Thesis Diploma

DEVELOPMENT OF AN ONLINE INTERACTIVE LEARNING APPLICATION

Georgios Harakis

UNIVERSITY OF CYPRUS

DEPARTMENT OF COMPUTER SCIENCE

June 2020

UNIVERSITY OF CYPRUS
DEPARTMENT OF COMPUTER SCIENCE

DEVELOPMENT OF AN ONLINE INTERACTIVE LEARNING APPLICATION

Georgios Harakis

Supervisor
Dr. Anna Philippou

The Individual Diploma Thesis was submitted for partial fulfillment of the requirements for obtaining the degree of Informatics of the Department of Informatics of the University of Cyprus

June 2020

Acknowledgements

I would like to thank IAESTE Cyprus for giving me the opportunity to work at the University of Oslo in Norway, where I gained important knowledge on how to develop a web application. I would also like to thank the faculty at the University of Oslo for also providing me with knowledge on the subject.

I would like to thank the professors at the University of Cyprus for the various courses that I had undertook in all four years of the program. I would also like to thank my supervisor for giving me the opportunity to work on a thesis that I am interested in and also for giving me the freedom on how to develop the application.

Abstract

The goal of my thesis was to develop an interactive web application, which would serve as a learning tool (for class and home) for the course EPL 211. In our age it is very important to use technology in our daily tasks. Teaching is one of these tasks and the benefits of using technology in combination with teaching are endless. Students are more engaged during lectures when there's technology involved. The days where the students read only from books are behind us and we need to embrace new teaching methods that involve our devices.

The aim of my thesis was the develop a web application that promotes active learning during lectures and at home. With the use of a web application, the student will have more initiative to pay attention during the class and have more motivation for study at home using the application. The aim of the thesis was to develop a useful web application for this purpose.

I have had many constraints during the development of the application. I had to create many prototypes and switch technologies in order to because some were more viable than the others. I had to learn these new technologies and that took a lot of time, where I had to spend hours to practise and fix small mistakes in my code. I was also constraint by time. By the end of the year I had created 3 prototypes each with more features than the other, but this cost me a lot of time that could have been used for the final prototype. This is because I was not very familiar with the technologies and had to learn them in a small period.

The final prototype is running on the web and has many features available. You can create your own user and profile. Then you can navigate through the web application and create your own exercises for other users to solve (both homework and for in class). You can also enter the discussion forums and post your questions for other users to answer. You can also answer other user's questions.

Table of Contents

Chapter 1

Introduction

1.1 Background and Motivation

In the past few years there has been a tremendous development of technology. Particularly, there has been a strong interest in using technology in the education system. Interactive tools such as web applications and web conferencing have gained an increase in their usage in classrooms around the world. The reason for this rapid development is the obvious improvement technology provides to the teaching quality. Students are more likely to be engaged in the classroom when they use their electronic devices, such as their laptops or their smartphones. Therefore, the aim of my thesis was to come up with an idea for an educational web application. EPL 211 is a good candidate for this subject, because it is a theoretical course, and will benefit from a web application that would add more practical interactivity to it.

Another reason for this project is the increasing use of active learning in classes. Active learning is a way of teaching, that makes students active participants in their learning during class time. Typically, these strategies involve several students working together during class but may also involve individual homework. These teaching approaches range from short, simple problem solving and discussions, to longer, involved activities or pedagogical frameworks like case studies, role plays, and structured team-based learning. This work during class can be made possible with the use of a web application.

The following plan was followed: in September of 2019 I started preparing a system's requirements and specification document, which served as the roadmap of what I was going to develop in the following months. I prepared a system's requirements documents, which

also contained the various constraints (GDPR, hardware, accessibility issues etc.) that all developers have to take note of before starting to develop their web application.
Then I started creating prototypes for the application. The first prototype was developed only with the frontend part of the web application, the second prototype with both frontend and backend and the third and final prototype was designed with frontend, backend (and database) and authentication. Then I did some testing of the third prototype and finally put it in the web.

1.2 Description of the system

In the web application, there are login and registration features, where you can create your own user and profile. After creating a user and logging in you can create your own multiple-choice questions for other users to solve. You can also ask questions in the discussion forums. You can also answer other user's questions. The final prototype has a user-friendly interface and satisfactory response times that provide a good user experience. Finally, all the data of the web application are stored in a database.

1.3 Outline of thesis

First in Chapter 2, I am going to present a research I have done during October, when I downloaded some web applications and I tried them out. This research is the first thing I am going to present, because it sets the tone of what kind of application I built. I chose the most popular educational applications that are available for free which include Duolingo, Language Drops, Coursera, Kahoot! Etc. Then after using them for some time, I wrote my observations and my thoughts on the advantages and disadvantages of each application.

In Chapter 3, there are explanations for the various theoretical terms that have to do with a web application. Particularly these are what are web application, frontend development, backend development and framework. Also, in chapter 3, the various technologies that were used to develop the applications are explained. Particularly these technologies are HTML, CSS, JavaScript, vue.js, react.js, node.js, express, mongo DB and npm.

In Chapter 4, I analyze the functional requirements of the final system and I also analyze the non-functional requirements of the web application and constraints that must be taken into consideration, such as security, privacy of information. Then there is an analysis of what interfaces should be included in the web application. Examples of these interfaces include Register interface, Login interface etc. I finally present the various, issues of reliability, availability and accessibility of a web application.

In Chapter 5, there are details about the implementation of the three prototypes. For all three prototypes there is an introduction on how the environment was setup and screenshots with descriptions from features of the actual implemented applications. For the third prototype there is explanation for the key parts of the code. All three prototypes have a section, which includes comments on the prototypes and the features, which were included in the end. Finally, in Chapter 6, I summarize my thesis.

Chapter 2

Educational application research

For the preparation of this section I have downloaded some of the most popular educational that were available in the google play store. My choices were also based on how highly ranked they were. I have made a critical assessment on which features are advantages and disadvantages of each application.

2.1 Amazon Kindle

Amazon provides a service which has a large amount of reference guides, how-to books, self-help books, textbooks, and much more. It is the traditional way of learning, by simply reading your textbooks through your device. [1]

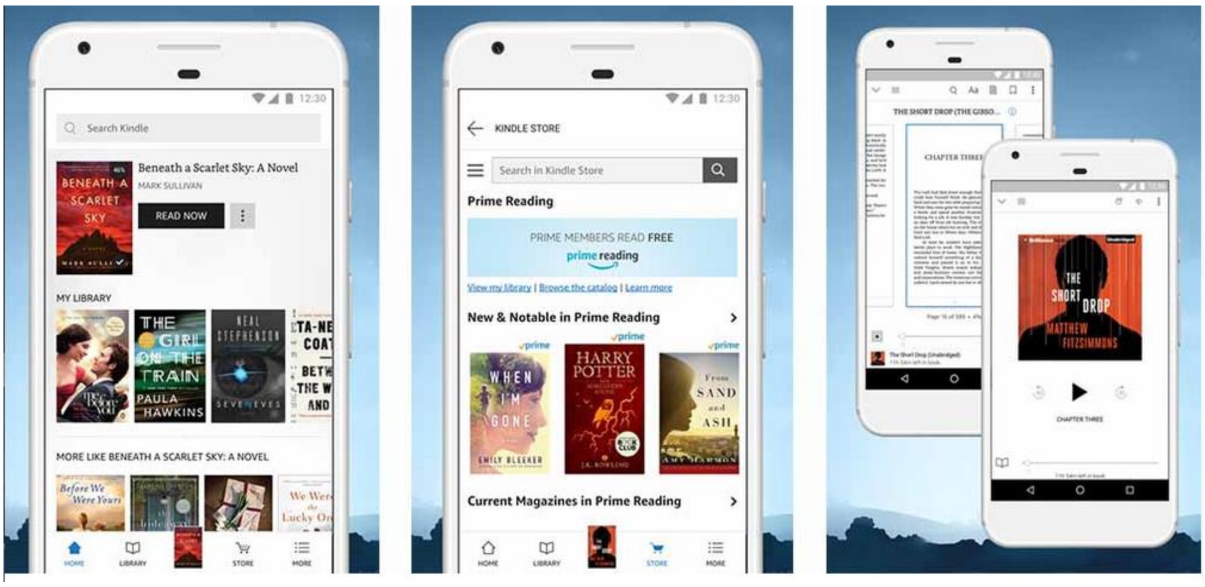

Search Kindle
Beneath a Scarlet Sky: A Novel
MARK SULLIVAN
READ NOW
MY LIBRARY
MORE LIKE BENEATH A SCARLET SKY: A NOVEL
HOME
LIBRARY
STORE
MORE
KINDLE STORE
Search in Kindle Store
Prime Reading
PRIME MEMBERS READ FREE
prime reading
View my library | Browse the catalog | Learn more
New & Notable in Prime Reading
Current Magazines in Prime Reading
CHAPTER THREE

Figure 2.1: Amazon Kindle application interface

Advantages

- Can hold a large amount of text books, reference guides.
- Can make notes next to important parts of a text book.
- User friendly and has night mode feature. Also, size customization is available. A night mode feature should be useful for the students that study late night, even though many browsers have night mode on their own.
- Can upload your own text books.
- Has the option to listen to e-books.

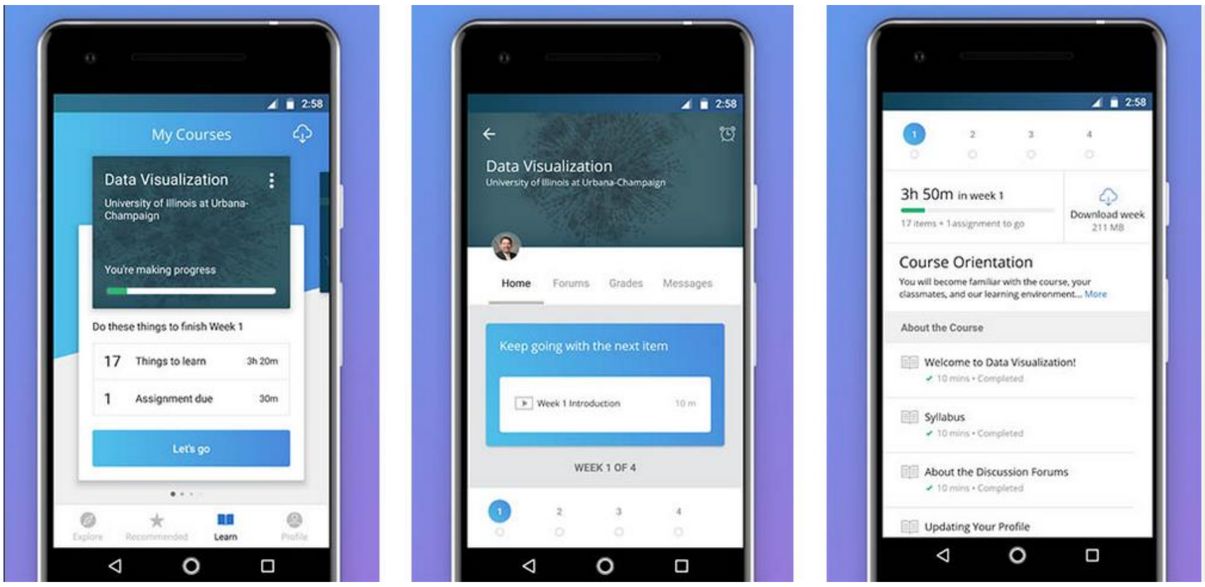
Disadvantages

- Only contains e-books that might be boring for some people.
- Not engaging because it has no interactive activities or mini games.
- The price of some of the e-books can be discouraging to some people.

2.2 Coursera

Coursera is an education application where you can enlist yourself in several classes or topics in many categories of subjects including IT and technology. The classes are available in the form of video lectures and some also contain homework, which can be corrected by a teacher.

Also, there can be multiple choice questions or simple fill the gaps exercise in between lectures. Finally, you can receive a certificate for completing certain courses, so it is more official. [1]



My Courses
Data Visualization
University of Illinois at Urbana-Champaign
You're making progress
Do these things to finish Week 1
17 Things to learn 3h 20m
1 Assignment due 30m
Let's go
Explore
Recommended
Learn
Profile
Data Visualization
University of Illinois at Urbana-Champaign
Home
Forums
Grades
Messages
Keep going with the next item
Week 1 Introduction 10 m
WEEK 1 OF 4
3h 50m in week 1
17 items • 1 assignment to go
Download week
211 MB
Course Orientation
You will become familiar with the course, your classmates, and our learning environment... More
About the Course
Welcome to Data Visualization!
10 mins • Completed
Syllabus
10 mins • Completed
About the Discussion Forums
10 mins • Completed
Updating Your Profile

Figure 2.2: Coursera application interface

Advantages

- The video format of the lecture is very flexible as you can re-watch lectures, play them at the pace you want to play them.
- Has very easy to use forums and thus discussion is very active.
- People can review assignments of other students via the peer review system.
- Has a strong authentication mechanism.

2.3 Duolingo

Duolingo is a language learning application. But in my opinion as an avid user of Duolingo myself it has some great features that could be included in other web applications as well. Duolingo teaches you languages through little mini games and the lessons get harder the more you play a certain language. [1]

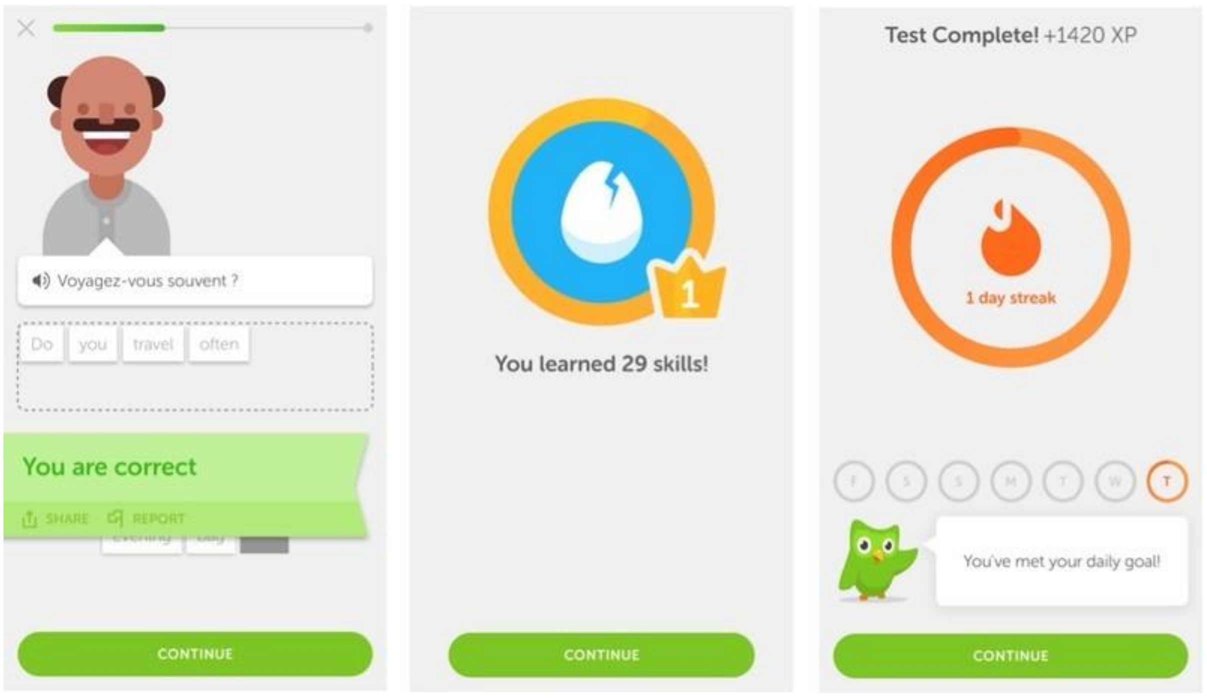


Figure 2.3: Duolingo application

Advantages

- Has a beautiful, simple and practical interface, which motivates the user to spend time solving many exercises. The user interface is simplistic with eye friendly colors.
- Duolingo has a competitive nature, because with every sets of exercises you complete you gain experience. The more points you have the higher climb the leaderboard and the top players get extra lingots (Duolingo currency).
- You can spend lingots (currency) in an in-game store unlocking cool features, which helps with competition and motivation.
- Contributors who are native in a language can moderate a language course and add exercises.

2.4 Khan Academy

Khan Academy is an application where you can enlist yourself in several classes or topics in many categories of subjects, like math, science, physics, economics and many others. The classes are available in the form of video lectures and sometimes as text. Furthermore, khan academy offers quizzes in between some lessons to test your skills. [1]

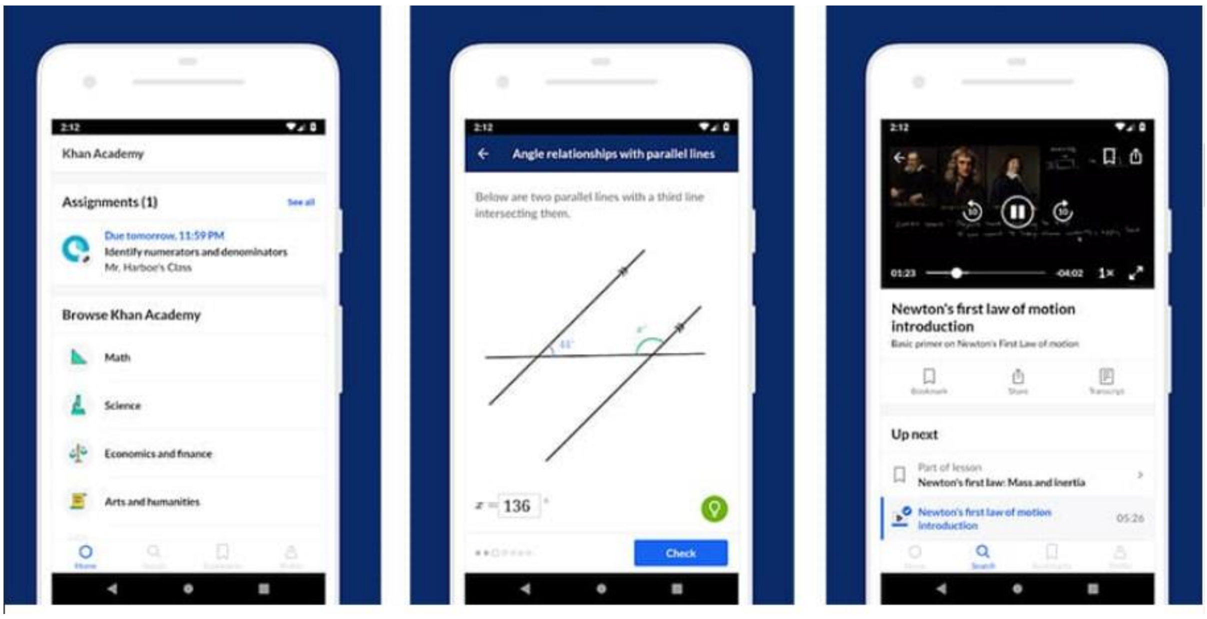

Khan Academy
Assignments (1)
See all
Due tomorrow, 11:59 PM
Identify numerators and denominators
Mr. Harbor's Class
Browse Khan Academy
Math
Science
Economics and finance
Arts and humanities
Angle relationships with parallel lines
Below are two parallel lines with a third line intersecting them.
136
Check
Newton's first law of motion introduction
Basic primer on Newton's First Law of motion
Up next
Part of lesson
Newton's first law: Mass and inertia
Newton's first law of motion introduction

Figure 2.4: Khan Academy application interface

Advantages

- Very similar to Coursera but Khan Academy has a lighter feel when It comes to its material, as it offers very basic classes to very advanced. Coursera mainly offers University courses.
- Has a points system similar to Duolingo (collecting stars for completion of exercises or tutorial).
- Some users are also teachers that create material for students. They also answer questions if they want to. Similarly, the teacher of the web application can post questions.

2.5 Quiz Up

Quiz Up is a competitive quiz application, in which you select one of over 1000 categories and compete against other people in the world. Unlike Duolingo the quizzes are always in the style of a competition, where you play directly against a player. [1]

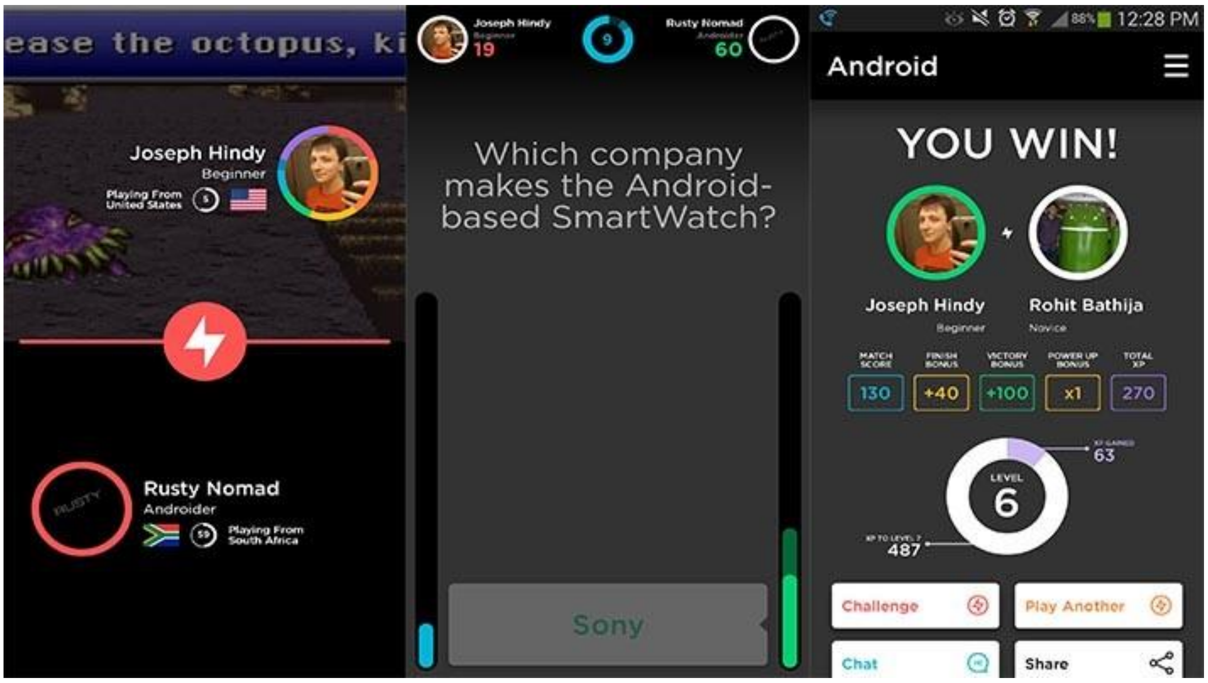


Figure 2.5: Quiz up application interface

Advantages

- Its competitive nature gives initiative to students to exercise regularly in order to beat their opponent in the quiz competitions.
- Huge number of categories available to keep users interested.

Disadvantages

- User interface is not friendly. Interface is loaded with advertisements and surprisingly, it is loaded when you try to select category to quiz.

2.6 Language Drops

Language drops is a language learning application that supports dozens of foreign languages .The learning is done using mini games (such as crosswords as seen in the images below), and the focus is primarily on vocabulary and not grammar. [1]

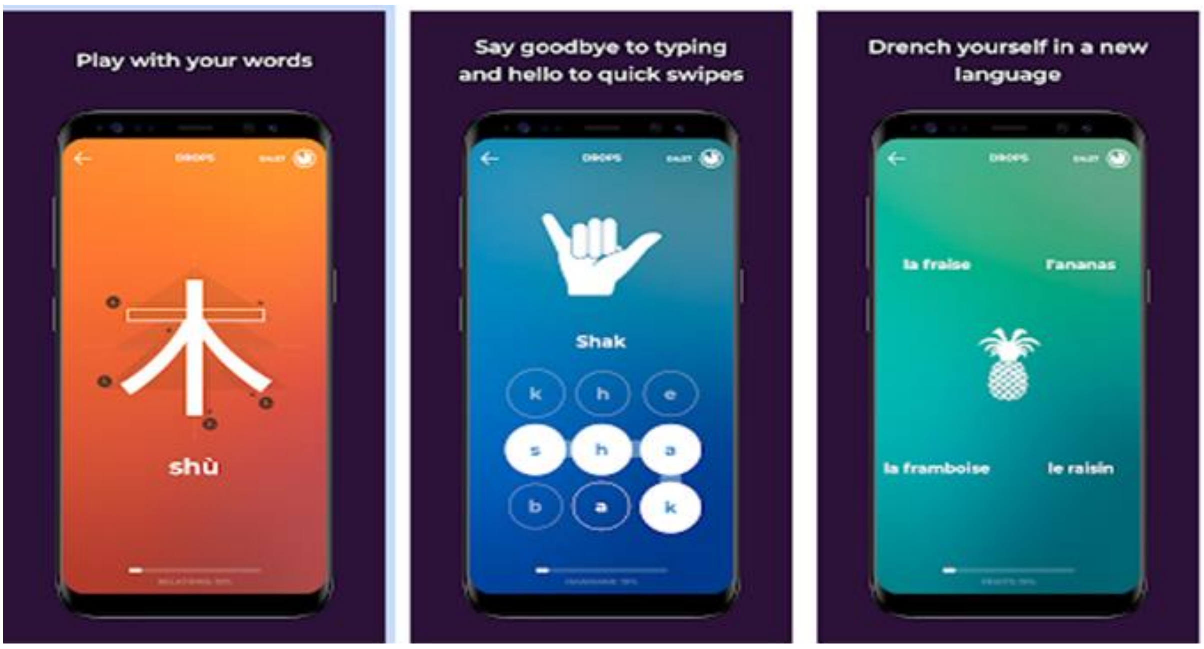

Play with your words
shù
Say goodbye to typing and hello to quick swipes
Shak
Drench yourself in a new language
la fraise
l'ananas
la framboise
le raisin

Figure 2.6: Language Drops application

Advantages

- User interface is very simple with beautiful colours. There is also a varying change in colour depending on what type of exercise you are doing.
- Learning via recognizable images.

Disadvantages

- Limited Material for free users.
- Sometimes it runs slow, which is not ideal when you many people want to use it.

2.7 Kahoot

With Kahoot!, teachers and students can create multiple-choice quizzes and also polls and surveys. It is also used to include questions that do not award points so that you can gather opinion data, or include a question with multiple correct answers without skewing point totals. [1]

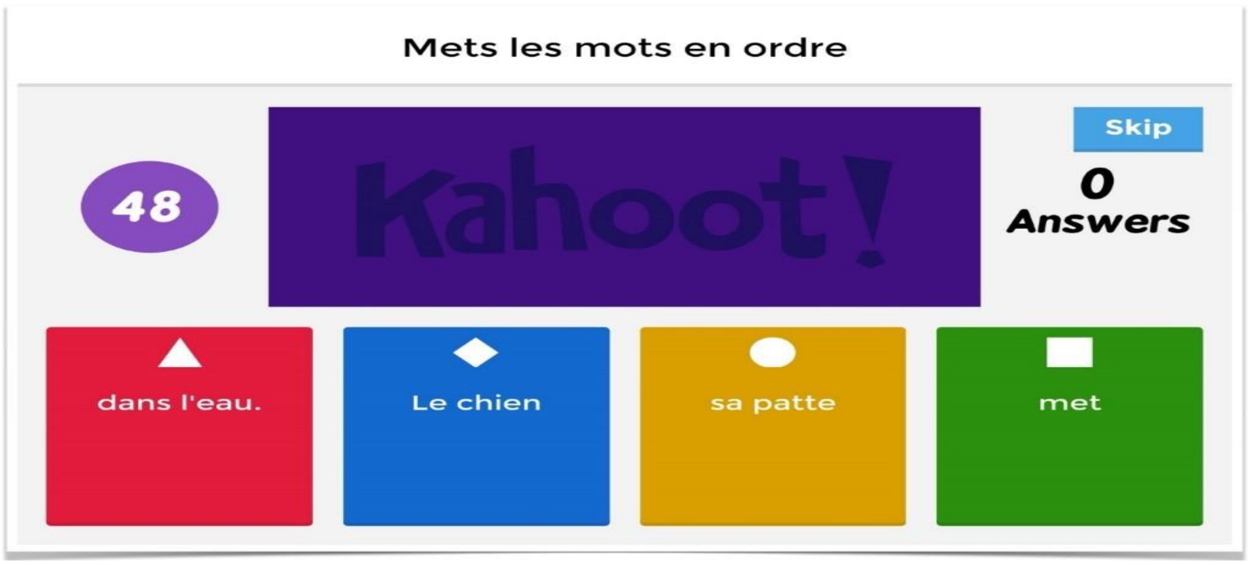


Figure 2.1: Kahoot! application

Advantages

- Provides quiz questions and polls that stimulate quick instructional decisions as well as whole-class discussion.
- Students can connect with other students globally to play or can connect with their peers to compete for new scores.
- Data is saved from each round of play and can be exported to Google Drive or downloaded.

Disadvantages

- The reports are helpful, but they are limited because of the way students connect to the platform. For teachers, this means it can be time-intensive to analyze students' growth patterns or individual problem areas using Kahoot! quizzes.

Chapter 3

Theoretical Background

For the development of the system I followed a prototyping model. Particularly, I have created three prototypes. The first prototype only had frontend, the second prototype had frontend and backend and the final prototype had frontend, backend and authentication. The prototyping model gave me the chance to try out various tools and technologies. These technologies are explained in this section of the thesis. More information about which technologies I used on each prototype is on the next chapter. This chapter also focuses on some important theoretical terms.

3.1 Web application

Web application is a client-server application where server-side scripts are used to retrieve information from a server and client-side scripts are used to display this information to the user. This allows the user to interact with the service using forms, posting requests etc. An advantage of the web application is that its portability depends on the web browser and not on a specific operating or hardware configuration.

A website relies heavily on the server to load pages. On the other hand, a web application does a lot more on the client side. Also, a website is uses static pages and has no interactivity unlike a web application. [2]

3.2 Frontend development

Front-end is the part of the web application you can view. It is the part of the website that a user or customer interacts with. For example, when we go to Amazon, we do not think about their database solutions etc. and we only care about the part the user interacts with to buy stuff. This includes the layout of the website and specifically buttons, images, pagination and even how quick the website loads. A frontend developer has tasks such as designing and developing new layouts, make sure the website is responsive on the mobile phone and make sure the website is accessible all people (e.g. disabled).

Everything on the frontend is designed with a mix of HTML, CSS and client-side scripts of JavaScript. HTML (Hyper text markup language is the skeleton of the web page and is both the structure and the content of the website. CSS (Cascading style sheet) is the styling of the website. Finally, JavaScript is the interactions, how the website moves. JavaScript can be used in both the server and the browser making it very powerful.

Apart from these technologies there are some more specialized platforms called front end frameworks like react.js and vue.js, both of which I use later for my prototypes. [11]

3.3 Backend development

Back-end is the part of the development of a web application that is concerned with the interaction of the server, the application and the database. It can also be described as the logic of the web application because it describes how the application works. The server part of the backend processes the data that is received from the front-end and returns the result to the client. Specialized frameworks are also available for the backend.

A backend developer must be comfortable with database creation, integration, and management. Also, he/she must know how to use back-end frameworks to build server-side software and know how to make checks for data validation. Finally, he/she must integrate user-facing elements with server-side logic.

Technologies usually associated with backend include MySQL, MongoDB, Oracle. In our web application I used MongoDB for database and node.js for data management.

An example of backend is when you login into a web application. When you press submit your data is sent to the server-side software, which makes sure that the text is in the right format and right after will send it to the database where it checks if a user-password duo exists. [2]

3.4 Framework

Framework is a platform for developing software (and web) applications. A framework contains reusable code (predefined classes and functions) that can be used to help the development of an application. Frameworks for web application, offer a variety of tools to handle databases, security, session management, dynamic web development. [3]

3.5 Basic frontend technologies (HTML, CSS, JavaScript)

HTML (Front-end)

HTML is a hypertext markup language, which means it defines the overall structure of the page. HTML features various elements such as paragraphs, lists, headers, links, tables etc. To provide us with options on how to display our data to the user in the browser. HTML is interpreted by the browser and then the formatted text is displayed in the computer or mobile device screen.

An example of HTML code is “<p> Hello World </p>”, which creates a paragraph tag with the content “Hello World”. [10]

CSS (Front-end)

CSS is cascading style sheets and it provides us the tools to design how to interpret text is displayed on the screen. In other words, we can create the style of our web application with CSS. For example, we can change fonts, colors, layout etc. Of our html tags. Finally, CSS enables the web application to become responsive.

An example of CSS code is: “p { text-align: center; color: red;}, where a paragraph’s text is of color red and the text alignment is centered. [10]

JavaScript (Front-end and Back-end)

JavaScript adds interactivity to our web application. For example when you click submit to a form JavaScript methods are invoked in order to take action. Now with the new frameworks (node.js and express) JavaScript can also be used as a back-end language. [10]

3.6 Frontend frameworks (vue.js, react.js)

Vue.js (Front-end framework)

Vue.js is a progressive and performant JavaScript framework used for building user interfaces and front-end applications. It is very popular because it has less of a learning curve than other frameworks. It is also, fast, lightweight. [4]

Below there’s an example of Vue.js code. The Component is divided in the actual content of the component(output), its functionality and its styling.

Output	Functionality	Style
<template>	<script>export default {	<style scoped>
<div class=”user”>	name:”User”,	h1{Font-size:2rem;}
<h1>{{user.name}} </h1>	data(){	</style>
</div>	return{	
</template>	user {name:”Brad”	
	} } }</script>	

Figure 3.1 Example of vue.js code

React Js (Front-end library)

React.Js is a JavaScript library created by Facebook and is used for building user interfaces and front-end applications. It is often mistaken for a framework because of its many capabilities. React is very popular, because it makes frontend easier and makes working in teams easier. It uses self-contained, independent components with their own state. It features JSX, which incorporates JS in mark-up (the HTML). React uses components, where each component can have state which is an object that determines how that component renders and behaves. [6]

Below there is an example of React Js code. The Class of the Component is divided into the state, which is something like variable of a Component and the rendered component, which returns HTML.

```
Class Post extends React.Component{

  state{title : ‘Post One’,body: ‘This is my post’ }

  render()  {return(

     <div>

       <h3> {this.state.title} </h3>

        <p>  {this.state.body} </p>

     </div> )}}
```

Figure 3.2 Example of React.js code

3.7 Backend frameworks and tools (node.js, express)

Node.js

Node.js is a JavaScript Runtime and neither a language nor a framework as it is usually mistaken for. It is built on the V8 JavaScript engine, it is written in C++ and its main task is to allow JavaScript code to run on the server. Some of the advantages of Node is that it is fast, efficient, scalable. Its model is event-driven and non-blocking I/O, which means it does not wait for an input or an output to continue do other tasks (works on a single thread though). This allows it to support a great amount of concurrent connections. [7]

Node.js is best used in REST API & microservices, CRUD applications and basically anything that is not CPU intensive. [7]

Express (backend framework)

Express is a fast, minimalist web framework for Node.js. It can be used in combination with front-end frameworks to build full stack applications. Makes building web applications with Node.js easier and it is light, fast and free. With express we have full control of request and response in the web application. Great compatibility with other frameworks as it's all JavaScript in the end. [7]

Example of express server syntax. Here we initialize a connection with the server and if it is successful it will output "Hello World". The port in which the connection is run is the port number with the number 5000.

```
const express = require(‘express’);

//Init express

const app = express();

//Create your endpoints/route handlers

app.get(‘/’, function(req,res) {

  res.send(‘Hello World’);

//Listen on a port

 app.listen(5000);});
```

Figure 3.3: Example code of Express

Express Middleware functions are functions that have access to the request and response object. Express has built in middleware but middleware can also come from 3rd party packages as well as custom middleware. Middleware functions can execute any code, make changes to the request/response object and call the next middleware in the stack. [7]

3.8 Database (mongo dB)

MongoDB is a No-SQL database. No-SQL databases are much different than relational databases such as SQL or Postgress. With relational database you have to map out exactly what you need in the schema, such as what table you will create, what fields each table will have and their types. With NO-SQL you don’t need all this planning and it’s much more scalable. It is document based and all the data is stored in JSON format. It is also much faster than relational databases. [7]

3.9 Others (NPM, Redux)

React with Redux (State management framework)

Redux is a predictable state container for JavaScript apps. It helps you write applications that behave consistently, run in different environments (client, server), and is easy to test.

Redux is a state management framework that can be used with a number of different web technologies, including React. In Redux, there is a single state object that's responsible for the entire state of your application. So in our React web application each component had its own local state, the entire state of the app would be defined by a single state object that resides in the Redux store.

Redux store is the single source of truth when it comes to application state. Every change in the state of the program must go through it. All state updates are triggered by dispatching actions. An action is simply a JavaScript object that contains information about an action event that has occurred. The Redux store receives these action objects, then updates its state accordingly. Sometimes a Redux action also carries some data. For example, the action carries a username after a user logs in.

After an action is created and dispatched, the Redux store needs to know how to respond to that action. This is the job of a reducer function. Reducers in Redux are responsible for the state modifications that take place in response to actions. A reducer takes state and action as arguments, and it always returns a new state

Although React components can have their own state locally, when you develop a big app, it's best to keep the state in one location with Redux. Redux is not designed to work with React out of the box, you need to use the react-redux package. It provides a way for you to pass Redux state and dispatch to your React components as props. [7]

NPM – Node Package Manager

NPM is used to install 3rd party packages(frameworks, libraries, tools etc) and everything you download is saved in the node_modules folder of your project and the dependencies are listed in package.json file. [7]

Chapter 4

Planning the web application

In this chapter there is a description of the requirements (functional and non-functional) of the system (final prototype). This chapter also contains the interfaces of the web application. Also, there is a discussion about all the issues that involve reliability, availability accessibility of the system and also the constraints (privacy, security etc.) in web development.

4.1 Functional and Non-Functional requirements

Functional Requirements: These requirements consider the needs of a course at University. Particularly, it must be interactive, so the student has to be able to create and solve exercises. Also, the student needs to be able to interact with other users of the web application through discussion forums. They must be able to customize their profile with a few options. There is not a distinction between users, so every user can create and delete their own exercises and discussion posts. The home exercises are in their own section, and the in-class exercises are in their own section as well. The application is meant to be used both in the students' free time and in-class during lectures.

1. User CRUD (create, read, update, delete) login: A user wants to create, read, update, and delete his/her user profile and then sign in the web app.

2. Homepage with overview of the web application: A user wants to view an overview of what to expect in the web application.

3. Exercise CRUD: A user wants to create, read, update, delete an exercise.

4. Exercise type, Multiple Choice: A user wants to be able to create and solve multiple choice exercises.

5. Discussion forums: A user wants to take part in a discussion forum, where users can post their questions, thoughts and other users can answer them in a comment section

6. Dashboard of user with activity: A user wants to have a dashboard page for their profile.

7. Web application available online: A user must be able to access the web application online

Non-functional requirements: These requirements are concerned with user friendliness, privacy, security, performance, and reliability of the system.

1) User Friendliness: The application needs to be easy to use, not complicated to navigate, have a good pattern of colors so the user doesn't get irritated. Also there needs to be a logical way the use can navigate through the navigation bars and buttons.

2) Privacy: The user needs to know and consent with what data he/she needs to provide before he signs up to the web application.

3) Security: The web application needs to be protected from intruders and the data of users needs to be saved securely.

4) Performance and Reliability: The web application needs to have fast response times to the requests of the users, from a simple mouse click on a button to more complicated tasks such as fetching a lot of data from the database. Also, the systems need to be reliable, in the sense that it needs to be working online all dthe time.

4.2 System Interfaces

The user should be able to access the web application via his personal computer, laptop, or his/her smartphone as it will be fully responsive and adaptive for these interfaces.

4.3 User Interface

This is a brief description of some of the interfaces that will be in the web application. These are always subject to change as the development of the web application progresses.

<u>*More specifically the interfaces are:*</u>

Login Interface: It is a form that will include two text boxes one for the username and the other for the password. Also, there will be three buttons below the form. The “register” button is to register a new user and “submit” to submit the login credentials for validation. In order to press “submit” you need to have inputted matching password and username and also that both two text fields are filled in and not empty.

Register Interface: It is a form that prompts you to enter a username, password twice and an e-mail. The username must not be taken from other users, the two passwords needs to match with each other and the e-mail needs to be in the correct format. Also, there will be two buttons below the form. The “submit” button is to submit the register form and the cancel to return to the login page.

Home Interface: Contains a welcoming message to the web application, which also contains the options to navigate to the login or register pages

Dashboard Interface: Contains the page where a user can delete his account or click on a link which takes you to the official EPL 211 website.

Exercises Create Interface: Contains the form to create a new exercise.

Exercises Choose Type Interface: An interface where you choose between two buttons whether to complete in class exercises or home exercises.

Exercises Interface: Contains the list of all the exercises that have been posted by users. There is also a form at the top of the page, where a user can add his own multiple-choice exercise for others to solve

Discussion Interface: Contains the list of all the posts that have been posted by users. There is also a form at the top of the page, where a user can add his own post for others to view and comment on.

Specific Discussion Interface: Contains the page of a discussion and the comments on a certain discussion.

4.4 Software Interfaces

The web application should be working on the following in web browsers such as Google Chrome, Mozilla Firefox and mobile browsers, such as the Samsung browser.

4.5 Reliability, Availability, Accessibility

Most performance issues are user experience issues which all boil down to bad response times. This can be a result of or a combination of unavailable servers, insufficient numbers of servers, unexpected spike in user traffic, inefficient code, delays from third-party APIs.

Reliability encompasses factors that make a web application strong, usable and effective.

Performance and scalability issues are factors that affect the response time of a user request. The web app is not be taking more time than the expected response time when the user requests are increased, as this would cause user dissatisfaction.

Availability issue, the server should be available at all times, because you can’t deny a user from practising in case, they need it immediately.

Fast response time in fixing errors, updates on the code should be made fast when a bug occurs in order to minimize the time the application is not working as intended.

Some basic rules for the web application’s reliability are the following:

- Should be able to support up to at least the current students that are enrolled in EPL 211 for that semester.
- Should have a very small response time when submitting exercises, must be less than a second.
- Should not freeze because of high memory consumption.

- In class questionnaires and exercises need to give results in a fast fashion as the professor wants an instant feedback.
- The connection with the database is consistent. There can be serious consequences if it fails to update new values as fast as possible.
- Average page load (from a user perspective) must be less than 1 second.
- Slowest page load cannot take more than 5 seconds.
- The app must be available more than 99.5% of the time.

4.6 Constraints (privacy, security)

Regulatory policies: There are some regulatory policies the web application follows.

The right of users to grant consent for the use of their data. A privacy compliant was created as it's important for individuals who sign up on the website, to understand how their personal data will be processed. There's a tab in the web application called "Privacy", where users can view their rights. Users registering to the web application have to agree with the privacy compliance text, before they can submit the form. Finally, no cookies are used in the web application so the privacy message does not include a section about cookies.

Accessibility, when designing a web application, we need to take care for some precautions such as

- Low contrast on text.
- Missing alt text on images.
- Missing link text.
- Ambiguous link text.
- Too many navigation links.
- Empty form labels.
- Unclear form controls.[9]

Cyber security and protecting personal data, take appropriate steps to prevent personal data from being accidentally or deliberately compromised. The passwords are salted and encrypted when they are saved in the database and every request is HTTPs, which are HTTP requests over the secure SSL protocol.

Respect copyright, which includes using licensed pictures or documents. Also, this includes protecting the web application from other websites that might steal the data. Every source and tutorial used for the prototypes are listed in the Bibliography of this paper.[9]

4.7 Agile (Prototypes, git)

In the course of the year I have done three prototypes, and for some I have used different technologies. The only fully completed prototype is the third and final, which is available online at epl211webapp.herokuapp.com. Hence the software engineering paradigm that I had used in my development is incremental development where each prototype is better and has more features than the previous one.

For the storage of my code, I used a git repository. Git is a version control system for tracking changes in computer files. For the final prototype I have created a separate Git account with the email epl211webapp@outlook.com, which will help with transferring the ownership of the application from me to my supervisor.

4.8 Service Oriented software engineering

Building applications based on services allows us to have more freedom and scalability. Service based applications can be built by linking services from various providers using most of the programming languages.

RESTful web services are the latest standard for web services and it’s less heavyweight and simpler than the previous standards of web services (such as WSDL). The fundamental element in a RESTful architecture is a resource. The operations that can be performed on a resource are Create resource (POST), Delete (DELETE), READ (GET), Update a value of the resource (PUT). When a RESTful approach is used, the data is exposed and is accessed using its URL. More on how I used these commands on my resources in the implementation chapter. [9]

Chapter 5

Implementation

5.1 Prototype 1

5.1.1 Technologies used and environment setup

For the first prototype I have used Vue.js for the frontend but I have not done anything backend or database related.

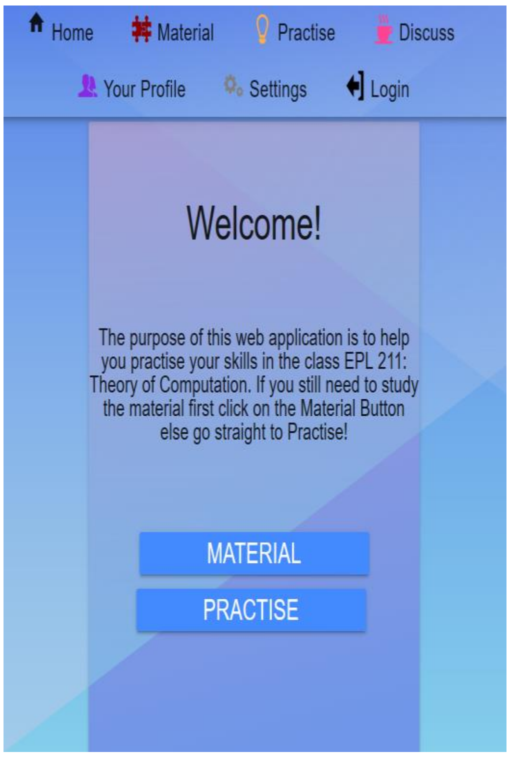
In my opinion the big mistake that I had made was not doing the backend before doing the frontend. I wanted visual proof for my work and that cost me time. Another problem with Vue.js that I had stumbled upon was the fact that I could not find many sources to help me with the development. When I had a problem there weren't many sources that could redirect me to a solution to my problems. Below, I present screenshots from some of the features of the first prototype.

I wasn't satisfied with the colors of the application and the responsiveness of the prototype and that is why I really wanted to start something from the beginning.
My idea for the exercises in the beginning was for the professor to submit a list of exercises, and they would be meant to be presented to the user like in Duolingo. Like a series of exercises and in the end, you would get a score on how many of those you did correct and how many you did wrong.

5.1.2 Screenshots from the prototype and description

Prototype 1 – Screenshots

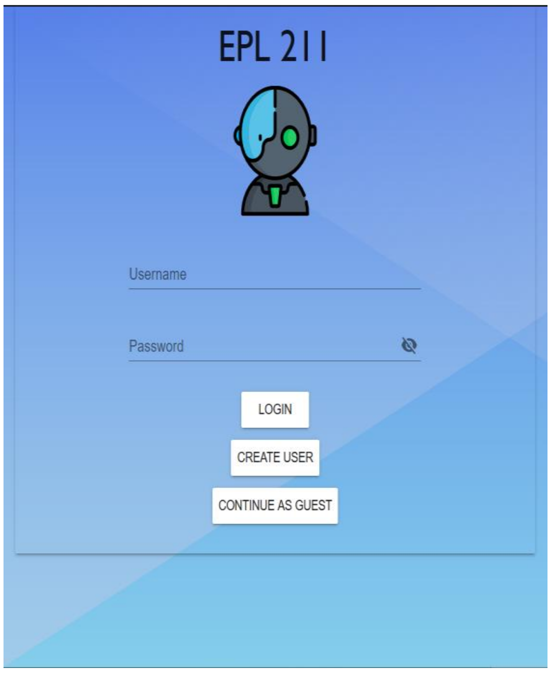
1. First of all I have created a home page, where I explain the purpose of the web application and where the user can be redirected to the material of the course or he/she can choose to go to the Practice.



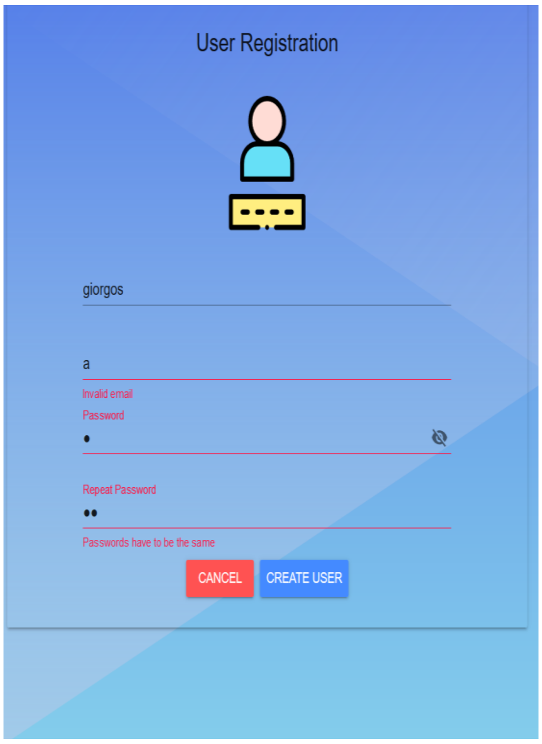
Welcome!

The purpose of this web application is to help you practise your skills in the class EPL 211: Theory of Computation. If you still need to study the material first click on the Material Button else go straight to Practise!

2. I have included a frontend design for the login page, where I put an option to continue as a guest to the website. Also, there is an option if the user does not have an account to create one with "Create User".



3. I have included validation to my form, like the email must be of the right format (xxxx@yyy.com), and the passwords must match.



giorgos

a

Invalid email

Password

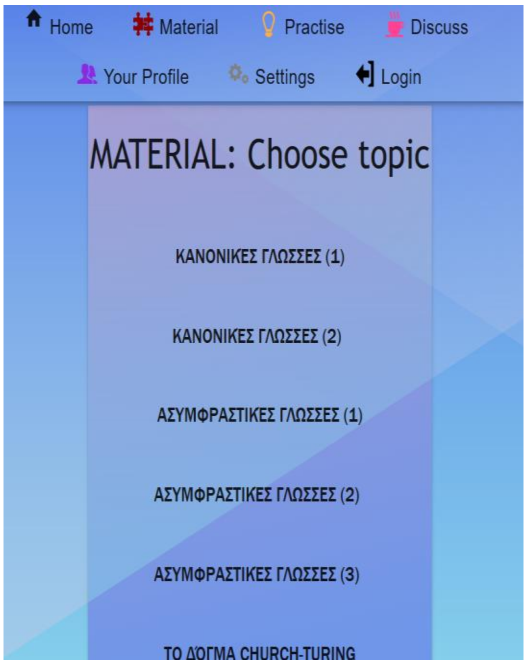
•

Repeat Password

••

Passwords have to be the same

4. When you choose Material or Practice you are presented with a page of all the chapters of the course. In Material, each option is a link to the respective slide presentation of the chapter of the course.



MATERIAL: Choose topic

ΚΑΝΟΝΙΚΕΣ ΓΛΩΣΣΕΣ (1)

ΚΑΝΟΝΙΚΕΣ ΓΛΩΣΣΕΣ (2)

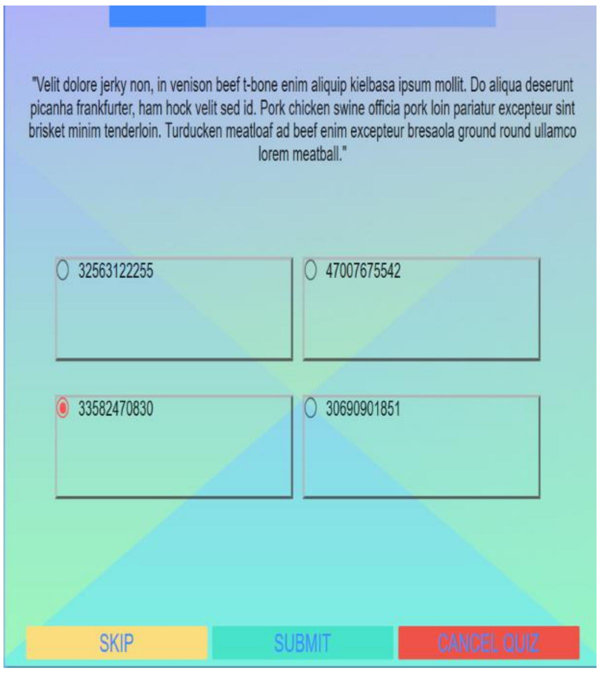
ΑΣΥΜΦΡΑΣΤΙΚΕΣ ΓΛΩΣΣΕΣ (1)

ΑΣΥΜΦΡΑΣΤΙΚΕΣ ΓΛΩΣΣΕΣ (2)

ΑΣΥΜΦΡΑΣΤΙΚΕΣ ΓΛΩΣΣΕΣ (3)

ΤΟ ΔΟΓΜΑ CHURCH-TURING

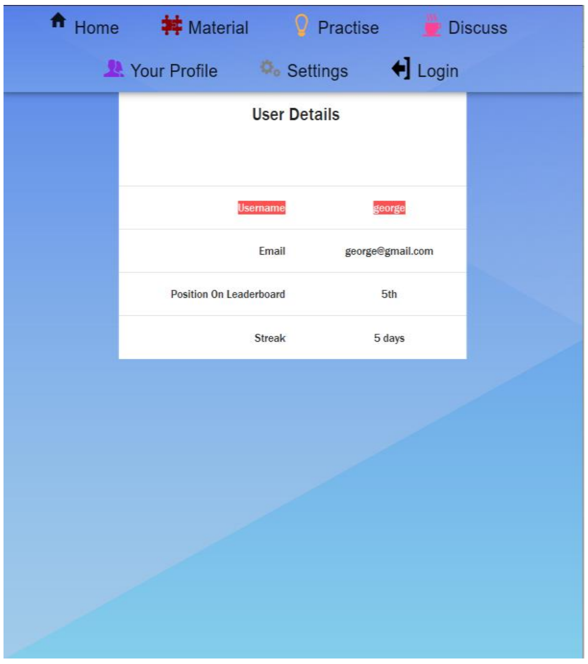
5. Similarly, if the user chooses "Practice" from the home navigation bar the user will be presented with the same interface as the "Material", but when the user clicks on a chapter he/she is automatically redirected to a quiz. The quiz has a series of multiple-choice questions. Each time the user skips or submits one of the questions, the process bar is augmented. The user can always cancel the quiz at any time you wish. After the quiz is done, the user will be redirected to a page where he/she is asked whether he/she wants to repeat the quiz or do another one. I have not finished the feature which counts how many answers were correct by the users



"Velit dolore jerky non, in venison beef t-bone enim aliquip kielbasa ipsum mollit. Do aliqua deserunt picanha frankfurter, ham hock velit sed id. Pork chicken swine officia pork loin pariatur excepteur sint brisket minim tenderloin. Turducken meatloaf ad beef enim excepteur bresaola ground round ullamco lorem meatball."

- ○ 32563122255
- ○ 47007675542
- ● 33582470830
- ○ 30690901851

6. Finally, I made a page that included the details of the user, which was unfinished. Other unfinished pages were the Discuss and Settings pages.



User Details

Username	george
Email	george@gmail.com
Position On Leaderboard	5th
Streak	5 days

5.1.3 What I did not like and what needs improvement

User Interface and Design: I didn't follow any color scheme for this prototype, which resulted in a not so great color user interface. The colors of a web application should be dominant and bold. Variations of light blue do not give a nice contrast between e.g. the exercise been solved and the background. In the next prototype the use of light blue with contrast to white or black instead of another light blue is a better choice. Also, there must be an improvement in the fonts (make them more professional and formal).

Accessibility: The top navigation bar follows basic accessibility rules, like a having a logical order of features of the web application. First, we have the home button, then material and exercises and finally account related features like login and settings.

Missing features: Repeating from above, the backend, databases and authorization are missing. Also, some features from the MUST have features in the system's requirements are also missing, for example.

Below there is a list of the features of prototype 1 (taken from the screenshots of the previous section):

- Welcome page with a description of the web application.
- Login page.
- Registration page with continue as guest option.
- Choose chapter to study material buttons in a separate page.
- Choose chapter to complete a series of exercises buttons in a separate page.

- User details page.

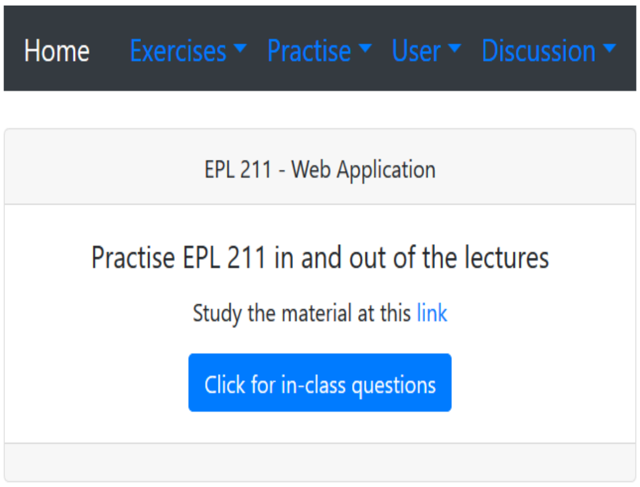
5.2 Prototype 2

5.2.1 Technologies used and environment setup

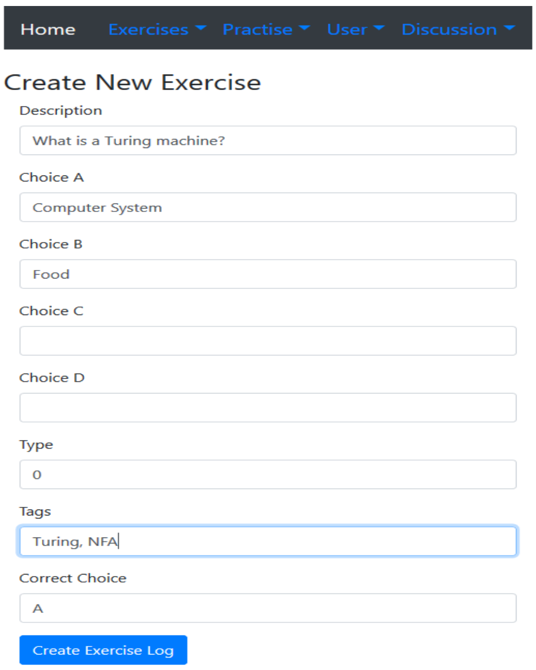
For the second prototype I followed a tutorial by FreeCodecamp on YouTube, which is included in the bibliography. In the tutorial the instructor created a task tracker web application. After studying it I proceeded to make the EPL 211 web application. For this prototype I used React.js for the frontend and node.js for the backend and mongo dB for the databases part. The only part missing from this prototype was authentication for the users. This proved to be a big problem for me in the end because it was very difficult to think of a way to authenticate the users when you have already done the frontend and backend work. I also did not implement the login and register functionalities. [12]

5.2.2 Screenshots from the prototype and description

1. The home page contains a link to the material of the course in the official website of EPL 211. There is a navigation bar at the top of page with the various sections of the application. The plan for the "Click for in-class questions" was to be a quick way for the students to click on the button and be redirected to a quiz in class.


Home
Exercises
Practise
User
Discussion
EPL 211 - Web Application
Practise EPL 211 in and out of the lectures
Study the material at this link
Click for in-class questions

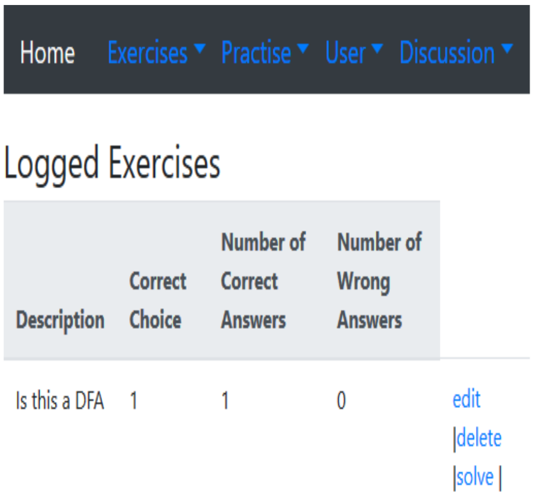
2. When the user clicks "Exercises -> Create an exercise" he/she is redirected to this page, where the user should input the description of an exercise, choices (A to D), what type of question it is (multiple choice, radio etc.), tags that can be used to identify a question and the correct choice of the exercise. Finally, the user clicks on the "Create exercise Log" to create an exercise.



Create New Exercise

Description	What is a Turing machine?
Choice A	Computer System
Choice B	Food
Choice C	
Choice D	
Type	0
Tags	Turing, NFA
Correct Choice	A

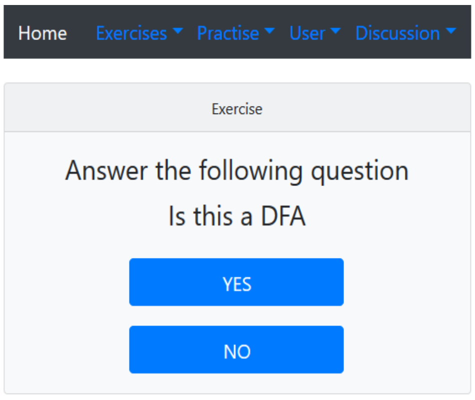
3. In the exercises list the user can view the correct choice for the question, how many students answered correctly and how many answered wrong. There is the option to edit, delete and solve each exercise.



Logged Exercises

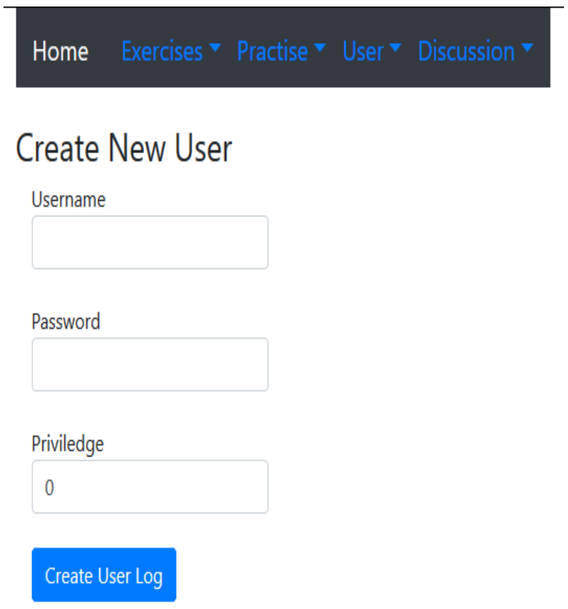
Description	Correct Choice	Number of Correct Answers	Number of Wrong Answers	
Is this a DFA	1	1	0	edit |delete |solve |

4. When the user clicks solve, he/she is redirected to this page, where the user should select the choice to complete the exercise.



Answer the following question
Is this a DFA

5. This is the registration page, where the user can create a new user and assign a privilege. The thought process behind the privilege was to have different privilege levels for a professor and a regular student.



Create New User

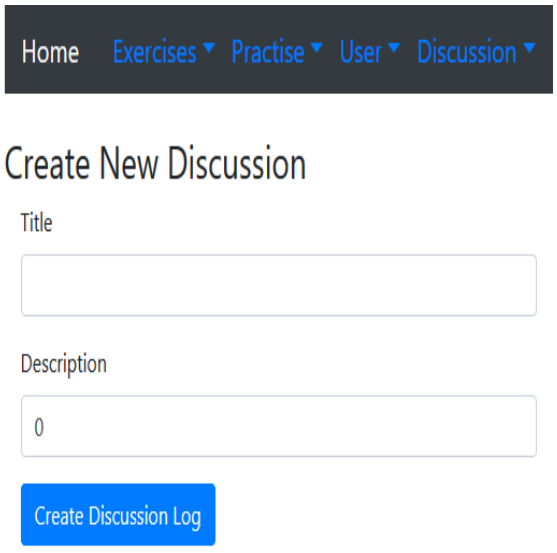
Username

Password

Priviledge

0

6. Finally, there was an idea to create a discussion forum for the web application. Here the students can create new discussions



Create New Discussion

Title

Description

0

5.2.3 What I did not like and what needs improvement

User Interface and Design: This time the colors are much better, with the combination of light blue, grey, black and white harmonizing together to deliver a professional and tidy look to the web application.

Accessibility: The top navigation bar follows basic accessibility rules, like a having a logical order of features of the web application. First home, then practice and exercises and account and social aspects of the web application are last in the order.

Missing features: Even though in the creation of the user I included a privilege field, which can be the basis of having distinct privileges between professors and students. Also, in this prototype are some checks to authorization of users is missing and will be implemented in the third and final prototype.

Below I have included the features of the second prototype:

- Welcome page with link to study material.
- Welcome also has a quick button that redirects to a page for an in-class question.
- User creation page.
- Discussion creation page.
- Exercise creation page.
- Exercise list page, with options to edit and delete.

- Exercise solving page.

5.3 Prototype 3

5.3.1 Technologies used and environment setup

The structure of my code and the way of my thinking for developing the third prototype were taken from the Udemy course: MERN stack front to back, from Traversy Media. It’s a tutorial, where the teacher uses react.js, mongo dB and node.js with express just like the other prototype. [7]

For the final prototype I used React once again. Some very useful React features include:

- **JSX**: JSX is a JavaScript syntax extension.

- **Components**: React is all about components. Every part of React should be a component.
- **Unidirectional data flow**: React implements one-way data flow which makes it easy to think about the logic of your application. [7]

React advantages:

1) Uses a virtual DOM (Document object model) which is a JavaScript object and it is faster than regular DOM.

2) Can be used on both client and server side and with other frameworks.

3) Very good readability because of the use of Components.

4) Using JSX templating makes the application faster, because it performs optimization while compiling JavaScript code. Also, most errors are caught during compilation. Finally, it makes it easier and faster to write templates, if you're familiar with HTML. [7]

Local Environment Setup

Before installing React we need to first install **Node.js**. The Node.js distribution comes as a binary installable for most operating systems and processor types. The prompts in the installation window will guide you to install node.js and by default the installer uses the Node.js distribution in C:\ProgramFiles\nodejs on windows. So, the installer should set the C:\ProgramFiles\nodejs\bin in window's PATH environment variable. [7]

Creating the react.js project

There are two ways of creating a react.js project, a manual and more automatic way. I followed the automatic way because it's more convenient. These are the instructions.

1st) We go to the folder of the project and do the command "npx create-react-app my-app".

2nd) Remove any unnecessary files you won't need for your project (e.g. pictures).

3rd) Start adding your own files.

4th) You can run your application in local host by running the command "npm start".

The next step was creating my user in a service that provides databases. I chose MongoDB. [7]

MongoDB Database contents

MongoDB is not a relational database so there is no way to create ER diagrams, because the contents are documents and not tables with relationships. Below I have included the documents that are in the database.

1. **Exercises Document**: The exercises documenss contain fields for _id, description, choiceA, choiceB,choiceC, choiceD (the four choices for the multiple choice question), correct_choice (the correct choice of the four) , in_class_or_home (wether the exercise is for home or for class), week (which week the exercise will be posted under), chapter (in which chapter the exercise will be posted under), password (an optional field for an exercise password), name (the name of the user that created the exercise), user (the ID of the user that created the exercise), answers (an array with the answers for each user that answered), date (the date that the exercise was posted) and finally correct_users (an array with the users which answered "correctly").

2. **Posts document**: The post documents contain fields for _id, text (the text of the post), name (the user that created the post), user (the ID the user that created the post), likes (an array of the users that have liked a post), comments (an array of the comments that users left on a post), date (the date that the post was posted).

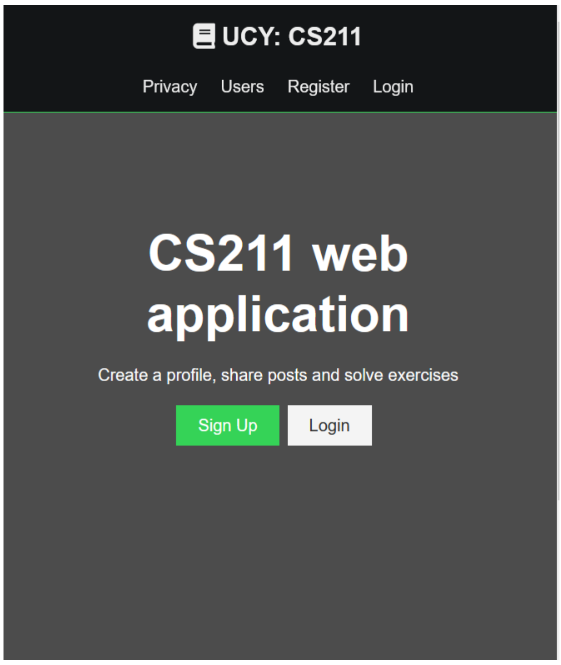
3. **Profiles document**: The profiles documents contain fields for _id, user (the user that creates the profile) company (which can also be a University and not a company), bio (a small description) and finally the status (whether you're studying or learning etc.).

4. **Users document**: The user document contains fields for _id, name(the full name of the user), email (the email of the user, which doesn't have be an actual valid email), password (the password for the user), and the date (the date that the user was created).

5.3.2 Screenshots from the prototype and description

1. When we first enter the web application at https://epl211webapp.herokuapp.com/, we're presented with the following page. As you can see the navigation bar at the top, has four

options. Privacy is a prompt for the GDPR, Users is a list of registered users, Register to register a user and login to login if you already have a registered user. This homepage also contains Sign Up and Login buttons in the centre of the page.


UCY: CS211

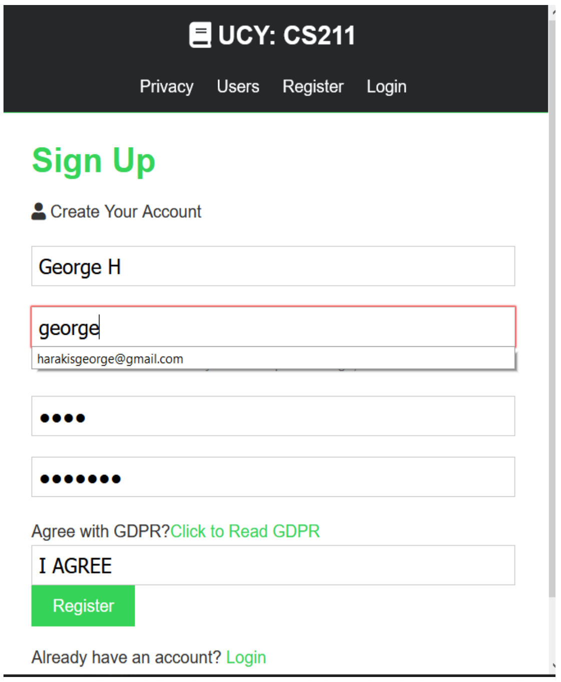
Privacy Users Register Login

CS211 web application

Create a profile, share posts and solve exercises

Sign Up Login

2. When we click Sign Up or Register, we’re redirected to the register page. Here the user has to input a username, an email (not a valid one), a password twice and consent to the privacy prompt of the web application.


UCY: CS211

Privacy Users Register Login

Sign Up

Create Your Account

George H

george

harakisgeorge@gmail.com

••••

•••••••

Agree with GDPR?Click to Read GDPR

I AGREE

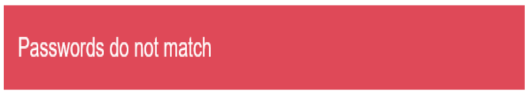
Register

Already have an account? Login

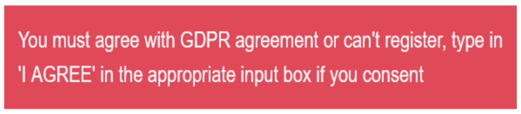
There are some checks in the register form and these include that you have to input a valid email (**@**). This is indicated this a red border and also a small message (but you have to click on Register).

yyyu.com

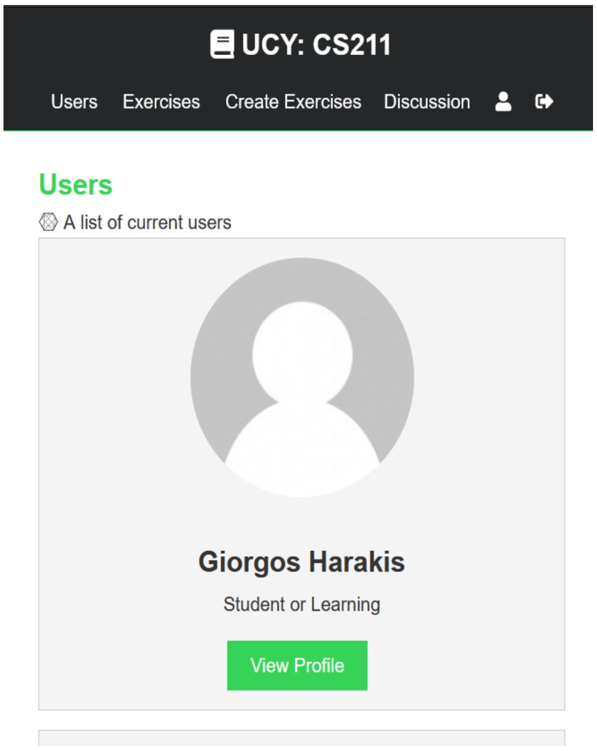
If passwords do not match the following message will appear at the top of the page.


Passwords do not match

If you do not type exactly "I AGREE" the following warning will appear at the top of the page. You can view GDPR by clicking "Click here to read GDPR".


You must agree with GDPR agreement or can't register, type in 'I AGREE' in the appropriate input box if you consent

When everything is okay the user ready to submit by clicking on the "Submit button".
3. If at the navigation bar you click on "Users" the following page with the users of the web application can be viewed.



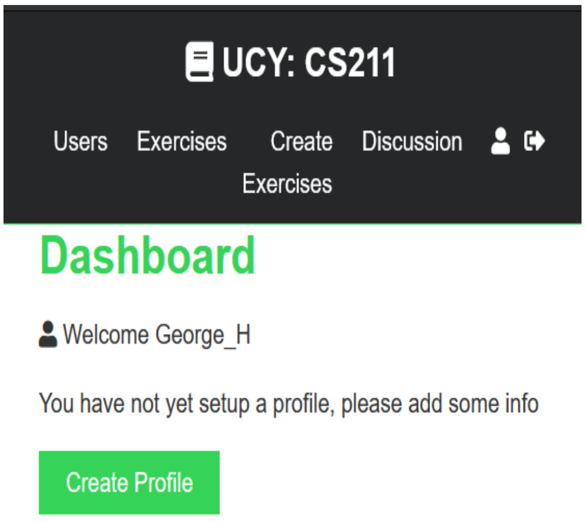
Users

A list of current users

Giorgos Harakis

Student or Learning

4. After registering the user is redirected to the Dashboard where he/she has to click on "Create a profile" to create his/her profile.


UCY: CS211

Users Exercises Create Exercises Discussion

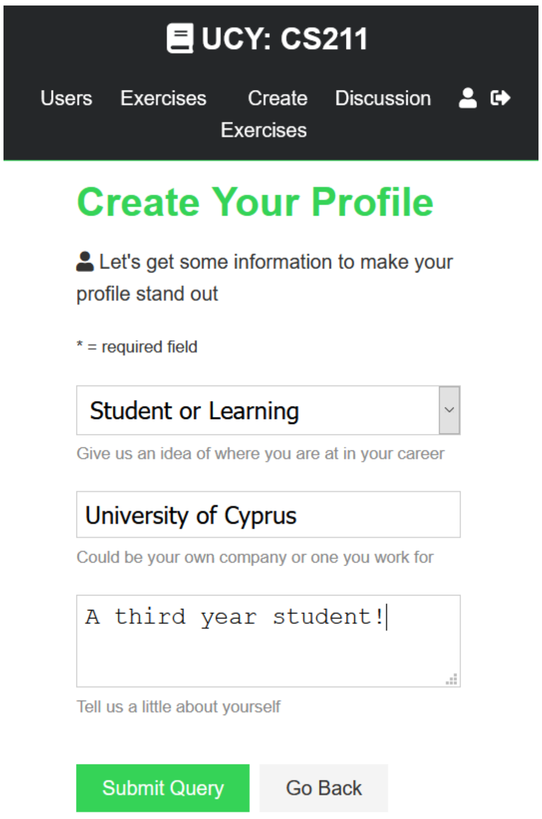
Dashboard

Welcome George_H

You have not yet setup a profile, please add some info

Create Profile

5. The fields that need to be filled in are Status, university or company, and a free input field where the user can type in whatever you want about himself/herself.


UCY: CS211

Users Exercises Create Exercises Discussion

Create Your Profile

Let's get some information to make your profile stand out

* = required field

Student or Learning

Give us an idea of where you are at in your career

University of Cyprus

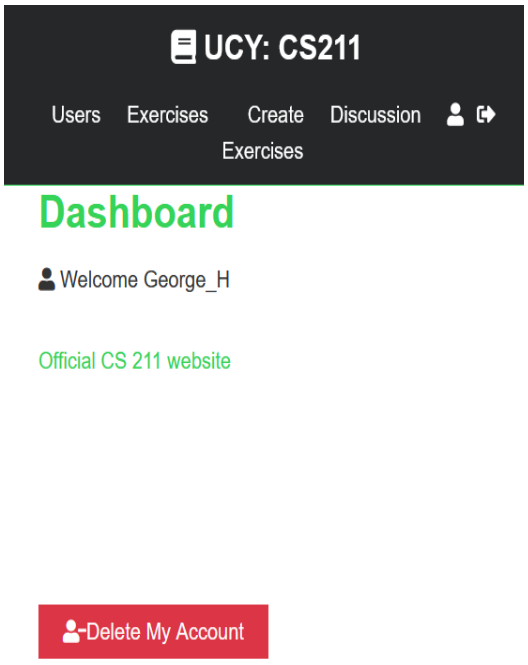
Could be your own company or one you work for

A third year student!

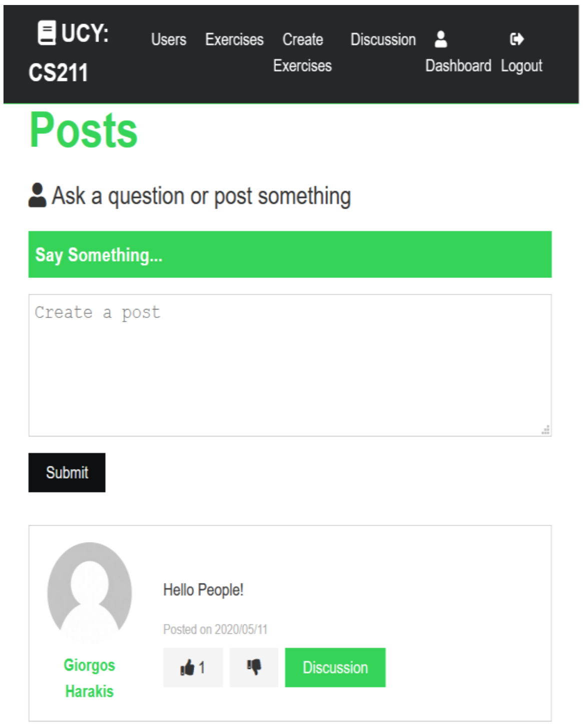
Tell us a little about yourself

Submit Query Go Back

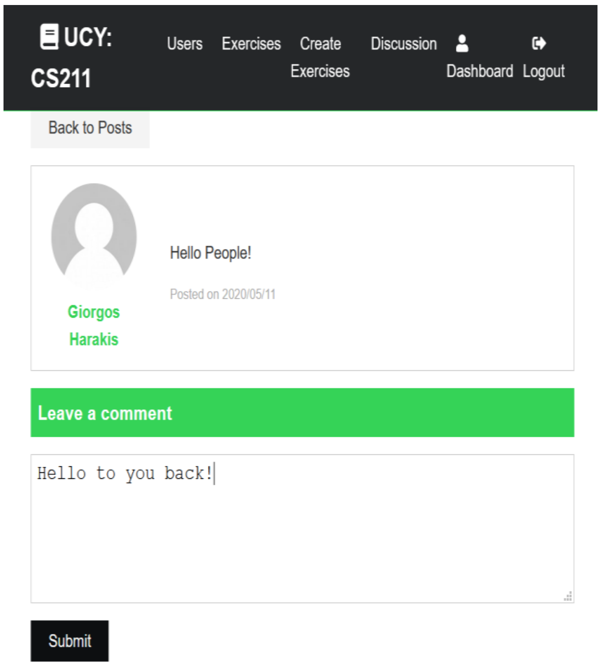
6. Now the user’s dashboard is updated and the user has the option to permanently delete his/her account or click on the link “Official EPL 211 website” to be redirected to the official EPL211 website where he/she can find the material of the course.



7. At the navigation bar at the top (which has now changed since the user is logged in) the user clicks on Discussion. Here the user can start a discussion by inputting in the input box at the top of the page and clicking on Submit. As we can see in the screenshot below, the user Giorgos Harakis has already started a discussion. The user can also like and unlike a discussion post, so people know which discussions are important.



8. When the user clicks on “Discussion” from number 7, he/she is redirected to the page of the specific Discussion where he/she can comment below that post.


UCY: CS211 | Users | Exercises | Create Exercises | Discussion | Dashboard | Logout

Back to Posts

Giorgos Harakis

Hello People!

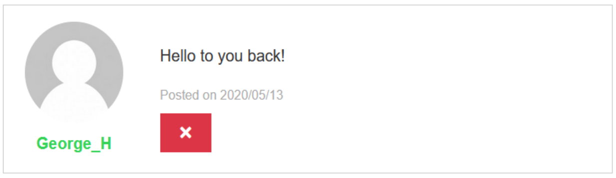
Posted on 2020/05/11

Leave a comment

Hello to you back!

Submit

The user can always delete your comments and discussions posts by clicking on the “X”.

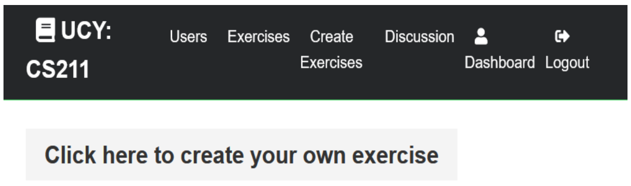

George_H

Hello to you back!

Posted on 2020/05/13

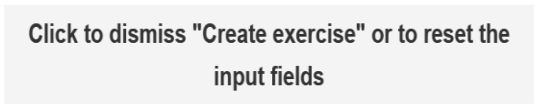
×

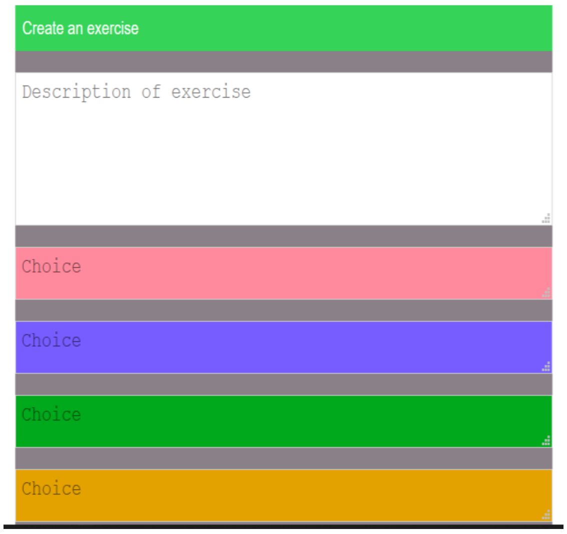
9. When the user clicks on “Create Exercises” on the navigation bar at the top the user will be redirected to the page to create an exercise. There’s a button that slides down the create an exercise form.


UCY: CS211 | Users | Exercises | Create Exercises | Discussion | Dashboard | Logout

Click here to create your own exercise

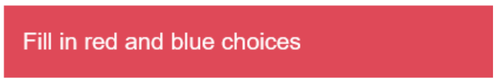
10. There are a lot of fields to fill in. But first, it must be noted that if you want to reset your input fields you can always click on the Dismiss Button. The rules for the first five input fields are the follows. Description, red and blue choices are mandatory, while green and orange can be empty. None of the choices can be the same as another choice.

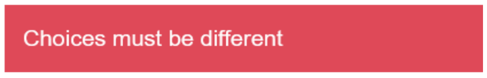

Click to dismiss "Create exercise" or to reset the input fields



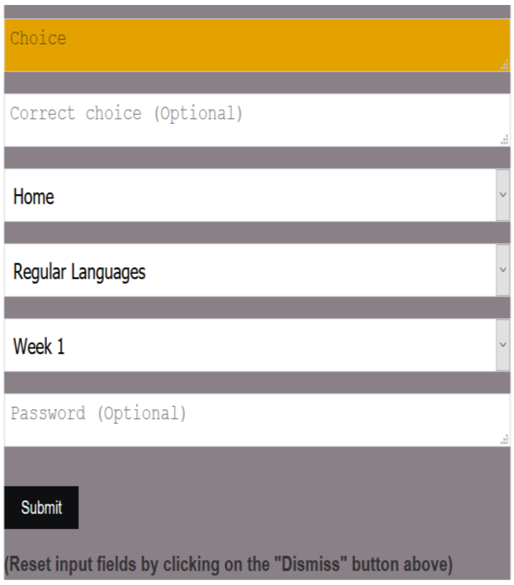
The following error codes might be displayed depending on the mistake.

Fill in description


Fill in red and blue choices

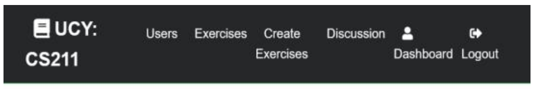

Choices must be different

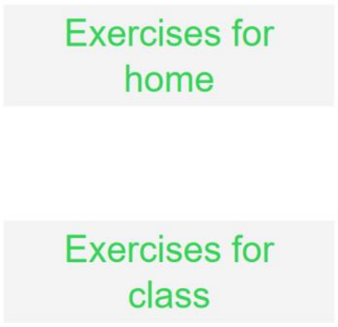
There are also some other fields after the orange choice. Firstly, the user can choose an optional correct choice but makes no difference to what is displayed to the user. There are three drops downs, where the user can choose whether the exercise is for Home or Class, the chapter of the exercise and the week that it will taught at. When the user is done inputting he/she can click on the submit button.



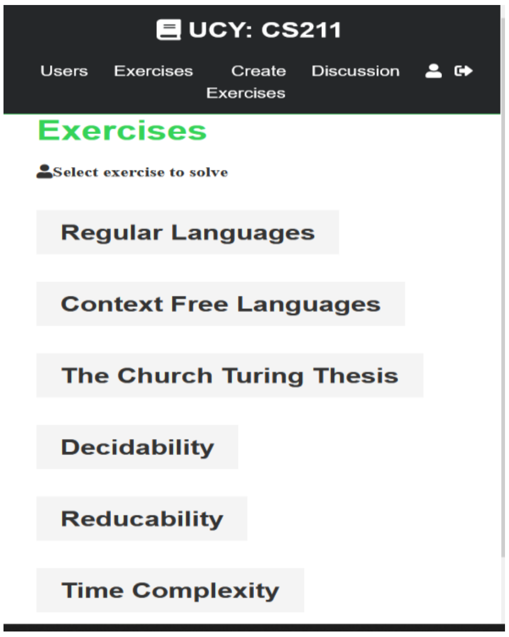
(Reset input fields by clicking on the "Dismiss" button above)

11. When you’re done and press submit, you can click on the Exercises from the navigation where you can choose from Execises for home and Exercises for class.





12. If you choose Exercises for home the user is presented with buttons for each of the chapters of the course to click on. Click to display and click again to dismiss a certain chapter.



Exercises

Select exercise to solve

Regular Languages

Context Free Languages

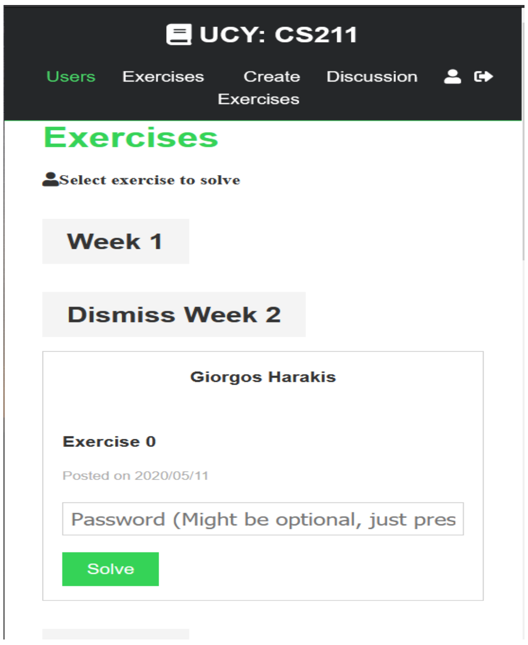
The Church Turing Thesis

Decidability

Reducability

Time Complexity

13. When you click on Exercises for class you can view the buttons for each week, where you can select the exercises of what week you want to complete. Below we chose week 2, and in that way you can view the exercises for week 2. In this example the box for the exericse does not show the description, only a number and the question might be password protected. If it is not password protected you can just press "Solve"



Exercises

Select exercise to solve

Week 1

Dismiss Week 2

Giorgos Harakis

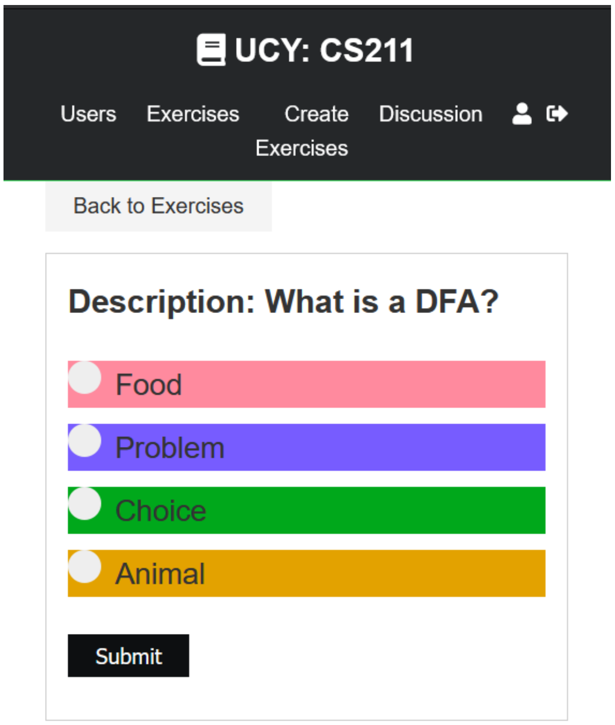
Exercise 0

Posted on 2020/05/11

Password (Might be optional, just pres

Solve

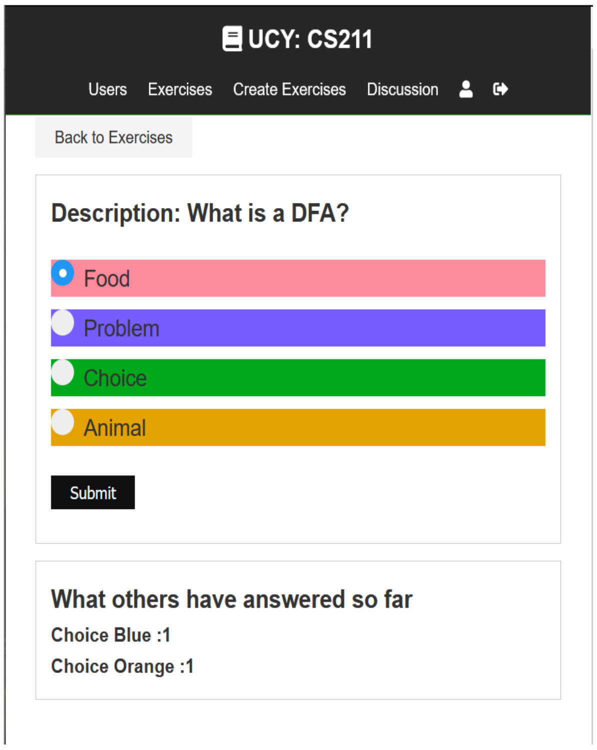
14. When the user puts his/hers password (or not if optional) then the user can view the page for solving the particular exercise.



Description: What is a DFA?

- Food
- Problem
- Choice
- Animal

After the user clicks and submits his/her answer then he/she can view what others have answered (not including your answer). The user can only answer once (every other answer the user attempts will not count on the total).



Description: What is a DFA?

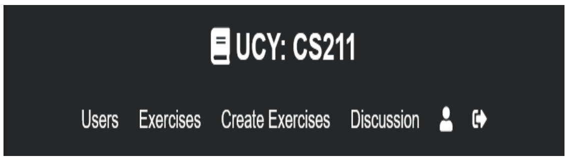
- [✓ Food]
- Problem
- Choice
- Animal

What others have answered so far

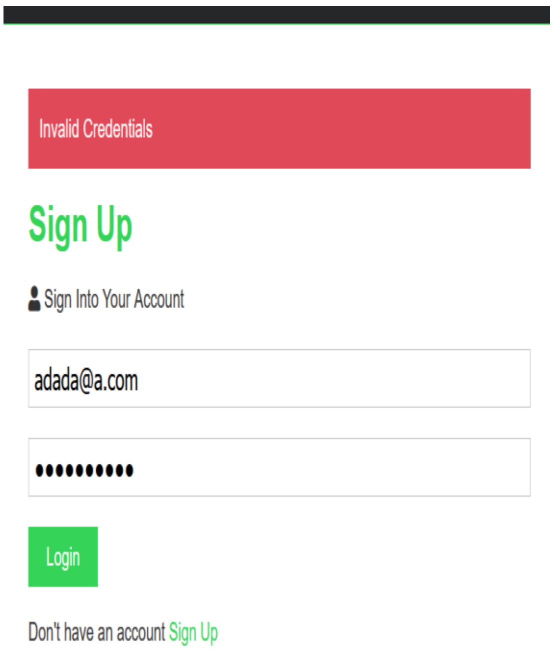
Choice Blue :1

Choice Orange :1

15. The user can logout by clicking on the arrow button at the right part of the navigation bar.



16. When the user logout you can login again. If the user puts wrong credentials then, a message will be displayed.


Invalid Credentials

Sign Up

Sign Into Your Account

adada@a.com

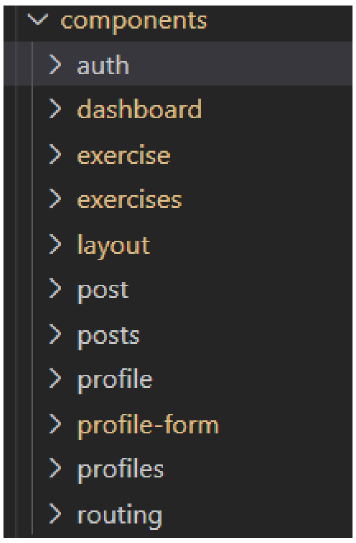
••••••••••

5.3.3 Code structure

Src folder:

In the src folder we have actions, components, reducers and utils folders. We also have App.css, App.js, index.js, store.js

Components folder: Here are the components, which were created using react.js. The following image shows what is included in this folder:



Below I will explain the components that are included in each of these folders.

Auth:

1)Login.js contains the page for the Login. If the user is already logged in and authenticated instead of the Login page, he/she will be redirected to the dashboard page. The main features of this page is the Login form.

2) Register.js contains the page for the Register. If the user is already logged in and authenticated instead of the Register page, he/she will be redirected to the dashboard page. The main features of the file is the Register form, where the user is prompted to input a name, email and also to type in twice his/her password, and they are checked through a function if they match.

Dashboard:

1) Dashboard.js contains the page for the dashboard of a user. If the user has no profile, then there is a link for a page to create a profile. If he/she has a profile already then, the dashboard includes information about the user and also a button that allows him/her to delete his/her account.

Exercise:

1)Exercise.js is the component for the presentation of an individual exercise. This is the component page which displays the exercise to be solved (with the choices).

Exercises:

1)ExerciseChapterButton.js is the component that is the button that appears when you click on one of the chapters in the Home exercises page. There is a mechanism to click on the button and then the exercises for that chapter appear and you can always click it again to dismiss it.

2)ExerciseChoose.js contains the page for the two-button choice of whether you would like to do home exercises or in-class exercises.

3)ExerciseCreate.js contains the page for the form that contains the form that you can create an exercise.

4)ExerciseForm.js contains the actual form for creating an exercise. The form includes all the necessary fields of an exercises (e.g. description, choices etc).

5)ExerciseItem.js is the component for the presentation of an individual exercise. This means the individual box of an exercise when it appears on the exercises list in various parts of the application.

6)Exercises.js contains all the buttons for individual chapters (ExerciseChapterButton.js)

7)ExerciseWeekButton.js is similar to the ExerciseChapterButton, but instead of the chapters it’s for the weeks of the semester.

Layout

1)Alert.js contains the box of alerts that show up on the top of the page, when a user does an action (e.g. creating an exercise), or a user makes a mistake (e.g. an alert for wrong password).

2)Landing.js is the page, which the user views when he first visits the website (the homepage), before he logins or registers. That is why there are buttons on this component prompting him to login or register.

3)Navbar.js is component for the navigation bar at the top of the web application and mainly contains buttons to the many parts of the application (e.g. exercises, profiles etc).

4)Spinner.js is the component for the spinner effect when a page is loading data from the database.

Post

1)CommentForm.js contains the form for creating a comment under a post. The form includes a text input field for the comment

2)CommentItem.js is the component that displays a comment that has been posted, with all the authorization that is needed in order for user that has created a certain comment to delete it.

3)Post.js contains the page for an individual post, where the post itself is being displayed and below that are the comments from other users about the post.

Posts

1)PostForm.js contains the form for creating a post. The form includes a text input field for the post.

2)PostItem.js is the component that represents box that contains a post (including description, dates etc) in the Posts page of the web application

3)Posts.js is the actual pages with all the posts and the Post Items in them.

Profile

1)Profile.js is the component that has the page that displays the profile of a certain user

Profile-Form

1)CreateProfile.js is the component that contains the form for creating a profile and includes all the required fields (e.g. status, thoughts etc.)

Profiles

1)ProfileItem.js is the component that represents box that contains a profile(e.g. username etc) in the Profiles page of the web application

2)Profiles.js is the component for the page that contains all the registered profiles in the web application’s database.

Routing

1)PrivateRoute.js is a special component for the administration of private routes in the web application, for example when the user has already logged in, a private route is used to redirect the user to contents of the web application that were not available before logging in.

Next, I will explain the Redux part of the project and particularly the actions and reducers folders, which I explained above at the Redux sections.

Actions folder

1)alert.js includes only the set_alert action, which dispatches a set_alert type of action

2)auth.js includes the actions that tell us of the success or failure for the authorization actions (load user, register user, login user, logout)

3)exercise.js includes the actions that tell us of the success or failure for the exercise actions (get exercises, delete exercises, add exercise, add answer, add correct answer by user, get exercise, authorize exercise)

4)post.js includes action for posts (get posts, add like, remove like, delete post, add post, get post, add comment, delete comment)

5)profile.js includes the actions for the profiles (get current profile, get profiles, get profile by id, create profile , delete account

6)types.js includes all the constants which are used by all the other action files. This is usually made for convenience to have all the types at one place. a

Reducers folder includes alert.js, auth.js ,exercise.js ,post.js, profile.js and also the index.js which is the reducer's combine file.

Utils folder

SetAuthToken.js is the file that contain the code for the authentication token. Token is a unique identifier for every user.

Apart from folder src contains and a lot of files

App.css contains all the styling for the web application. Every styling (e.g. fonts, alignments, colours etc) are all included in this file and not on the individual components.

App.js is a very important file. This is where we put all the components which are to be pages with a URL (e.g. /exercises redirects to the exercises component). Here are also the Provider tag, which is tasked with passing down the store for Redux. There is also the Router tag, which is responsible for the routing for the pages.

Index.js is is the main file of the application, where the actual application is rendered. This is like the main for other

Store.js is where the store for Redux initialized, created and finally exported to the application.

The following section is dedicated to the backend of the application. In this section I will document the config, middleware, models, routes folders and also the server.js file.

Config folder

1)db.js is the file related with the database configuration

Middleware folder

1)auth.js is the file for the authorization, where we create the token for each user.

Models folder contains the model files for each entity of the application.

1) Exercise.js, Post.js, Profile.js, Users.js contains the schema (the properties in other words), for the various entities in the application. For example the Exercise.js contains properties such as description, name, choice A, choice B etc of an exercise entity

Routes folder contains the route files for REST requests of the application. In these files I will explain every RESTful request.

1)auth.js contains

GET request to "/", which finds the current user that is logged in the application.

POST request to "/", which sends email and password of the user that logs in to authenticate that it is really him/her.

2)exercises.js

POST request to “/”, which creates and posts an exercise (with its details) to the server.

GET request to “/”, which brings us all exercises from the database.

GET request to “/:id”, which bring an exercise by id.

POST request to “/exercises/:id” , which is when you put a password to solve a password-exercise.

DELETE request to “/exercises/:id”, which deletes an exercise by id

PUT request to “/exercises/answer/:id”, which puts an answers of a certain user for a certain exercise by id.

PUT request to “/exercises/correct_users/:id”, which puts which users have answered correctly to an exercise by id.

3)posts.js

POST request to “/”, which creates and posts a post(with its details) to the server.

GET request to “/”, which brings us all posts from the database.

GET request to “/:id”, which bring a post by id.

DELETE request to “/posts/:id”, which deletes a post by id

PUT request to “/posts/like/:id”, which puts a like of a certain user for a certain post by id.

PUT request to “/posts/unlike/:id”, which puts an unlike of a certain user for a certain post by id.

POST request to “/posts/comment/:id”, which puts a comment under a post.

POST request to “/posts/comment/:id/:comment_id”

4)profile.js

GET request to “/profile/me”, which gets the current logged in user’s profile.

POST request to “/profile”, which posts the profile’s information to the database.

GET request to “/profile”, which gets a profile from the database.

GET request to “/profile/user/:user_id”, which brings a profile by id.

DELETE request to “/profile”, which deletes the current profile (removes user posts, profile and user).

5)users.js

POST request to “/users” which is the request that creates a user and sends it to the database.

Server.js file is the file where we connect to the database, initialize middleware and define routes of the application.

5.3.4 What I did not like and what needs improvement, according to me and others who have tried the web application.

What I liked: The user interaction is the best so far. The web application is very responsive and has some very nice colour combinations. The total look of the web application is more professional than the predecessors and there aren’t any missing parts this time.

What needs improvement: Some responsiveness on a mobile phone can be improved. Especially on small screen smart phones. This does not hinder the usability of the application, but things could improve.

In this section I will analyse whether the functional requirements from Chapter 3 have been met. The proof of my work can be validated when you visit https://epl211webapp.herokuapp.com/.

1. User CRUD (create, read, update, delete) login: A user can create a new account in the web application by

2. Homepage with overview of the web application: There is a homepage which states the purpose of the web application, and also provides buttons to Sign up or Sign into the system.

3. Exercise CRUD: A user can choose from the navigation bar the option “Create an exercise” to view the form to create a new exercise.

4. Exercise type, Multiple Choice: The only exercise type available are multiple choice questions, and they can have from 2 to 4 options.

5. Discussion forums: When you click on discussion the user can post anything he/she wants others to view. The user can also comment on other users’ posts, and like or unlike them.

6. Dashboard of user with activity: The dashboard has a link to the official EPL211 website and also an option to delete the account.

7. Web application available online: The web application is online at all times.

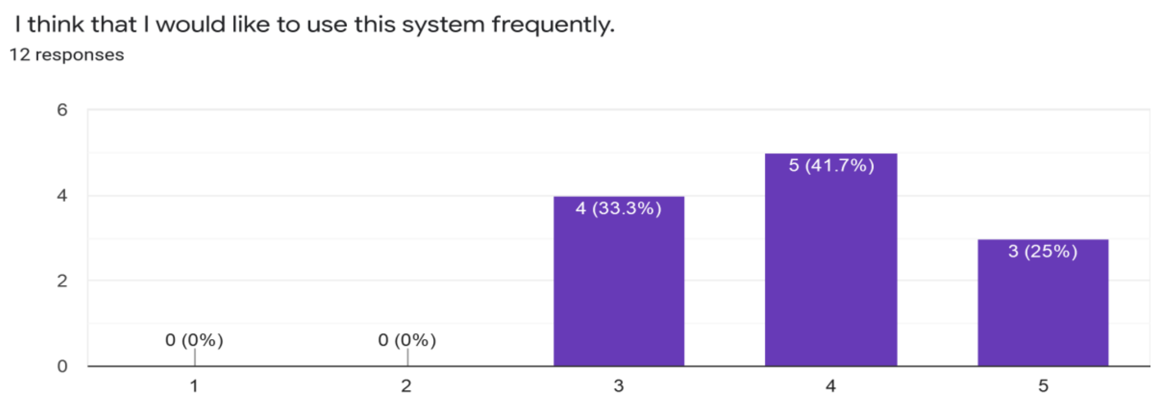
I also provide results for the **non-functional requirements**. After completing the final prototype I have come with satisfactory results. First of all, I did not use third-party APIs, so delays related to third-party APIs will not occur. I have used the reliable Heroku.com servers, and even though it is the free version it still ensures that the web application will have an almost 100% uptime. Most of the interactivity of the application belongs to the browser (frontend) so it all depends on the device of the user and the browser he/she uses. But even if it is a slow browser or device the application will likely still run smoothly as the technologies, which are explained in chapter 4 of the thesis are mostly lightweight and portable for all devices. When it comes to security, the passwords are salted and saved safely in the database. The connection is running on HTTPS, which is very secure protocol for network communication, so there is no risk for intrusion.

5.3.5 Survey on the third prototype

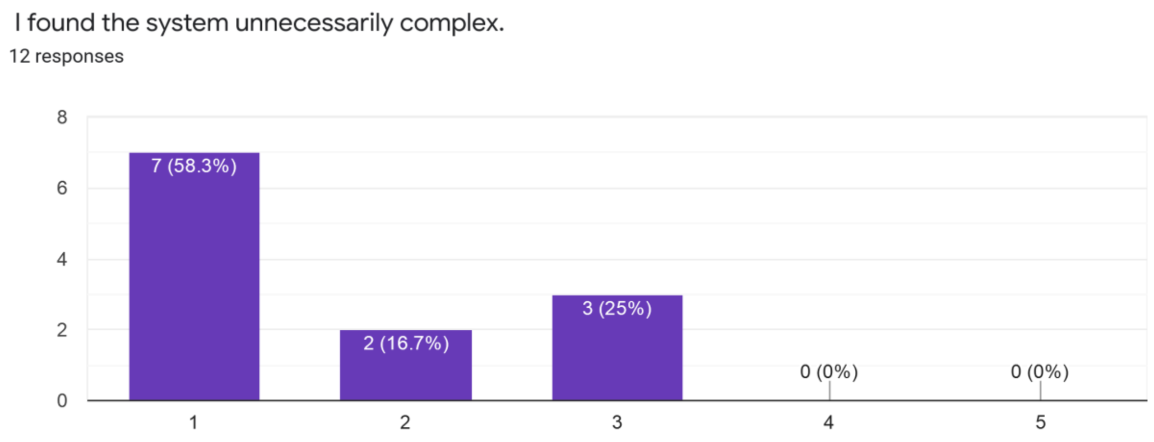
I have created a questionnaire from people that have tried the application. These people are my personal friends and students from my year at University. I have asked around 13 people to try my web application and complete the survey. The full results of the survey are at Appendix C. I will summarize the results here.

1)Most users would use the system frequently.
2)Most users did not find the system unnecessarily complex.
3)Most users found the system easy to use, except from one person who did not find it easy at all.
4) Most users did not feel the need for a technical person in order to use the system.
5) Most users found the functions in the system to be well integrated.
6) Most users found the system consistent but there were a few others who did not find it consistent.
7) Most users think that everyone can learn the system very easily.
8) Some users found the system cumbersome to use.
9) Most users felt confident using the system.
10) Most users felt they did not need to learn a lot of things before using the system.

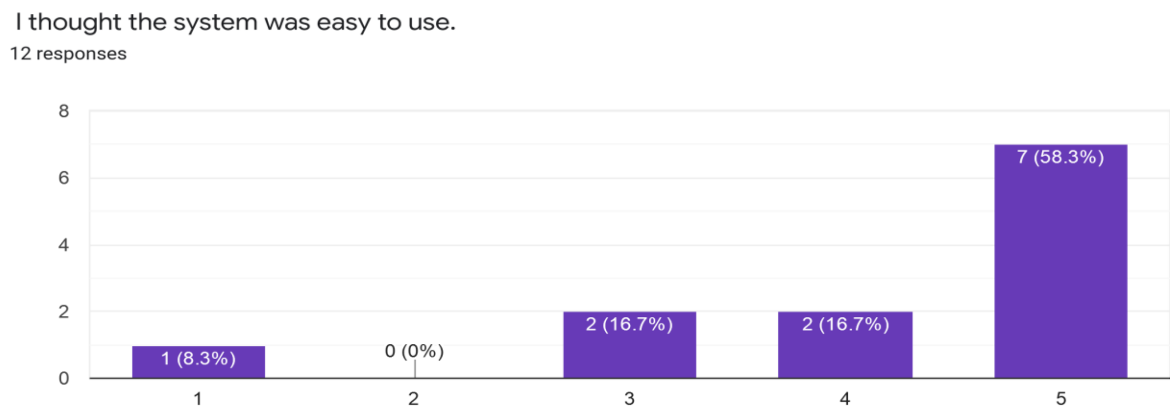
Below is the actual survey. In this part I analyse the results in more detail. The scale is from 1 to 5, where 1 means “not at all” and 5 means “very likely”.
In the first question, the answers were mostly positive, but some answers also tended to the middle. This can mean that these people would not use the system if it would not be mandatory to use in class.



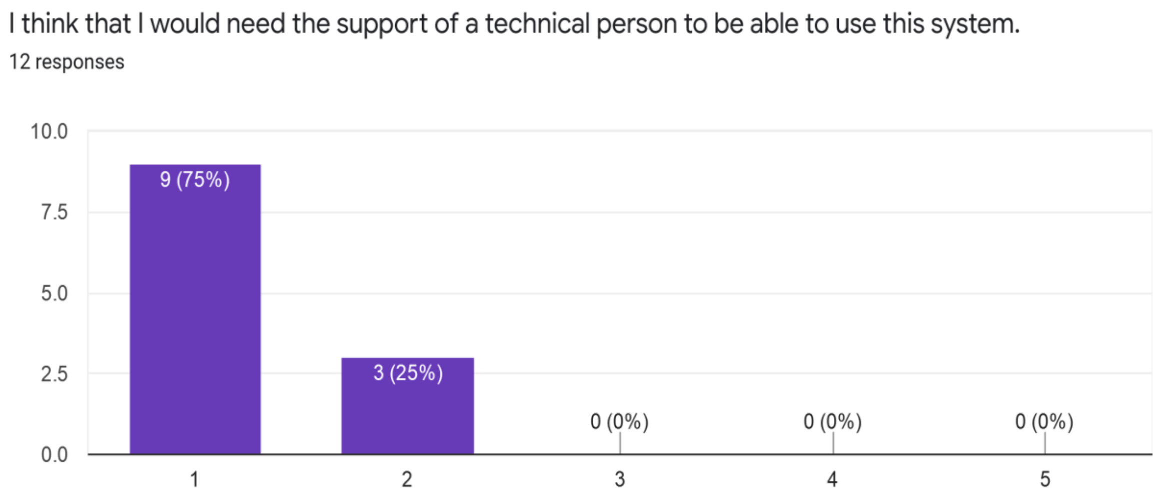
Next, most people did not find the system complex at all, and only 3 were neutral on the subject. This means that the system is not complex and can be used by most people.



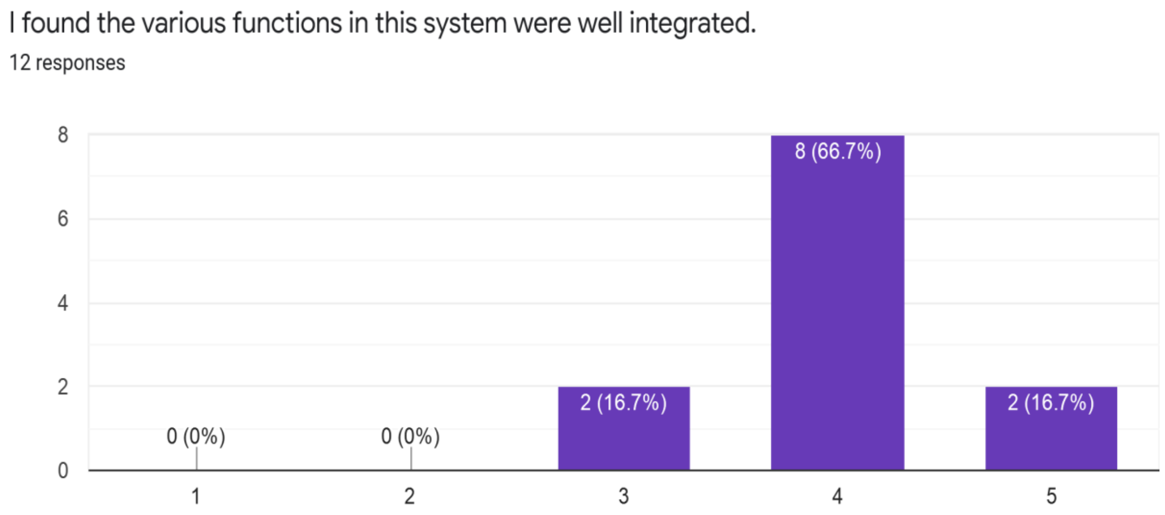
A very similar question to the previous one. One person had a difficult time navigating the web application. Perhaps a page with instructions on how to use the web application could help the person.



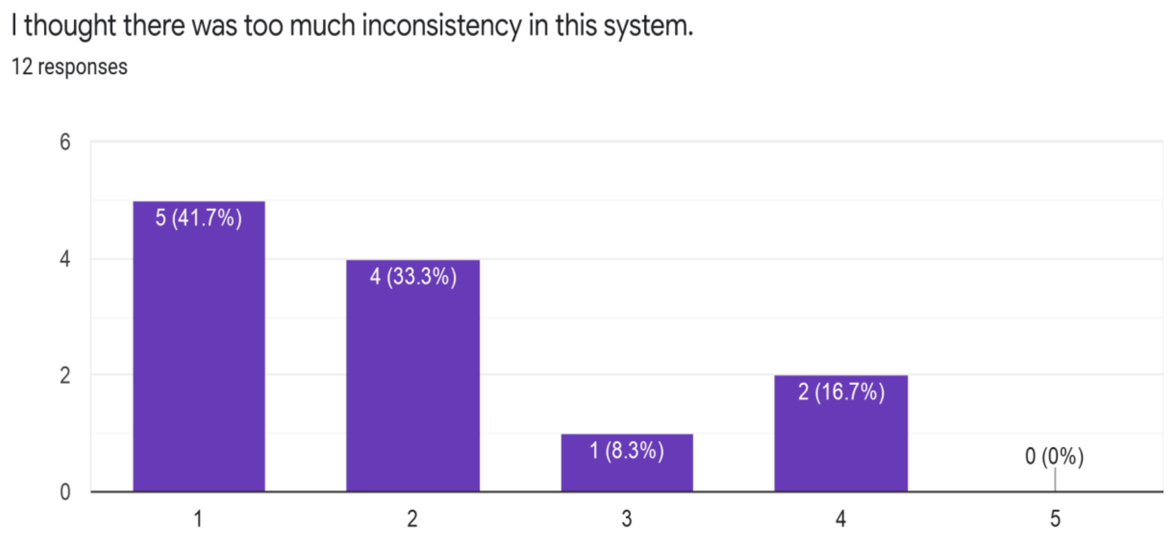
Next, the need for a technical person is not needed according to almost all the participants of the survey. This is a good sign, because it means that there isn't a big amount of technical complexity that would require the assistance of a technical person.



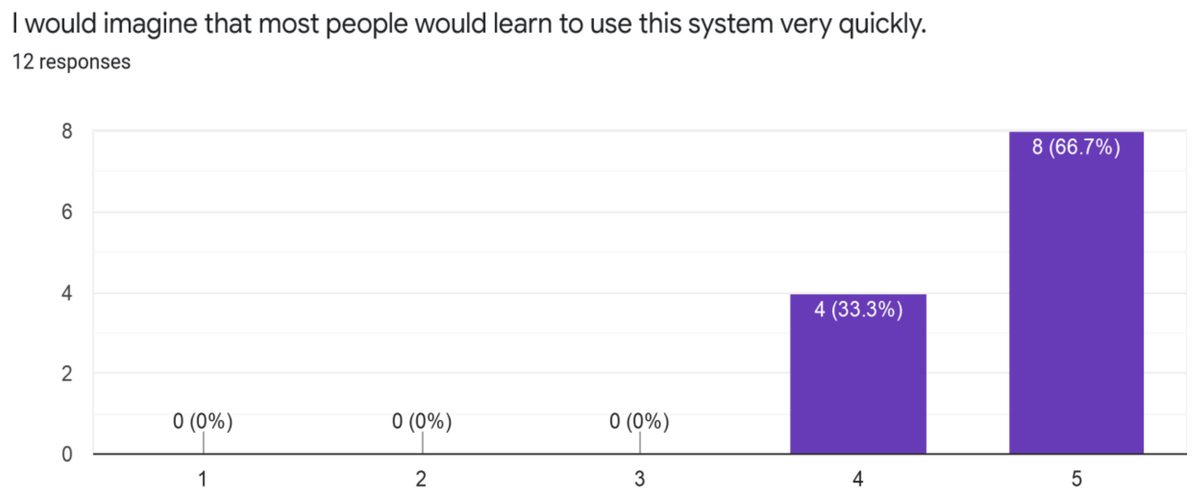
Next, most people agreed that the functions of the system were well integrated, but the 8 people voted for the second-best answer and not the "Very likely". This means that there can be an improvement to integrate the features of the web application a little more.



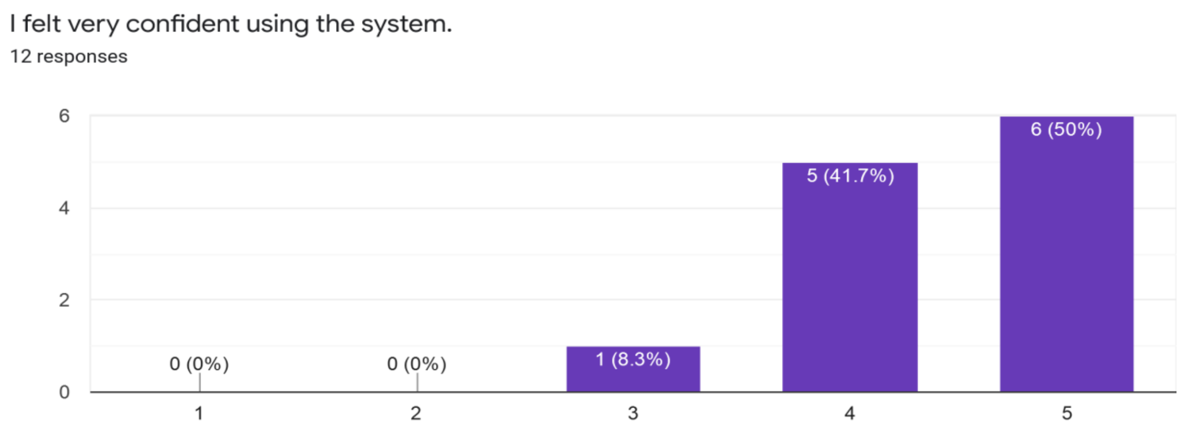
Next, most people did not find much inconsistency in the system, but there were some people that found it inconsistent.



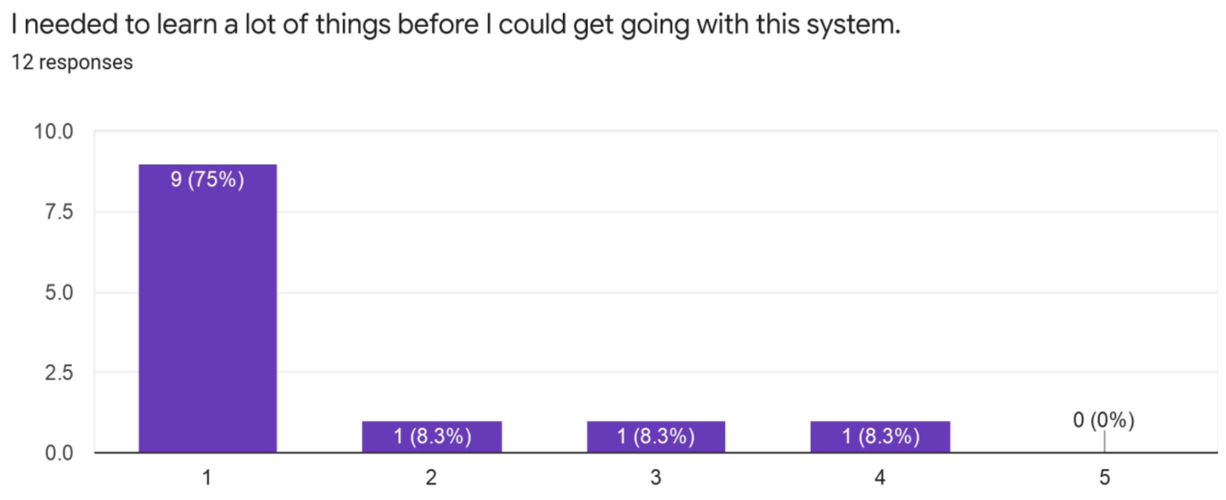
Next, almost everyone believes, from their experience using the application, that others who will try to pick up the application will not find it hard to learn to use. This means that future users, such as students will find it easy to learn the application.



Most users felt confident using the system, which is a very good sign.



Most users did not feel like they had to learn a lot of things before using the system and this is positive. It means that the web application is beginner-friendly, and everyone can learn how to use it fast.



Chapter 6

Conclusion

6.1 Summary

The aim of my thesis was the develop a web application that promotes active learning during lectures and at home. We always use technology for various tasks, so it is only natural that we use technology for educational reasons as well. Students are more engaged during lectures when they are engaged through their devices. With the use of a web application, the student will pay more attention during the class and have more motivation for study at home using the application.

I followed prototyping model of development which involved the creation of three prototypes. So, it was natural that I faced some obstacles during the process of creating the application. I had to learn new technologies and that took a lot of time and practise. This cost me a lot of time that could have been used for the final prototype. Unfortunately, I was not very familiar with the technologies and had to learn them in a small period of time.

The final prototype is running on the web and has many features available. You can create your own user and profile. Then you can navigate through the web application and create your own exercises for other users to solve (both homework and for in class). You can also enter the discussion forums and post your questions for other users to answer. You can also answer other user's questions on the discussion forums. The final prototype is scalable and new features can be added if needed. The web application was tested from me and other people, who have also answered a survey on the usability of the system.

6.2 Assess my work and future improvements

An improvement could be to have an administrator for the web application that can administrate from the website and not through the Mongo dB database. because now the system of pre assigning the administrators might not be the most ideal, and most commercial applications would not use such a system.

More features can be added to the web application. It would be nice if the application would use more multimedia options, such as graphical representation of automata or an even more difficult task of generating an automaton from a description given by the student. Another good feature would be to have an offline version of the application which could be used on the computer or smart phone.

There can be more interaction between users in the application. I have already developed the features to post and comment in discussion forums. Also, there can be options to share exercises or posts to a group of people. Furthermore, there can be virtual communities or teams between users of the application with the same issues about the class.

There can be more focus on the exercises as well. An idea is to have a flagging mechanism to highlight your favorites exercises or put tags on exercises with a search engine to be able to find the exercises easier. Furthermore, there can be a difficulty system, where users can rate exercises whether they are easy, hard or medium difficulty. The feature from my first prototype of having a series of exercises to solve (like a test) can be added to the third prototype as a new feature.

There can be features for the improvement of the student's academic performance. One way to do this is to put competition to the application. For example, there can be a points system, where a student who completes the most correct exercises gets higher points than someone that does not answer correctly or is not active at all in the web application. This can count on the final grade of the course. Finally, there can also be a leader board in the forums, where the student with the most activity or the most correct answers can advance to the top of the leaderboard.

Some other ideas include: Multimedia communication between users (using video or voice call), multimedia tutorials (tutorials for exercises), drawing of automata and Turing machines, more categories of exercises, autosuggestion of exercises based on previous solved exercises, university email verification (like in regular commercial applications).

Bibliography

[1] Android Authority. (2020, February 20). 10 best Android learning apps to increase your knowledge. Joe Hindy. from https://www.androidauthority.com/best-android-learning-apps-566227/

[2] Guru 99. (2019, May 20).What is a backend developer?. from https://www.guru99.com/what-is-backend-developer.html

[3] Hackr. (2020, April 09). What is frameworks. Viljaya Singh. from https://hackr.io/blog/what-is-frameworks

[4] YouTube – Traversy Media. (2019, January 09). Vue.js Crash course. Traversy Media Channel. from https://www.youtube.com/watch?v=Wy9q22isx3U&t=1522s

[5] Log Rocket. (2018, August 31). Why use Redux? Reasons with clear examples. Neo Ighodaro. from https://blog.logrocket.com/why-use-redux-reasons-with-clear-examples-d21bffd5835/

[6] Traversy Media YouTube. (2019, January 20). ReactJs crash course. Traversy Media. from https://www.youtube.com/watch?v=sBws8MSXN7A&t=5107s

[7] Traversy Media Udemy. (2019, July). MERN stack front to back. Traversy Media. from https://www.udemy.com/course/mern-stack-front-to-back/f

[8] Hallam. (2018, October 16). Website legal requirements: laws and regulations in the UK (2018). Susan Hallam. from https://www.hallaminternet.com/internet-marketing-and-the-law-legal-issues-affecting-you-and-your-website/

[9] Accessible Metrics. (2020, February 20). Top 8 Most Common Accessibility Issues to Avoid and Solve. Sam Stemler. from https://www.accessiblemetrics.com/blog/top-8-most-common-accessibility-issues-to-avoid-and-solve/

[10] Medium Extend. (2020, September 5). What is REST — A Simple Explanation for Beginners, Part 1: Introduction. Shim Vrehm Abraham.

from https://medium.com/extend/what-is-rest-a-simple-explanation-for-beginners-part-1-introduction-b4a072f8740f

[11] Frontend Maters. (Date) What is Frontend Developers from https://frontendmasters.com/books/front-end-handbook/2018/what-is-a-FD.html

[12] FreeCodeCamp Youtube (2019 June 25) FreeCodeCamp Learn the MERN Stack - Full Tutorial (MongoDB, Express, React, Node.js)from https://www.youtube.com/watch?v=7CqJlxBYj-M

[13] W3schools () HTML Tutorial from https://www.w3schools.com/html/

[14] W3schools () CSS Tutorial from https://www.w3schools.com/css/

[15] W3schools () HTML Tutorial from https://www.w3schools.com/js/

[16] W3schools ()Node.js MongoDB Create Database from https://www.w3schools.com/nodejs/nodejs_mongodb_create_db.asp

[17] W3schools () React Tutorial from https://www.w3schools.com/react/default.asp

[18] W3schools () JSON Introduction from https://www.w3schools.com/js/js_json_intro.asp

[19] Node Source (2017 February 17) An Absolute Beginner's Guide to Using npm from https://nodesource.com/blog/an-absolute-beginners-guide-to-using-npm/

Appendix A

The code of the first prototype is in this github repository:

https://github.com/harakisgeorge/theapp

The code of the second prototype is in this github repository:

https://github.com/harakisgeorge/EPL211_web_app

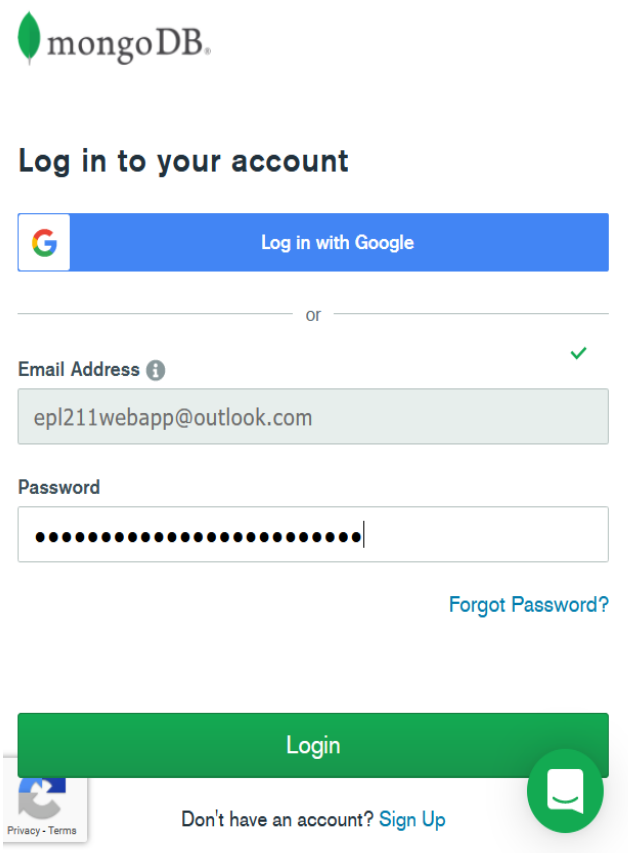
The code of the third prototype is in this github repository:

https://github.com/epl211webapp/social_media_app/

Appendix B

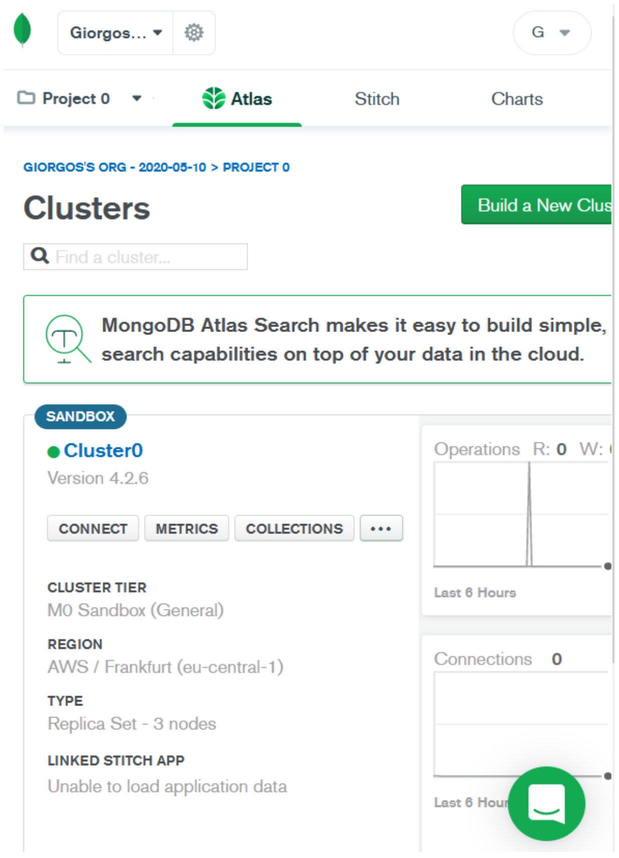
How to use MongoDB database.

1. Go to https://account.mongodb.com/account/login and login if you already have an account, otherwise register and then login.



epl211webapp@outlook.com

2. If your account is new, you will be prompted with a few options (e.g. your region) and then your first cluster will be created. Otherwise if you have already a cluster, you're presented with the following page, where you can view information about your cluster.


Giorgos...

G

Project 0 | Atlas | Stitch | Charts

GIORGOS'S ORG - 2020-05-10 > PROJECT 0

Clusters

Build a New Clus

Find a cluster...

MongoDB Atlas Search makes it easy to build simple,
search capabilities on top of your data in the cloud.

SANDBOX

Cluster0
Version 4.2.6

CONNECT | METRICS | COLLECTIONS | ...

CLUSTER TIER
M0 Sandbox (General)

REGION
AWS / Frankfurt (eu-central-1)

TYPE
Replica Set - 3 nodes

LINKED STITCH APP
Unable to load application data

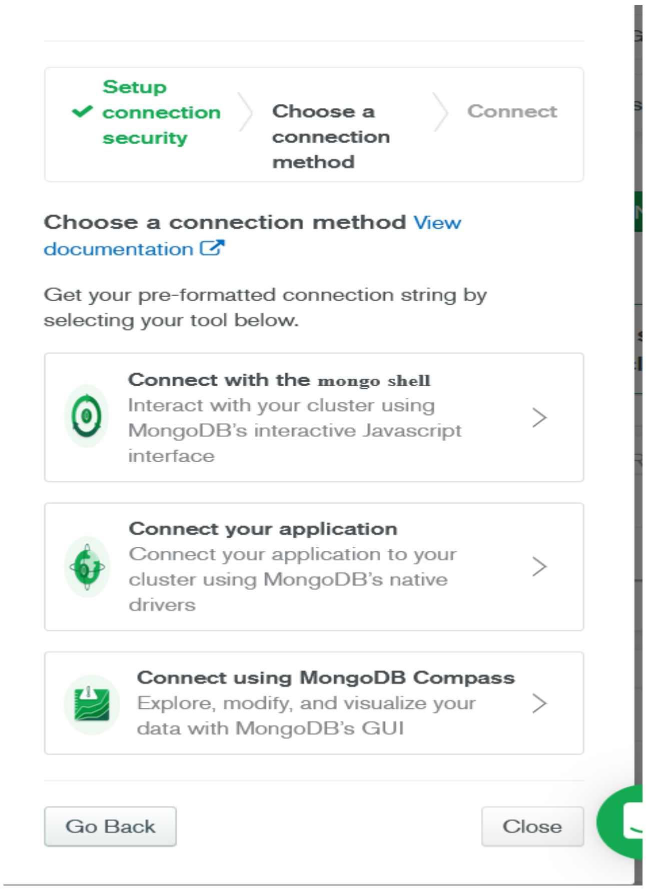
Operations R: 0 W:

Last 6 Hours

Connections 0

Last 6 Hour

3. In order to receive information about how to connect with your database you need to click on connect where you will be prompted with the following window.


Setup connection security | Choose a connection method | Connect

Choose a connection method View documentation

Get your pre-formatted connection string by selecting your tool below.

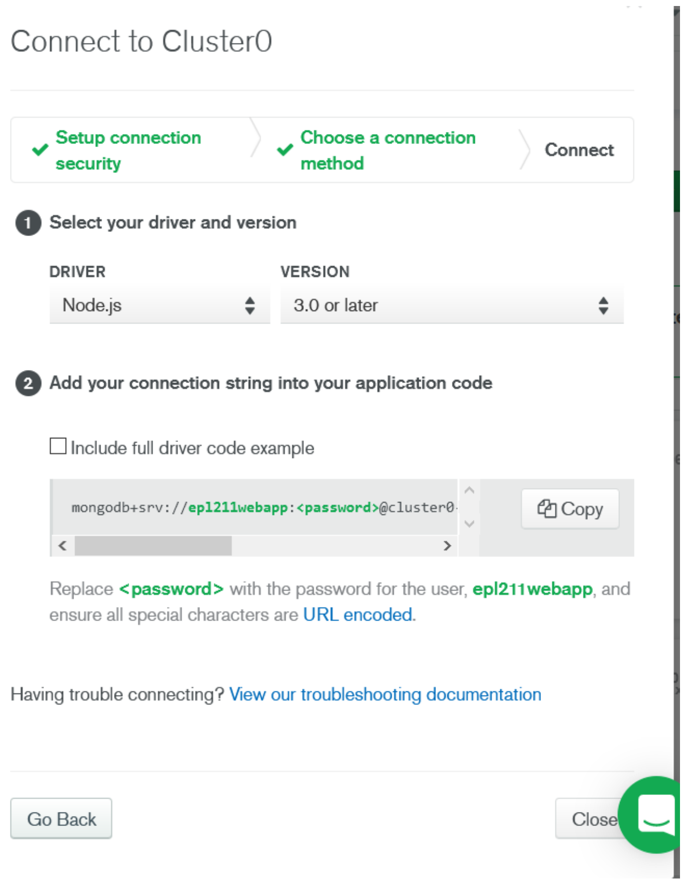
Connect with the mongo shell
Interact with your cluster using MongoDB's interactive Javascript interface

Connect your application
Connect your application to your cluster using MongoDB's native drivers

Connect using MongoDB Compass
Explore, modify, and visualize your data with MongoDB's GUI

Go Back | Close

4. We will usually choose “Connect your application” where you’re provided with a link, which you can copy and paste on the appropriate location in the code of your application. That link is the link between your code and your database, because it contains authentication information (username,password).


Connect to Cluster0

Setup connection security · Choose a connection method · Connect

1 Select your driver and version

DRIVER: Node.js
VERSION: 3.0 or later

2 Add your connection string into your application code

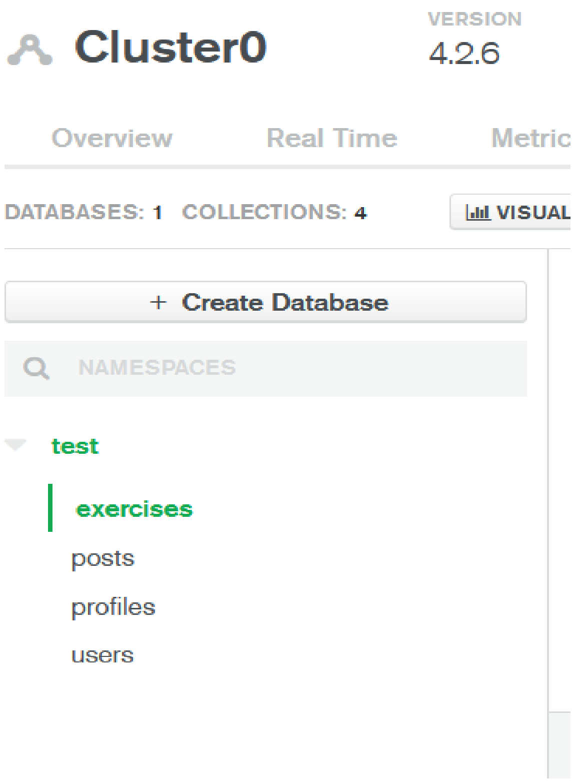
☐ Include full driver code example

```
mongodb+srv://epl211webapp:<password>@cluster0
```

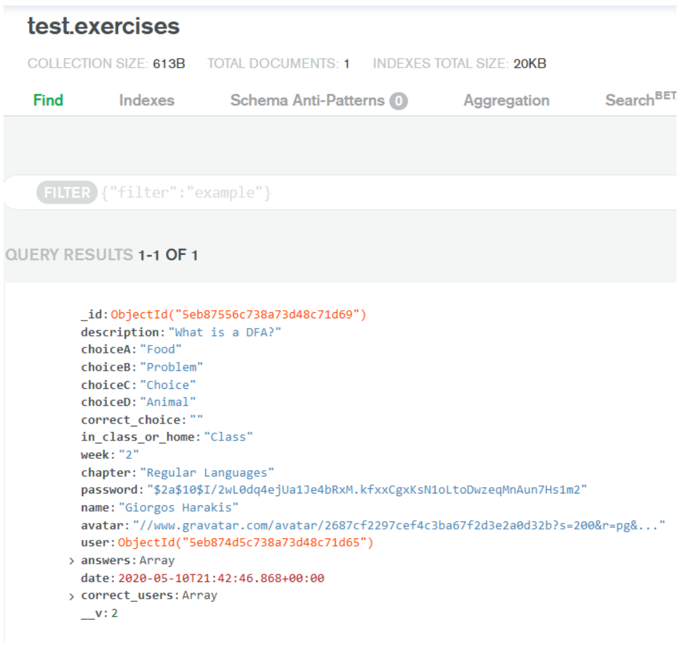
Replace <password> with the password for the user, epl211webapp, and ensure all special characters are URL encoded.

Having trouble connecting? View our troubleshooting documentation

5. Now if we’re already connected we don’t need to do this everytime. If you just want to check your collections you have to click on “Collections” from point number 2, so you can view your collections. Then in my case I can view the list of the collectiosn that are in the web application (exercises, posts, profiles, users).



6. If for example I click on “exercises” I can view in JSON format the contents of the exercises data collection.



```
  _id: ObjectId("5eb87556c738a73d48c71d69")
  description: "What is a DFA?"
  choiceA: "Food"
  choiceB: "Problem"
  choiceC: "Choice"
  choiceD: "Animal"
  correct_choice: ""
  in_class_or_home: "Class"
  week: "2"
  chapter: "Regular Languages"
  password: "$2a$10$I/2wL0dq4ejUa1Je4bRxM.kfxxCgxKsN1oLtoDwzeqMnAun7Hs1m2"
  name: "Giorgos Harakis"
  avatar: "//www.gravatar.com/avatar/2687cf2297cef4c3ba67f2d3e2a0d32b?s=200&r=pg&..."
  user: ObjectId("5eb874d5c738a73d48c71d65")
> answers: Array
  date: 2020-05-10T21:42:46.868+00:00
> correct_users: Array
  __v: 2
```

Note: Of course there are many more other packages that I used that I can’t explain here, but most of the packages I used are the same as the Udemy tutorial I mentioed at the beginning of the section. [7]