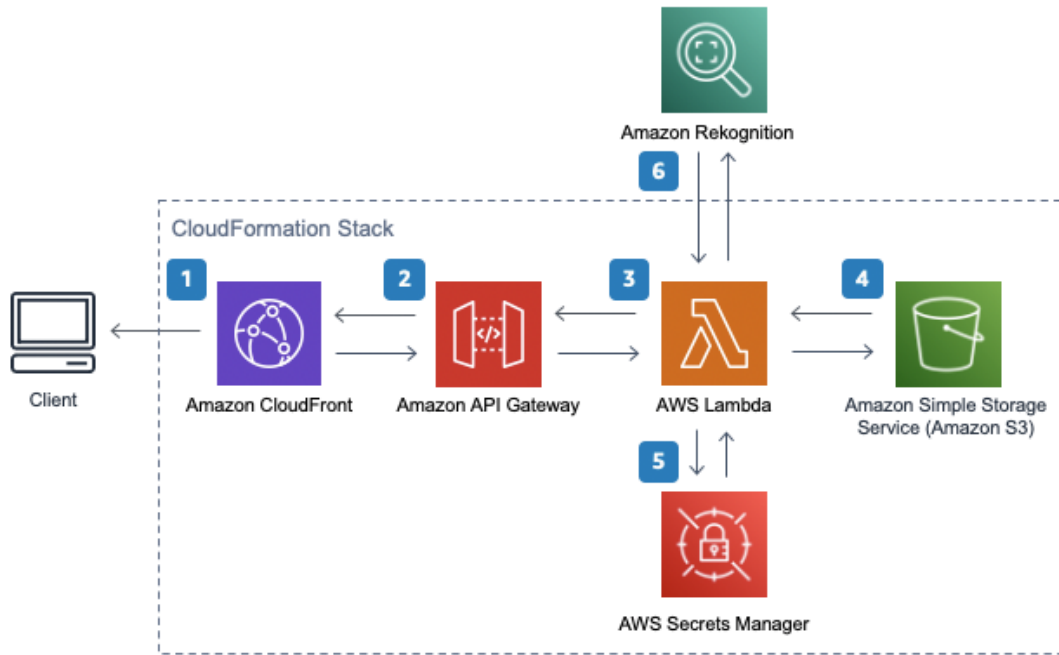


Figure 5.17: Serverless Image Handler Architecture

## 5.4 Implementation

The following section details some of the components of the application, together with code snippets of their usage and integration.

### S3 Serverless Image Handler

AWS proposes a solution for embedding images on websites and mobile applications to drive user engagement. The solution is called "The Serverless Image Handler".

Using this solution, developers can create public apps that with the ability to dynamically alter or manipulate their public images. This way, the solution consists of a publicly accessible, unauthenticated Amazon CloudFront distribution and Amazon API Gateway endpoint in the aws account, allowing public access.

A diagram of the architecture can be seen in figure 5.17.

The components of the Serverless Image Handler architecture are described below:

- *Amazon CloudFront* distribution with a caching layer to reduce image processing costs and associated image delivery latency. The image handler API is accessible via the CloudFront domain name, which is cached.
- *Amazon API Gateway* helps in initiating the lambda function described next and to provide endpoint resources.

```
aws : {  
  region: 'eu-west-1',  
  identityPoolId: 'ar:aws:iam::111111111111:identitypool/11111111-1111-1111-1111-111111111111'  
},
```

Figure 5.18: Configuring Identity Pool Id for Amazon Cognito

- *AWS Lambda Function* retrieves an image from the Amazon S3 bucket containing the images and applies the requested modifications to it before returning it to the API Gateway using Sharp.
- *S3 Bucket* for storing the images storage. Another S3 bucket is used for logging purposes
- *Amazon Rekognition* can be used with the purpose of being triggered by the Lambda function to analyze the images and delivers the results when utilizing the smart crop or content moderation capabilities. For an e-learning system, it may be useful, for example, to validate the images against inappropriate content.

The S3 bucket containing the images is protected *Amazon Cognito* such that it can only be accessed by the systems that are authenticated with the Identity pool Id (figure 5.18).

Further, using the Identity Pool Id, one can obtain credentials for the browser script and initiate the authorized S3 Buckets (code snippet in figure 5.19).

An example of image request can be seen in 5.20. To access the feature for processing images, the image request contains an attribute called `imageEdits` which further contains the requested processing over the image.

### Serverless Image Handler Cache

To further reduce cost, alongside AWS CloudFront, the application also uses a local cache. This way, the requests to the Serverless Image Handler can be kept under control. The images can be stored in the local storage for a certain amount of time before expiring, and fetch them without hitting the AWS solution. A code snippet of how the local cache works can be seen in figure 5.21.

### Intelligent Quiz Scheduler

This section presents the intelligent scheduling model used by the application for the Practice Cards feature.

```
export class S3Service {
  private buckets: Map<string, S3>;

  constructor() {
    this.buckets = new Map<string, S3>();

    AWS.config.region = environment.aws.region;
    AWS.config.credentials = new AWS.CognitoIdentityCredentials( options: {
      // eslint-disable-next-line @typescript-eslint/naming-convention
      IdentityPoolId: environment.aws.identityPoolId
    });
  }

  public initBucket(s3BucketName: string): S3 {

    let bucket = this.buckets.get(s3BucketName);
    if (!bucket ) {
      bucket = new S3({apiVersion: '2006-03-01'...});
      this.buckets.set(s3BucketName, bucket);
    }
    return bucket;
  }
}
```

---

Figure 5.19: Initiate S3 Bucket

```
const listImagesRequest = {
  s3BucketName: album.s3BucketName,
  albumTitle: album.title,
  imageEdits: {
    resize: {
      width: 200,
      height: 200,
      fit: ImageHandlerFit.COVER
    },
  },
}
} as ListImagesRequest;
```

Figure 5.20: Image Request

```
public getObservable<T, TResult>(
    request: T,
    cachedFunction: (r: T) => Observable<TResult>
): Observable<TResult> {
    const result: TResult = this.loadFromCache<T, TResult>(request);
    let observable: Observable<TResult>;

    if (result) {
        observable = of(result);
    }
    else {
        observable = cachedFunction(request).pipe(share());
        observable.subscribe( next: (r: TResult) => {
            this.saveToCache<T, TResult>(
                request,
                r,
                this.settings.expiresInSeconds
            );
        });
    }
    return observable;
}
```

Figure 5.21: Local Cache Request Intercepting

```
allprojects {  
    repositories {  
        maven { url 'https://jitpack.io' }  
    }  
}
```

Figure 5.22: JitPack Dependency

```
implementation 'me.aldebrn:ebisu-java:v2.0.0'
```

Figure 5.23: Ebisu Dependency

Ebisu [Fas] is a public-domain library that answers two questions in the domain of memorizing facts:

- Which facts need reviewing?
- How does the performance of one memorizing the facts on a review change the fact's future review schedule?

With such a system, quiz applications can move away from “daily review piles” caused by less flexible scheduling algorithms. For instance, a student might have only five minutes to study today; an app using Ebisu can ensure that only the facts most in danger of being forgotten are reviewed.

To integrate Ebisu in a java system using gradle, one must first use the `JitPack` repository as in 5.22.

Further, the actual Ebisu dependency 5.23;

Using Ebisu means creating a memory model for each card. Each time a learner wants to learn a new card, a new entry in the `UserCard` table along with a `CardModelHistory` entry for that `UserCard`. For an existing `UserCard`, an updated `CardModelHistory` will be added to the database with a new timestamp. The code describing how `UserCards` obtain an updated recall model is described in figure 5.24.

## 5.5 Testing

As [MSBT04] defines the process of software testing, it is designed for making sure that computer code does what it was designed from the beginning to do and at the same time it does not do anything it was not intended to. At the same time, they express the need of software being predictable and consistent, as offering users no surprises. Some techniques that could be used to achieve this goals are described below: Unit testing, Integration testing and end-to-end-testing.

```
private void updateCardModelHistoryOfUserCards(List<UserCard> userCards,
                                              Map<Long, Integer> evaluations) {
    userCards.forEach(userCard -> {
        CardModelHistory cardModel = cardModelHistoryRepository
            .findFirstByUserCardEqualsOrderByTimeStampDesc(userCard);

        EbisuCardModel ebisuModel;
        if (cardModel == null) {
            ebisuModel = EbisuCardModel.defaultModel();
        } else {
            ebisuModel = EbisuCardModel.updateModel(cardModel,
                evaluations.get(userCard.getCard().getId()));
        }
        ebisuModel = ebisuCardModelRepository.save(ebisuModel);

        CardModelHistory newCardModel = new CardModelHistory();
        newCardModel.setUserCard(userCard);
        newCardModel.setTimeStamp(LocalDateTime.now());
        newCardModel.setCardModelHistory(ebisuModel);
        newCardModel.setRecallInHours(ebisuModel.getHalflife() / 2);

        cardModelHistoryRepository.save(newCardModel);
    });
}
```

Figure 5.24: Update CardModelHistory for UserCard



## Unit Testing

Unit testing also known as Module Testing is the process through which the code that is being tested consist of subroutines, individual subprograms, or procedures. This means that the testing has a first focus on the smaller building blocks of the program and testing the program as a whole is deferred to a later stage.

## Integration Testing

Along with individual module testing, there also exists the need of testing the manner in which the modules are combined to form a working program.

One example of integration testing can be seen in testing how the `web.rest` module integrates with the `repository` module and further with the database.

Figure 5.25 provides such an example, where it is tested whether receiving an HTTP request for creating a card will propagate correctly to the database. Since the test is involving the database, it is important that every transaction that is executed during the test to be rolled back. This could be achieved by annotating the test suite with Spring's `@Transactional` annotation. The meaning of the annotation in this context is that every transaction in the test suite, regardless of it's outcome, will be rolled back at the end of the test method.

The test memorizes the size of the card table in the before effectuating the transaction. Further, Using a mocked user and a mocked rest service, the test effectuates the transaction and expects a positive response. With the positive response received, the database is queried again to validate the outcome of the transaction. Finally, Spring framework rolls back the transactions so the database is not affected.

## Cypress End-to-end testing

The JavaScript automation testing framework, Cypress [Mwa21], was especially created for performing front-end testing. One of its advantages is the way it re-invents how software testing is carried out, with an accent on web applications. As opposed to Selenium WebDriver, one of the most used testing frameworks, Cypress has a better performance in seconds as it runs in the browser and it is also known for its simplicity to use.

An example of end-to-end test managed by cypress can look like in the figure 5.26

```

@Test
@Transactional
void createCard() throws Exception {
    int databaseSizeBeforeCreate = cardRepository.findAll().size();
    // Create the Card
    restCardMockMvc
        .perform(post(ENTITY_API_URL).contentType(MediaType.APPLICATION_JSON)
            .content(TestUtil.convertObjectToJsonBytes(card)))
        .andExpect(status().isCreated());

    // Validate the Card in the database
    List<Card> cardList = cardRepository.findAll();
    assertThat(cardList).hasSize(databaseSizeBeforeCreate + 1);
    Card testCard = cardList.get(cardList.size() - 1);
    assertThat(testCard.getTitle()).isEqualTo(DEFAULT_TITLE);
    assertThat(testCard.getType()).isEqualTo(DEFAULT_TYPE);
    assertThat(testCard.getThumbnailUrl()).isEqualTo(DEFAULT_THUMBNAIL_URL);
}

```

Figure 5.25: Integration Test

```

describe('Authenticate', specDefinitions: () => {
    const username = 'admin';
    const password = 'admin';
    const root = 'http://localhost:8100';

    it('logs in as admin', assertion: () => {
        cy.visit(root);

        cy.get('#signIn').click();

        cy.url().should('include', value: '/login');

        cy.get('input[name="username"]').type(username);
        cy.get('input[name="password"]').type(password);

        cy.get('#login').click();

        cy.url().should('include', value: '/tabs/home');
    });
});

```

Figure 5.26: Authenticate E2E test

# Chapter 6

## Conclusions and Future Work

This chapter summarizes the purpose of the paper and presents the possible future paths of the application presented at Chapter 5.

As of the beginning of web 2.0, E-learning Systems have started to develop into the large pool of applications we see today. However, not all applications fulfill the purpose for which they were created, meaning that the needs of the target user, the learner, are not entirely addressed.

This paper brings a remedy to this problem by providing specific guidelines, aspects and attributes that could be used in the development process of asynchronous e-learning systems based on practice. These concepts are gathered from scientific studies upon e-learning as well as from studying particular applications that represent the top-line in this field.

As with modern technologies, applications can be built efficiently from the point of view of time, cost and development. The paper also proposes a set of useful technologies as well as a demonstration of how they can be adopted in the development process.

The system described in this paper is meant to impersonate the proposed guidelines into a minimum viable product. Since the product is just a baseline, a lot of different paths could be adopted in future works, some of them described below:

- integrating gamification within the system as proposed in Chapter 3.
- studying how the product could be used alongside a synchronous E-learning system or even a traditional learning system to facilitate learning.
- using the data gathered for the intelligent quiz scheduling models to further model the learning curve and analyze humans learning process on specific data.

# Bibliography

- [AAZ08] Ajlan Al-Ajlan and Hussein Zedan. Why moodle. In *2008 12th IEEE International Workshop on Future Trends of Distributed Computing Systems*. IEEE, 2008.
- [Ama] Amazon. Amazon Web Services. <https://docs.aws.amazon.com/>. 2022-12-05.
- [BBBB93] Harry P. Bahrick, Lorraine E. Bahrick, Audrey S. Bahrick, and Phyllis E. Bahrick. Maintenance of foreign language vocabulary and the spacing effect. *Psychological Science*, 4(5):316–321, September 1993.
- [Bev] Beverloo, Peter and Thomson, Martin. Push API. <https://www.w3.org/TR/push-api/>. 2022-12-05.
- [CM11] Ruth Colvin Clark and Richard E. Mayer. *E-learning and the science of instruction : proven guidelines for consumers and designers of multimedia learning*. Pfeiffer, 989 Market Street, San Francisco, CA 94103-1741 [www.pfeiffer.com](http://www.pfeiffer.com), 3rd edition, 2011.
- [Coc00] Alistair Cockburn. *Writing Effective Use Cases*. The Crystal Collection for Software Professionals. Addison Wesley, Boston, MA, October 2000.
- [Fas] Fasih, Ahmed. Ebisu: intelligent quiz scheduling. <https://fasiha.github.io/ebisu/>. 2022-12-05.
- [Fie00] Roy Thomas Fielding. *Architectural Styles and the Design of Network-based Software Architectures*. PhD thesis, UNIVERSITY OF CALIFORNIA, IRVINE, The address of the publisher, 2000.
- [Gre] Grebla Horea. Cloud Applications Architecture. Cloud Applications Architecture Course at Babes-Bolyai University of Cluj-Napoca; year of study 2021-2022.
- [Hra08] Stefan Hrastinski. Asynchronous and synchronous e-learning. *EDUCAUSE Quarterly*, 31(4), 2008.

- [Ion] Ionic. Ionic Docs. <https://ionicframework.com/docs>. 2022-12-05.
- [KK16] Nurul Nadirah Mohd Kasim and Fariza Khalid. Choosing the right learning management system (LMS) for the higher education institution context: A systematic review. *International Journal of Emerging Technologies in Learning (iJET)*, 11(06):55, June 2016.
- [Mos] Mosalingua. Mosalingua. <https://www.mosalingua.com/en/our-apps/the-mosa-learning-method/>. 2022-23-05.
- [MR] David Metcalf and David Rogers. Contextual learning and memory retention. In *Multiplatform E-Learning Systems and Technologies*, pages 309–320. IGI Global.
- [MSBT04] Glenford J Myers, Corey Sandler, Tom Badgett, and Todd M Thomas. *The art of software testing*. Business Data Processing S. John Wiley & Sons, Nashville, TN, 2 edition, July 2004.
- [Mun11] C. I. Muntean. Raising engagement in e-learning through gamification. *The 6th International Conference on Virtual Learning*, page 323–329, 2011.
- [Mwa21] Waweru Mwaura. *End-to-End Web Testing with Cypress*. Packt Publishing, Birmingham, England, January 2021.
- [Net] Networkstraining. Networkstraining. <https://www.networkstraining.com/>. 2022-12-05.
- [Nom] Noman, Shahid and Naveed, Hasan Sufi and Dylan, Martyn Desrosier and Sangmyung, Park. Client-server Architecture. [https://cs.uwaterloo.ca/~m2nagapp/courses/CS446/1195/Arch\\_Design\\_Activity/ClientServer.pdf](https://cs.uwaterloo.ca/~m2nagapp/courses/CS446/1195/Arch_Design_Activity/ClientServer.pdf). 2022-14-05.
- [PBB<sup>+</sup>07] H. Pashler, P. Bain, B. Bottge, A. Graesser, K. Koedinger, M. McDaniel, and Metcalfe. Organizing instruction and study to improve student learning. 2007.
- [PKH07] John Pettit and Agnes Kukulska-Hulme. Going with the grain: Mobile devices in practice. *Australasian Journal of Educational Technology*, 23(1), March 2007.
- [Rai19] Matt Raible. *The jhipster mini-book the jhipster mini-book*. Lulu.com, Morrisville, NC, January 2019.

- [RRL14] Hemant Rana, Rajiv Rajiv, and Manohar Lal. E-learning: Issues and challenges. *International Journal of Computer Applications*, 97(5):20–24, July 2014.
- [SG06] Philip M. Sadler and Eddie Good. The impact of self- and peer-grading on student learning. *Educational Assessment*, 11(1):1–31, 2006.
- [SM07] Aleksander Baumann Spro and Versvik Morten. Multiplayer on one screen entertainment system. Master’s thesis, Norwegian University of Science and Technology, 7 2007.
- [Spr] Spring. Spring. <https://spring.io/>. 2022-12-05.
- [Tho] Thorben Janssen. Best Practices for Many-To-One and One-To-Many Association Mappings. <https://thorben-janssen.com/best-practices-many-one-one-many-associations-mappings/>. 2022-16-05.
- [Twi] Twinkl. Twinkl. <https://www.twinkl.ro/>. 2022-12-05.
- [Vio11] Fabio Viola. *Gamification - I Videogiochi Nella Vita Quotidiana*. Arduino Viola, April 2011.
- [WE00] Elaine Walker and Steve Elsworth. *Grammar practice for elementary students with key new edition*. Grammar Practice. Longman, London, England, May 2000.
- [ZM19] Hao Zhong and Hong Mei. An empirical study on API usages. *IEEE Transactions on Software Engineering*, 45(4):319–334, April 2019.