6CCS3PRJ Individual Project
Java Automated Feedback For Assignments

Final Project Report

Author: George-Cristian RADUTA

Supervisor: Dr. Andrew Coles

Student ID: 1310162

April 25, 2016

Abstract

In 1822, Charles Babbage invented the difference engine and set the beginning of a series of innovations. The computers to follow were based on a simple principle: use a given set of instructions in order to be able to perform certain tasks. However, the fast paced rhythm of evolution has now brought us to a point where users are capable of implementing algorithms which operate complex tasks that would be otherwise impossible for humans to execute. When it comes to the programming language they choose, given a wide variety of options, they tend to choose Java both for professional matters and educational ones.

The goal of this report is to review a web based platform which aims to help and guide students in learning Java throughout reliable resources provided by their educators while at the same time, enhancing the student-educator relation in a more efficient and interactive manner.

Originality Avowal

Acknowledgements

I would like to express my gratitude to Dr. Andrew Coles. His support, assistance, excitement and advice led towards a steady progression in the development of the project. Special thanks to the Department of Informatics and King's Entrepreneurship Institute at King's College London.
Furthermore, I would like to thank my family which support has led to where I am today and friends for the continuous backing and moral support over the course of this project.

Contents

Chapter 1

Introduction

The moment one starts putting things into perspective when considering their purpose in life, role in society and lifetime goals coincides with the start of a freewill journey. As one sets his/her ideals, skills are ready to be developed and lessons are waiting to be learned and in the hope for the peak of this ideal, one is faced with a choice of varied paths. However, one of the ways towards achieving a goal is to view work as a way of giving life a purpose, rather than acquiring wealth or as the Chinese philosopher, Confucius, would say: "Choose a job you love and you will never have to work a day in your life".

One of the stepping stones on this journey is represented by undertaking a degree in a higher education institution. This, leads the way towards exceeding limits and further exploiting new possibilities while pursuing one's goal.

It is obvious that we live in the Age of Information as an outcome of the digital revolution of computers, but it is also well-known the fact that man is the best computer. Therefore, everything with regards to technology is the result of our ability to build intelligent programs and systems in order to ease our lives. As Ammaar Shaukat Reshi states, "It can be argued that the computer is humanity's attempt to replicate the human brain. This is perhaps an unattainable goal. However, unattainable goals often lead to outstanding accomplishment." This faithfully reflects my motivation for getting involved into research that can affect society by using computers as an invaluable tool for human learning and development. Computer Science and Engineering offer a myriad of options when it comes to conducting research and developing ideas concerning the creation of programmes that contribute to the public good. This will emerge as long as a society based on technological advancement and students fond of mankind's high standards of life exist.

1.1 Project Motivations

When it comes to endeavours, programming sets itself to be as one of the most laborious ones. From hand sketching ideas to computer generated algorithms, the same principles apply: practice does make it perfect and the support from one's more experienced peers goes a long way. The one great resource that students have is the educators, whose experience and feedback are there to guide them through their learning journey. This experience has changed over the years along with the exponential increase in students number and decrease in the amount of available time of an educator. Now, to that, add the not so reliable internet resources and you get an unorganised and unproductive way of learning which shifts the balance towards spending more time searching for information rather than actually learning. On the other hand, from an educator's perspective the shortage comes from the inability to provide each student with a physical learning environment outside the class and assist them in their learning process. This is where the internet play its part.

During my time spent in different environments of education I observed a recurrent problem; students are always reticent about asking questions during the lectures or even during the tutorials. Because of this, the information which the educator wants to pass does not reach the student properly. The ability to freely ask for clarifications when a topic is not understood properly is a valuable skill not only in programming, but in all circumstances in life.

Having identified the ways in which these problems deeply affect a student's life and performance, I noticed the great need for something more interactive, innovative and capable of providing personal and instant feedback on students assignments, which materialised in a web based platform.

Furthermore, taking into account the diversity of choices when it comes to universities and their tailored courses it can be concluded that there is a great need for each of them to have such a platform. This would provide students the chance of getting automated and personal feedback on different modules or assignments. Thus, it will help them to understand the motives behind their approaches success or failure.

All in all, based on the principle that at the core of every achievement or skill lays practice and perseverance, my project seeks to provide students with the necessary framework and support to spark their curiosity and enrich their programming experience.

1.2 Scope

"Java Automated Feedback for Assignments" consists of two major components:

- an educator side on which teachers are able to propose assignments, evaluate students and analyse their submissions;
- a student side on which pupils are able to see and solve recommended assignments, evaluate their progress and compete with other students in the ranking system;

There are various concerns that have to be taken into consideration when developing such a platform as both student's and educator's perspectives have to be analysed.
On one hand, the implementation has to be simple, intuitive and yet comprehensive so that it will not be a time consuming task for teachers when creating a course or project.
On the other hand, the platform has to be complex enough to support and generate effective evaluations and feedback for students so that they would easily understand how their answers fit or not the marking scheme.
Moreover, one of the biggest concerns regarding such an implementation is represented by the security of the platform itself. Solutions for different aspects on this concern have been added with a more in depth narration in **Section 5.7 - Security**

1.3 Project Objectives

Questions, answers, practice and hunger for knowledge are the main concepts on which passion and success develop. To complement the idea of constant practice and to solve issues encountered in my early years as an undergraduate student, I have built a platform on which students can solve coding problems in order to gain a deeper understanding of the subjects taught and also submit their solutions to get automated personal feedback based on comparisons made among multiple tests and the professors solutions.
While academia and industry seem at times to be divided by a difference in perspective when it comes to this, I have decided to bring my own contribution to the matter and find a way to bridge this gap. Therefore I believe the skills students develop while completing assignments on this platform are particularly relevant to industry. Based on a topic taught, educators would be able to propose assignments on which students are expected to submit their solution and the platform would return them a mark varying from 0 to 100 points. Students will be able then to see how well they have done and access the leader board, that is a simple ranking system,

which purpose is to motivate students with competitive incentives among themselves.
Furthermore, limited time and resources represent a recurrent problem in academia and the project which I developed will facilitate a major shift in the ways of teaching and learning beyond Computer Science courses, but across different fields in university.

1.4 Why a web-based approach?

The project was developed as a web application for laptops and computers but it also features a mobile friendly view so that students and educators would be able to access all the core functionalities from their mobile devices too.
A web platform is a website on which various functionalities can be added for the user through external Applications Programming Interfaces (APIs). By developing such a platform, which would help students and educators in that matter various gained advantages were taken into consideration.
When compared to a native application, a web-application has the "edge". A statement based on multiple advantages described by Jason Summerfield in "Mobile Websites vs Mobile App". Some of the most important improvements include:

- immediacy - mobile websites can be accessed instantly;
- compatibility – it can reach users across different types of devices;
- upgradability – a website built dynamic it has more flexibility when it comes to updating the content;
- findability – as web pages can be found easily by users in their searches;

Chapter 2

Background

This chapter of the report contains a summary and an analysis of the background of the project. That is a short description on student practices regarding the use of existing platforms that encourage digital learning and found solutions based on pre existing libraries and literatures that aim to improve the presented platform.

2.1 Students trends regarding learning

Studies show that the average student spends more than a third of his/her time in academic spaces such as lecture halls, laboratories or libraries. The impact of these places is seen on their perspective, as they are keen on spending more time in such places which leads to a visible increase in their productivity and efficiency.

In the past, students were dedicating their time to researching information in books and written documentations, but nowadays this experience has drastically changed. Furthermore, the exponential increase in students number along with the decrease in the amount of available time of an educator requires classrooms to be adapted to students preferences in order to prevent their loss of interest. That is why it is imperative for academic institutions to try to balance students and educators desires.

Moreover, the usage of computers, laptops or mobile devices increases yearly with no restriction when it comes to age, especially when students are brought into discussion. Undergraduate and postgraduate students find it easier to search for information on the Internet rather than skimming through multiple books until they find the desired answer and nobody can blame them for that.

Unfortunately, not all Internet resources are reliable and in many cases these lead to an ineffective way of studying.

2.2 How students' practices can be improved

Next we will discuss about a couple of statements that we found most inspirational for the project. When considering what defines practice as "good" we view it as a structure that can be analysed and discussed based on multiple topics and subjectivities. We are going to focus on the ideas described in *"Seven Principles for Good Practice in Undergraduate Education"* by Arthur W. Chickering and Zelda F. Gamson.

One of the most important principles is represented by: *"Good practice gives prompt feedback"*, a phrase that immediately encourages a connection in our mind to our past experiences. Students are eager to find out if they are right or wrong and if it is the case of the latter they always want to be provided personal feedback which sadly in most of the cases is not possible.

Another principle reflected in the work conducted for this project is *"Good practice encourages active learning"*. By providing multiple assignments to students, they get to practice rather than learn by heart. This particular approach to learning, deeply reflects in the quality of assignments on algorithms or data structures as they tend to come up with innovative solutions.

"Java Automated Feedback for Assignments" was developed with the seven principles in mind seeking to improve the current ways of teaching and learning.

2.3 Students in Computer Science

In universities, each module is based on a structure that the educator considered to be the most efficient for teaching students. In this section we will consider only courses that are taught in Undergraduate Computer Science modules.

One of the most important subjects refers to Algorithms, because it has an extensive applicability in real world usage; from finding the shortest path to a certain destination, to calculating how the weather will be next week. Our society uses algorithms in every moment of their life but not everyone may realize how important they actually are and that is why students need to understand their relevance since their first year.

2.4 What is an algorithm and how can we evaluate them?

The majority of algorithms are linked to computer programs but there are other domains such as Biology, Physics or Engineering in which algorithms are discussed. In the following chapters we will refer only to Computer Science.
An algorithm in Informatics is defined as a set of rules and steps that need to be followed in order to solve a problem by producing a result, known as the output. Usually, it is accompanied by a list of values, an input, which is provided by the user or a secondary program. The notion appeared long time ago, but it was only in 1928 that the term has received its current meaning, when David Hilbert started to think of ways of solving the *“Entscheidungsproblem”(”The Decision Problem”).*
Algorithms can be classified by different characteristics with the 2 most important ones being:

- Time Complexity
- Memory Consumption

Time Complexity in a simple definition can be view as “how many steps does the program need to execute to get the desired result”. The amount of time an algorithm needs to finish executing it is not always know since it depends on the machine that is running on and on the actual input that is provided. Because of this, computer scientists usually discuss about the Worst-Case Runtime of an algorithm which is ”How long it would take for the algorithm to run if it were given the most insidious of all possible inputs?”.
Memory Consumption refers to an estimation of the space that is needed to run the algorithm expressed as the “Big O” notation.
Most of the times there has to be a balance between how fast the algorithm is and how much memory it consumes. Although, depending on the algorithm chosen there has to be a compromise on the two above such that time is exchanged for memory and vice versa.

2.5 Format of an assignment

Testing and Analysis are the two important aspects when projects involving numerous user functionalities are built and released. With that in mind, given my focus on the two mentioned aspects, the platform currently features only one format for assignments with future work to be done to allow users to add their own unique and personal structured assignments. More details about future work can be found in **Section 9.2 - Future Work**.

Assignments will consist of the following mandatory components which are visible to the students too:

a *Title of the Assignment*

b *Deadline of the Assignment*

c *Name of the Course the Assignment belongs to*

d *Assignment Statement* which aims to give a general description of the problem that the student has to solve. All details which are related to the solution of the algorithm or what particular cases need to be taken into consideration should be included in this component.

e *Input Format* represents a short explanation on what the rules are, the size of the input and the values processed by the algorithm built, such as minimum and maximum of characters.

f *Input Example* represents a small test case which aims to help the student have a better understanding of what the assignment requires and how the input values will look like.

g *Output Format* defines how the output, produced by the algorithm, is expected to look like. Particular cases such as adding a new line at the end of the example or adding a space character should be specified here.

h *Output Example* serves as an illustration of what values or characters the algorithm should produce. It can be followed by a short explanation of how this values were reached from the input.

The following fields are to be completed by the educator and will not be seen by students. Their purpose is to provide the platform the necessary data to produce personal feedback for students and also mark the solution:

a *Test Sets* are represented by a set of 3 values. The educator has to insert at least one such set so that the system can mark the student's submission. The recommended number of tests to be inserted is 10 so that the system could generate personal feedback for students that are attempting the assignment.

- *Input Values* represent what should be read by the user's algorithm when executing.
- *Output Values* represent what the program should produce after compiling and running the code.

- *Description* is an optional field which has the purpose to provide the student a small explanation of what the test set is about without guiding him to the actual solution.

b *Skeleton Code* is a mandatory field which contains the code provided by the educator for the assignment. It is a compulsory component as the platform will test the input values provided by the educator on the code and will verify that the output values provided are the same as the output values that are produced. If this condition is not accomplished the assignment will not be added to the system.

2.6 Existing platforms

What makes this particular platform unique is the research that has been done beforehand. Thus, observing all shortcomings in other pre existing platforms, its aim is to address all this issues and offer a complete experience when it comes to enhancing the student-educator interactivity.
On one hand, some implementations only allow for fixed format assignments to be uploaded, providing feedback to those who submit their code in a manner of “Correct” / “Wrong”. An example of such a platform is “HackerRank”. Although, it is great that users can find out if their submissions are partially correct and it might help in small coding assignments or coursework, if no personal feedback is provided they are not able to understand where is their submission wrong or they might lose their interest quite easily after a few attempts in which their solution keeps failing. Everyone needs to understand why their approach was wrong or what special cases were not treated by their implementation.
On the other hand, platforms like “Socrative”, are trying to help educators raise a question during the class and students have to answer to that question. It is quite difficult for a teacher to keep track of all the answers and some of the students might not get the chance to find out if there answer was right, wrong or sufficient. Moreover, the format on which questions and answers can be submitted is limited and it is not that practically.
Furthermore, there are applications such as “Codecademy” or “Code School”, that are trying to assist the student from the start to the end of the course. On such websites, users can find loads of courses which goal is to teach users about the syntax of different programming language. Unfortunately, these tend to be based on a single and not complex enough structure which means they are not accepting multiple solutions. In this way, students thinking is limited to a single method and they have to follow the steps that are provided even though there might

be multiple solutions to the course.

In conclusion, both educators and students cannot benefit too much from these platforms. Teachers need an implementation on which they can build their assignments based on their own methods of teaching while students need to be allowed to submit their own solutions in various formats. There is no platform that offers the possibility to integrate different assignments based on different topics which would suit and benefit any university. In order to provide a useful application, both student's and educator's experience and perspective have to be taken into consideration.

Chapter 3

Requirements and Specifications

The purpose of the project is to create a web application which integrates two types of users:

- “Type E” refers to users that are part of an academic course committee such as educators or teaching assistants. This category is allowed to create courses and assignments and can also access all students submissions and data.
- ”Type S” involves users such as students which goal is to solve assignments in Java as a programming language and most importantly to receive personal feedback based on some comparison algorithms.

Because of the two types of users, since the beginning of the project, it was clear that the need of evaluating, analysing and classifying two completely different perspectives was imperative. In order to be able to develop a useful and practical platform the first step was to get as much experience as possible to get a proper impression about different opinion regarding such a project.

3.1 Type ”S” User Views

- As a student, the need of having a platform containing all the materials at one click away it’s imperative, especially for a BSc Computer Science student;
- As a computer scientist, it would be great to have more feedback on the assignments. Usually the feedback we get is: “Yes/No” and the simple answer for the assignments and in most of the cases is not enough;

- As a student, I believe we could use more practice like Small-Group Tutorials or a platform like you suggest;
- As a student I would like to always have feedback based on my solutions and not on general solutions that are submitted;

3.2 Type ”E” User Demands

- As an undergraduate teaching assistant, it is difficult to explain to each student in particular why they do not have the complete or correct solution, so I would like a platform that would help me in that matter;
- As a Teaching Assistant, there is definitely not enough time to talk to each student and explain the solution. I would like to have a program that would automatize this work;
- I find it difficult to mark all students submissions and provide personal feedback. I would like to try to give them feedback based on some comparisons with the result of the coursework;
- There are tools that are helping educators in this matter, but the repeated problem that I find is that none of these platforms allows for the creation of a unique assignment. All are based on the developer structure;

3.3 Functional Requirements

Functional requirements are viewed as specific behaviour or functions of the implementation that aim to provide the user the proper experience and functionality.

3.3.1 Students’ Functional Requirements

- **SF1** – The user should be able to login on the system;
- **SF2** – View all the Courses he is part of;
- **SF3** – View and access all the Assignments he was assigned;
- **SF4** – Submit solutions to the Assignments Section;
- **SF5** – Receive personal, automated and instant feedback;

- **SF6** – Access all past submissions of the student;
- **SF7** – View all the relevant details of each submission;
- **SF8** – View a basic profile of other users;
- **SF9** – View the Ranking System.

3.3.2 Educators' Functional Requirements

- **EF1** – View and access a dashboard;
- **EF2** – View all the Courses created by users with such permissions;
- **EF3** – Create a new Course and assign permissions to other Educators;
- **EF4** – Edit/Delete Courses that he is owner of;
- **EF5** – View the Leader board ranking system;
- **EF6** – View all the Assignments;
- **EF7** – Edit/Delete Assignments the he is owner of or has permissions to do;
- **EF8** – View all Students enrolled on the platform;
- **EF9** – View all students' data;
- **EF10** – Access all students' submissions;
- **EF11** – Access others educators profiles.

3.4 Specifications

In order to guarantee a steady and efficient progress and an implementation with core functionalities embedded and tested, all the functional requirements were assigned a priority label (low/medium/high). In this way, high importance components were approached first and only after lower components were developed.

Table 3.1: Student Requirements' Specifications

Index	Specification	Priority
SF1	Users log in system was done through credentials (username and password) through HTTPs Requests under the encryption of an SSL Certificate.	Medium
SF2	Courses are provided to the students as a list which allows them to open/close the elements so that assignments of each course will be visible.	Low
SF3	Accessing Assignments can be done through a list of assignments under the course they belong to.	High
SF4	Every Assignment is built on a specific template which explains the requirements of it. Students are offered to either send their own Java File or to write the code in the specific Java Text Area.	High
SF5	After the Submission of the Solution, the user is provided with feedback based on algorithms comparisons which determine the correctness of the submission.	High
SF6	Access to the past submissions can be done through a table in which they all are contained. The user has the option to see limited details about it in the columns of the table (Index, Assignment, Date, Mark).	Medium
SF7	Students can view all their past submissions in a HTML Template page which for each will show: ID of the submission, date of the submission, mark, name of the assignment, code of the submission and feedback received.	Low
SF8	A student can view only minimal details about other students: name, username, total number of points and department.	Low

Continuation of Table 3.1		
SF9	The ranking system is a simple HTML Page which shows a table with students ordered based on their total number of points.	Medium
End of Table 3.1		

Table 3.2: Educator Requirements' Specifications

Index	Specification	Priority
EF1	A dashboard consists of a list of the last 10 submissions sorted in chronological order. There is the option for the educator to access all submission.	Medium
EF2	Courses are shown in a table with the following details: name, type, department, owner name, date created and 2 options: Edit/Delete. The options are only for owners of the course.	Medium
EF3	Only Educators are allowed to create courses by filling in HTML fields with the following details: Name, Type, Department and a list of educators that are allowed to add assignments to the new course.	Medium
EF4	Edit/Delete Functionalities are done by updating/ unlinking the details of the course from the databases of Users and Courses. Data is not permanently deleted and an activity log is kept.	Low
EF5	The leader board system is represented by a sorted list based on the total number of points of each student.	Medium
EF6	Assignments can be viewed in a table format which allows user to sort them by different criteria.	High
EF7	Edit/Delete Functionalities are done by updating/ unlinking details of the assignment from the databases of Users and Assignments. Data is not permanently deleted and an activity log is kept.	High

Continuation of Table 5.1		
EF8	Students are shown in a table format only for users with permissions by extracting only minimal data from the database.	High
EF9	Each student is assigned a profile with data about themselves, including data about all their submissions. Only educators are allowed to view this data.	High
EF10	Each submission is shown in an HTML Template Page which shows the educator the code that was submitted and the results.	High
EF11	Other profiles can be access with only minimal data to be shown.	Low
End of Table 5.1		

3.5 Non-Functional Requirements and Specifications

The Non-Functional Requirements section specifies how the system should behave and what the attributes of the implementation are:

a Performance

- The platform should efficiently get all data from the server and integrated it into the user interface.
- Alerts and Error Messages should be given to the user in case of back-end errors, HTTP requests errors, front-end errors or user input errors.
- No slowdown should be experienced by the user in case of multiple users using the platform simultaneously.
- No slowdown should be experienced by the user in case there is too much data to be shown. Data requests should be structured in packages.

b Scalability

- Depending on the specifications of the hardware components, platform should be able to scale up when high number of students use the system.

c Reliability

- The implementation should be built in such a manner that all components are reliable.
- Software errors or wrong user input should be accompanied by messages which would let the customer understand their action was not successful.
- The system should block any attempts of submitting malicious code.
- Backups are needed in case of hardware failures.

d Usability

- The platform should be intuitive and easy to use by both categories of users so that it will not require extra time for integration.

e Security

- The implementation it is built for students and educators within King's College London and it is expected to run on the university's servers which imply security as a major concern of the project. More security concerns were discussed in **Section 5.7 - Security**.

f Maintainability

- As the project is represented by a web-based platform, it should be built with a high maintainability. New features and functions should be easy to add to the system.
- Platform should be built with clear delimitations on the components so that elimination or addition of other libraries would be integrated with no harm to the platform itself.

3.6 Aditional Aims

The platform has been used as part of an Algorithms course held by me during the first part of 2016. Students were taught different topics about Algorithms in Computer Science which they afterwards applied in submitting solutions for an online competition hosted by this system. More details can be found in **Section 7.1 - Real World Usage**.

3.7 Limitations

Limitations of the platform:

- Assignments that can be created are limited to a specific format not allowing educators to create assignments based on their structure.
- Currently an assignment, as the title of the project suggests “Java Automated Feedback for Assignments”, only allows for one programming language, Java but it was developed so that other languages can be easily added in the future.
- Sources of the students are not compared in a plagiarism system.

Chapter 4

Design

The following chapter brings to attention details concerning the design of the platform and an in-depth look at how the system was structured. It also includes specific details about the user cases, components integrated and steps with technical choices followed by their correspondent results.

4.1 Before the actual design

Developing a platform for educational purposes can be considered complex and delicate as there are a lot of aspects and views that have to be considered. We can think of two educators from two completely different courses such as Informatics and War Studies. One of them might find using an educational platform quite easy while the other might have doubts about what is actually happening "behind the scenes". Because of that, a precise analysis have to be conducted before, in order to gain a deeper understanding when thinking of designing the project.
As it is stated in *"Educational Design Research"* by Susan McKenney and Thomas C. Reeves, "one of the most imperative motivations for initiating a design research is the actual desire of increasing the impact and relevance of the project for educational policy and practice".
Moreover, more than once it was argued and justified that those who do not seek to view educational projects from different perspectives, "do not strive to inspire knowledge with it" and are looking only after particular cases tend to receive more criticism once the project is sent into production.
Several experiences from various departments and different levels of education were involved in the interest of building a design not only for the Department of Informatics but a design

which in future would be easily adaptable and extended for other divisions. Following this, core objectives were established which will be presented in the next sub-chapter.

4.2 Design Targets

- *Speed and Efficiency* were two characteristics repeatedly expressed. Educational platforms tend to store and access a lot of data which if it is not managed properly, will slow down even a simple action of login or logout of the platform. This issue was tackled with specific functions that access only part of the data or make use of efficient searching algorithms.
- During the development of the project, multiple times the concept of *simple and intuitive design* of the implementation was tackled. Analysis of how much of an improvement would bring to the project even the slightest order of functions was carefully taken into consideration.
- *Easy Adaptation* - As it was stated above it was imperative that since the beginning of the project, the implementation would be developed on the idea of allowing future improvements. Thus, work will be added with no major changes on the base structure of the system. The code was structured in multiple functions, controllers, modules and files.

4.3 Use Cases

Use cases can be viewed as a series of related interactions between a user and a system. The Use-Case Diagrams (Figure 4.1 & Figure 4.2) represent what steps the user can follow to achieve his/her goals through the system functionalities in a minimal detailed format with more explanations showed in **Chapter 5 - Implementation**. Due to the fact that the system is built and designed in two separate sub-systems, the necessity of two diagrams has been expressed.

In order to access and use the web-application, the user will have to open the corresponding URL. Afterwards, a user-name and password will need to be provided to access the content of the application. If the user has not set up an account yet, he/she can register a new one by accessing the "/new-user" page on which, after filling the mandatory fields, platform access will be granted.

4.3.1 Type ”E” Users (Educators)

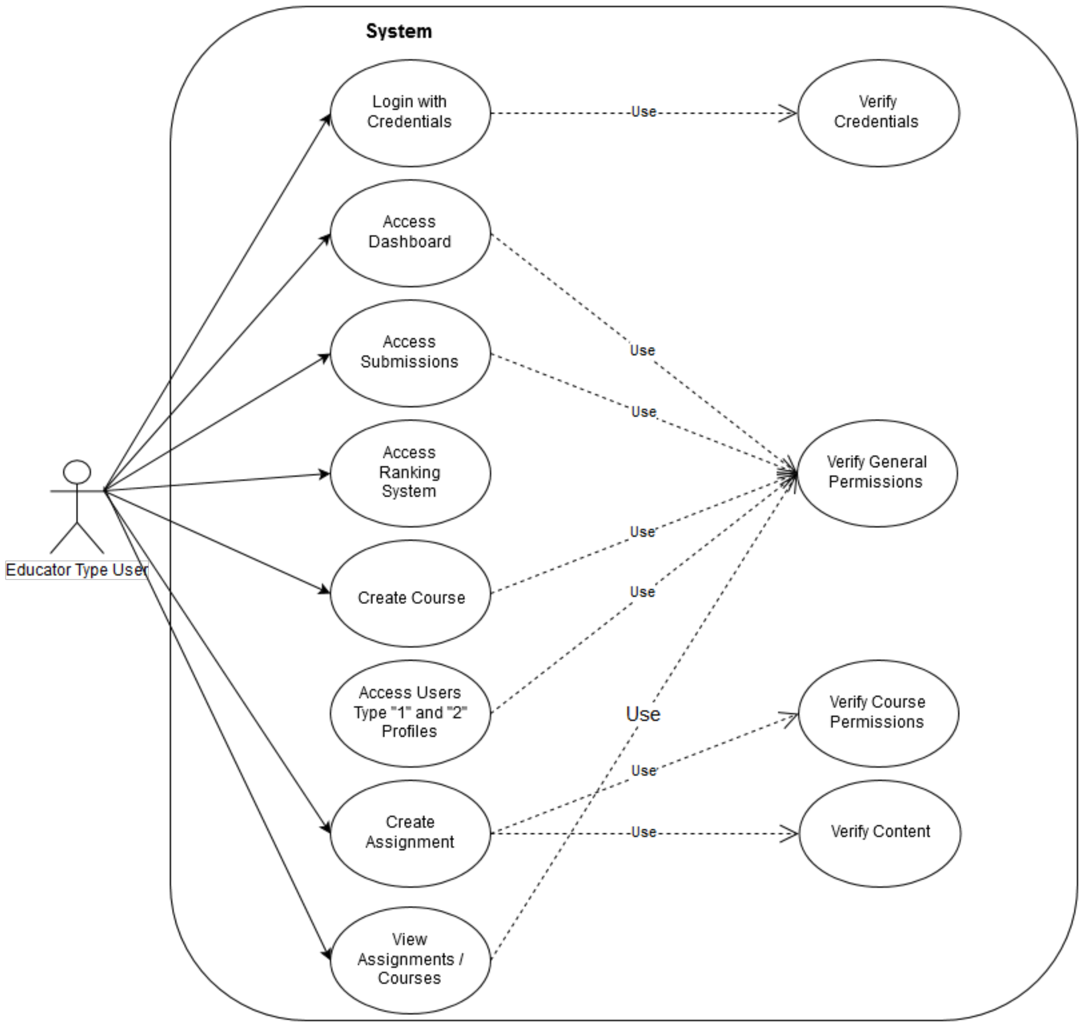


Figure 4.1

- After login, the user will be redirected to the “Dashboard” on which 10 students submissions will be available sorted in descendant order by the date of submission.
- By accessing the “All Submission” component, the user will be provided with all the students submissions in a table format. He/she has the option of sorting the table by:
  - ID – The unique id of the submission. To increase the security level, this ID (sub_id) is different from the ID from the Database.
  - Assignment Class Name – Represents the Java class name that the owner of the assignment created.
  - Mark – Number of points obtained on each submission.
  - Owner – User name of the student that solved the assignment.

  - Date of submission.

- A ranking system can be accessed by both students and educators. As the name suggest, this component will display the students sorted in a descendant order by their total number of points. Educators, unlike the students, have the possibility to access the student's profile from the ranking system and see all their details and submissions including feedback received on each test.

- Create Course – A simple form which asks for a few fields to be completed:

  - *Name of the Course*
  - *Department*
  - *Type of the course*(e.g ”Lecture”, ”Practice”, ”Tutorial”)
  - *A list of educators* to which permission of editing assignments will be granted. Only existing educators on the platform will be available for selection.

- View Courses – A table which contains all courses that currently exist. This table can be sorted both ascendant and descendant by the elements from the above sub-section “Create Course”. Only the owner of the courses is provided with the option of editing or deleting the course.

- Access Users Type “S” Profiles – It can be done by accessing the list with all students currently enrolled on the platform. From here the educator will be available to access basic details about the student such as the first name, last name and department he belongs to. He/she can also see and access all submissions of the student.

- Create Assignment – A more complex form compared to the one representing the “Create Course”. All fields and details about it can be found in **Section 2.5 - Format of an Assignment**.

- View Assignments – A table which on every row contains one assignment with columns consisting of the following details:

  - *Name of the assignment* - As the title suggests, this column shows the full name of the assignment;
  - *Course* – The name of the course the assignment belongs to;
  - *Date Created* – The date on which the assignment was created;

  - *Deadline* – The date starting from which the system will not accept submissions for the assignment anymore;
  - *Delete Function* – Only the owner of the assignment can remove it.

4.3.2 Type "S" Users (Students)

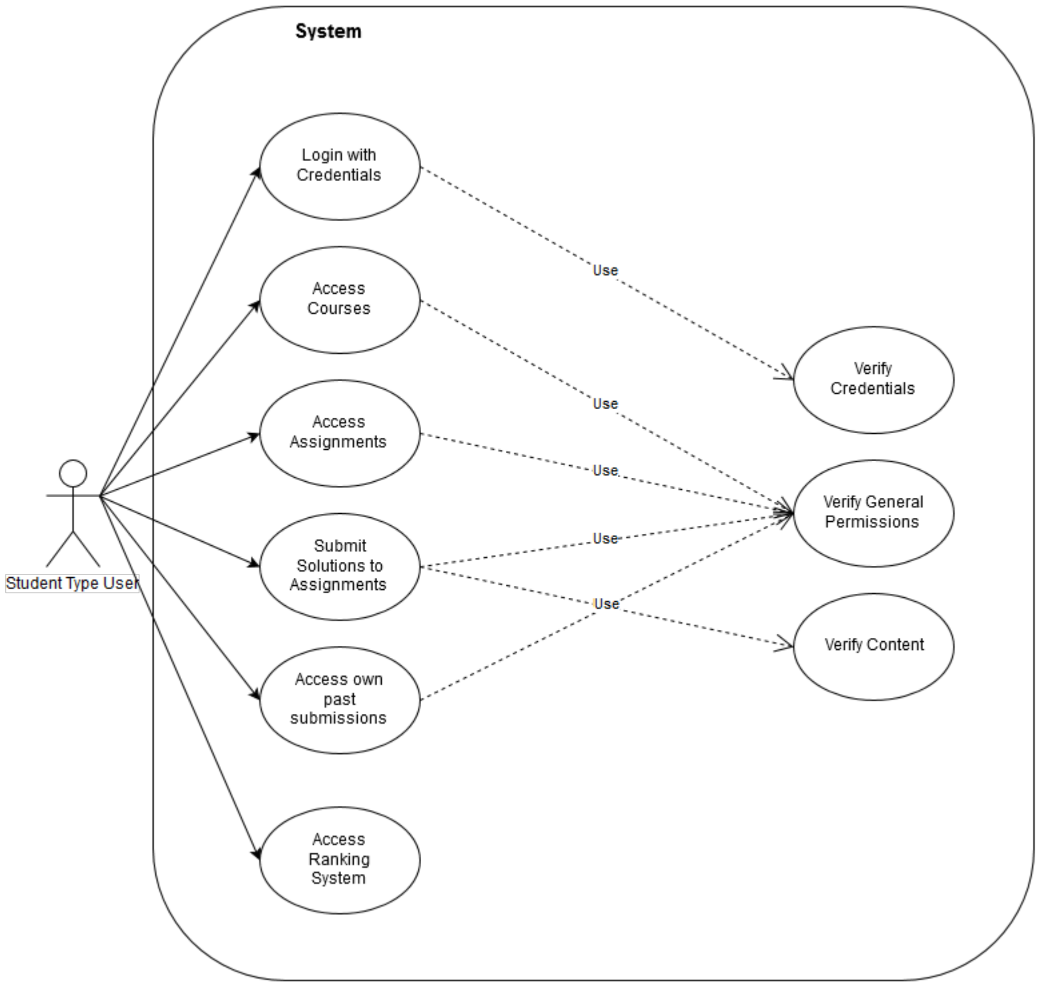


Figure 4.2

- *View Courses* – A list of elements which contains all courses that currently exist. The item are sorted in a descendant manner based on their creation order.
- *Access Assignments* – A set of assignments from which the user can select to solve.
- *Submit Solutions* – A simple template html page which contains all the elements of an assignment explained in **Section 2.5 - Format of an Assignment**. In order to submit a file, the student has to write the code in the special Java Text Area and then click

Submit. Feedback will be provided right after letting the student know what tests he passed or not.

- *Access own past submissions* – The student can view all his past submissions in the “mySubmissions” component. All details will be provided in a table format and can be sorted in both descendant and ascendant order:
  - *ID of the submission*
  - *Name of the assignment*
  - *Points* - The number of points based on how many tests the user managed to get right
  - *Date of submission*
- *A ranking system* can be accessed by students. As the name suggest, this component will show students sorted in a descendant order by their total number of points.

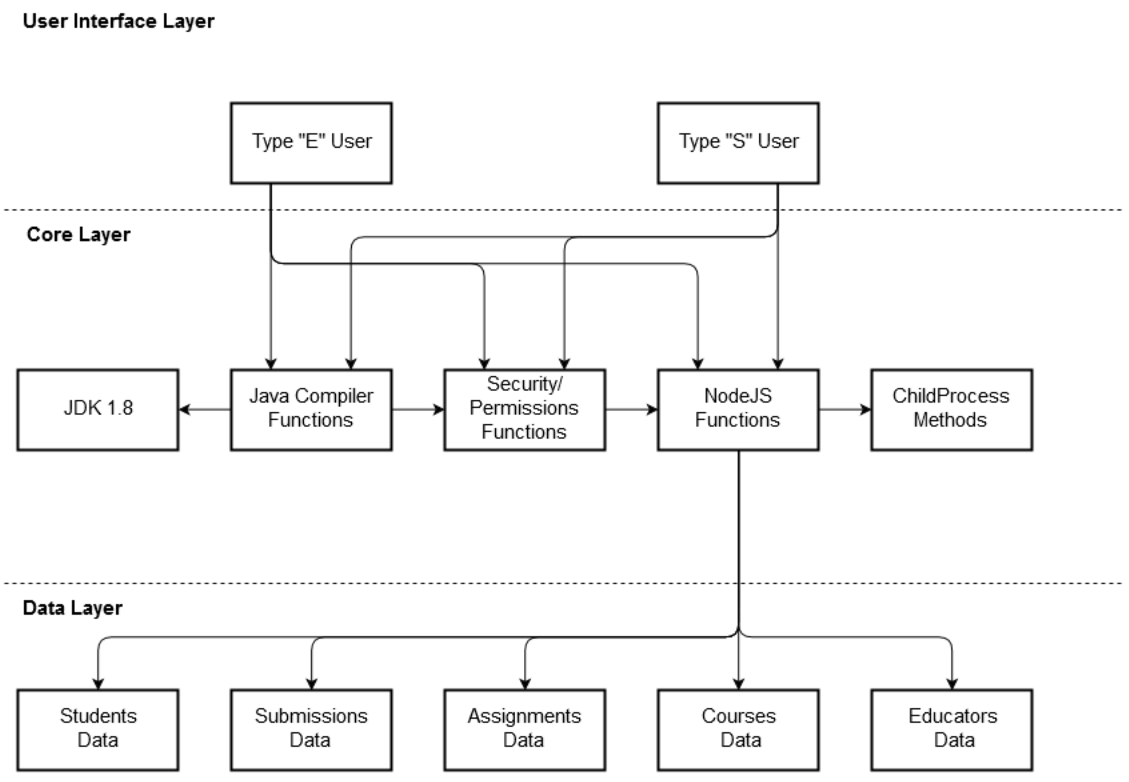


Figure 4.3

4.4 System Architecture

The System Architecture provides a conceptual model by partitioning the system in multiple sub-systems. An arrow is used to suggest the dependency between two systems which belong to different layers. The following partition, Figure 4.3, is suggested as it features a three tier architecture diagram which reveals the User Interface Layer, the Core Layer and the Data Layer.

As it can clearly be seen from Figure 4.3, there are 2 different User Interfaces with both having common and different functionalities. This approach has been chosen so that another layer of security could be added. Thus, 2 sub-platforms have been created with no direct access from one to another. The Core Layer represent the server functionalities which also includes different algorithms and methods used to evaluate users submissions and provide feedback. Moreover, with help of the NodeJS Library, the data layer is accessed. The Data Layer, consists of 5 different MongoDB Documents. All are stored in JSON Format allowing to easily extract data based on searching when needed.

4.5 Alternative Designs

In the past years, web technologies have evolved drastically and more and more developers prefer such an approach compared to a client side application for multiple reasons.

4.5.1 Client-Side Applications

Client-side Applications are those who require the user to download the project on their machine from the designated server. The application will run on the client side with data being extracted and saved from the server on to the 'home' machine.

There are various advantages and disadvantages that come with this approach but in this section, only topics related to the actual implementation of "Java Automated Feedback For Assignments" will be considered.

Firstly, we have to acknowledge what having a client-side application would bring in plus to the implementation of the project.

One of the biggest advantages is compiling and running the code on the user machine which would result in having no risks of accepting malicious code. If the user tries to submit code which is designed to harm the application then it will only affect his/her machinery.

Moreover, having all the necessary data already downloaded might be considered a big plus when it comes to the speed and efficiency of the platform. On one hand, the user would not have to wait for multiple requests between the server and his/her machine and he/she could easily use the application. On the other hand, because all data would be hosted on the client side, some users might try to take advantage of it.

JAFFA involves testing students submissions and comparing them with educators solutions and answers which are hidden and not supposed to be seen by the type “S” users. If a client-side approach is implemented then all these answers and solutions would also be downloaded on the user’s machine. One way of not allowing such data to be visible is encryption. As it has been proven over the years, obtaining strong encrypted data is not an easy task and it involves a huge amount of effort and time to do it.

Furthermore, the project involves a collection of additional knowledge on each student’s submission which would imply retrieving results from the client side and storing it on the server. As the code of the student is tested on his/her machine there is no guarantee that the server would get the genuine results. With such an issue, a marking system would not be possible because it would imply the high risk of retrieving false results from users that might try to trick the system.

Chapter 5

Implementation

This chapter will feature a more detailed and complex view on how the system was developed based on the designs plans described in **Chapter 4 - Design**. Furthermore, a more in depth explanation of core functionalities and the reasons behind their implementation will be added. Third parties libraries will also be described with some of their advantages and disadvantages being brought into discussion to support the decisions and selection that were made during the implementation process.

5.1 Programming Languages Used

As Dr Axel Rauschmayer stated in *"Speaking JavaScript"*, JavaScript is one of the most open programming language there is and one of the most widely used for several reasons.
Firstly, by conducting a simple research nowadays, we can find numerous documentations regarding libraries that can be used with JavaScript, errors that one user might encounter and most importantly answers and solutions of how we can avoid or fix bugs in our implementation. All the above materials come in different forms such as books, blogs, forums and are accessible to a very large range of users.
Secondly, if we have a look on the analysis regarding upcoming success of JavaScript it can be easily seen that there are high chances of having a bright future. The language is evolving steadily as it is supported by a large affiliation of companies and no single person or company controls it. JavaScript engines have had a huge growth and evolved from slow to fast compilers. Moreover, during the past few years, browsers have adapted to such development programming languages. As an example, Chrome, has made a magnificent progress and managed to integrate

IDEs and compilers in their browser with more improvement to come in the upcoming years.
In conclusion, I have chosen JavaScript as a main programming Language in both front-end and back-end implementations.

5.2 Development Path

As it is specified and shown in *Figure 4.3*, the system was implemented with 2 major sub-systems in mind, both being developed on a common 3 layer structure. The main objective of the project was to allow students to submit, compile and run their code while at the same time providing them with feedback based on some test cases. During the first term, the Type "S" UI was developed with core functionalities on the backend. It was only in the second term when the work on the second sub-system started. After a careful analysis on the work done until that time, there were a few improvements and errors that were noticed.
To guarantee a better implementation of the system, new libraries and methods were used with more details to be found in **Section 5.5 - Difficulties during implementation**.

5.2.1 HTTP Requests

"Hypertext Transfer Protocol" serves as a functionality that ensures passing data such as files, XML structures, JSON Objects, pictures, videos between a web browser and a server. There are multiple variations of them with POST, GET, PUT and DELETE methods being among the most used ones.
During the implementation, the above methods were used to pass JSON Objects from the server to the user and vice versa. Depending on which functionality was accessed by the user different structures were used, with all of them having the following mandatory fields:

- *Date of execution* - was stored on each action that the user made which would modify or access data.
- *User Data(UserID and User-name)* - was passed so that verifications could be made in the event of a crash of the application.
- *Action performed.*

The above fields were also stored in separate files which were not included in the Database used. Those files were visible only to the admin of the system and were intended to increase the security of the implementation. More details can be found in **Section 5.7 - Security**.

5.3 Back-end Libraries and Implementation

5.3.1 Digital Ocean

Digital Ocean is a cloud infrastructure provider focused on simplifying web infrastructure for software developers. In this implementation, it was used to host both the front-end (HTML, CSS, JavaScript files and Images) and back-end which would support running the NodeJS server, hosting the MongoDB Database and compiling and running code sources provided by the users. There are several reasons for choosing this provider.

Firstly, they provide "SSD-only cloud" which means that their servers use only high-performance Solid State Disks. Thus, their speed directly benefits the performance of hosted web applications.

Secondly, there is an active developer community with tutorials, Q&A sections and various pages of documentation constantly being updated and moderated.

Moreover, a Linux system would be provided which would ease the process of using SSH connections between the system server and King's College London.

5.3.2 NodeJS

Once the hosting platform was selected, a careful analysis was needed in order to choose an open-source environment for developing the server-side application. NodeJS is a JavaScript runtime that uses the V8 engine developed by Google. It is well known that it is a fast environment as the JavaScript code is compiled into the native machine code.

One of the main advantages of the platform is that it makes efficient use of the "event loop". The *"Event Loop"* consists of running processes synchronously on a single thread or running them asynchronously by spamming parallel threads to perform the task. This method is not efficient and consumes a lot of time and memory. A NodeJS application sends an asynchronous task to the event loop, together with a callback function and continues to execute the program. Once the task in asynchronous operation is done, the system will return to the process and execute the callback function.

Having the possibility of running both synchronous and asynchronous tasks at high speeds, storing or securing data by using already integrated libraries in the environment and implementation and having a large developers community were the main advantages that made NodeJS a suitable choice for this project.

5.3.3 MongoDB

MongoDB is a NO-SQL Database which means that there is no typical schema represented by tables and relationships among these. MongoDB uses the idea of objects as a storing principle, where an object can store different fields, contents and can vary in size.

The implementation of JAFFA involves storing objects such as assignments, users and submissions. The main reason for which such an implementation was chosen is the fact that such an object can vary in size when it comes to its fields. For example one submission from one student might be stored as:

- *Submission_ID* – Automatically assigned by the database.
- *Date of submission* – Date of when the student submitted the code.
- *Owner_ID* – ID of the user that submitted a solution.
- *Owner Name* – A string component build from the first and last name of the user.
- *Assignment_ID* – The ID of the Assignment from the database.
- *Code_submitted* – Code submitted by the user to be evaluated.
- *Mark* – Number of points obtained from a total number of 100.
- *Results* – contains an array with the results from the test:
  - *Index of test*
  - *Input*
  - *Output*
  - *User Output*
  - *Feedback/Message*

Each submission is stored as a JSON Object and can easily be extracted to be send via an HTTP Request to the user with certain verifications that are made on each request. Data can be send in a JSON format which is easily integrated to the front-end by using AngularJS.

The project involves multiple such Arrays that are stored in different documents. Among the most important ones we mention:

- *Students*
- *Educators*

- *Assignments*
- *Submissions*

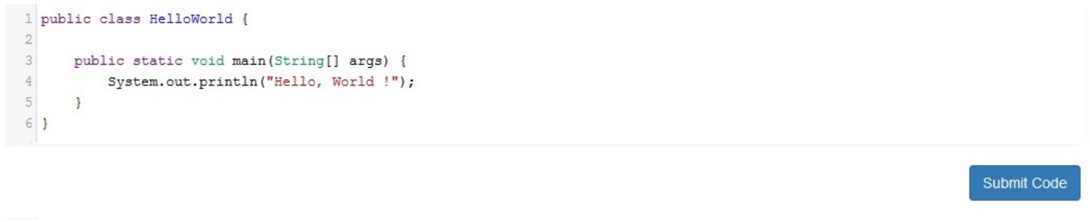
All database diagrams can be found in ***APPENDIX A*** with their corresponding components.

5.3.4 System Functionalities

In this section, we will discuss some of the cores functionalities of the system that are used in order to evaluate one submission and to create an assignment.

Evaluating Submissions

Type "S" Users are expected to submit Java code which is evaluated automatically by the system. The code can be submitted via the special Java Text Area. This component emulates the IDE experience/perception and is allowing the user to easily write code in a Java syntax format. For example, it will automatically close brackets and highlight specific words such as "static", "public", "String" and so on. In Figure 5.1, the Java Text Area can be observed with an example consisting of a simple program which is printing "Hello, World !" as an output result.



```
public class HelloWorld {

    public static void main(String[] args) {
        System.out.println("Hello, World !");
    }
}
```

Figure 5.1

After the user submits the code, data is sent to the server in a JSON Object with the following details:

- Date of Submission
- Owner_ID
- Owner_Name
- Name of the Java Class

In order to avoid testing malicious code, a dictionary containing names of libraries and objects that are thought to be harmful to the system has been created. Before compiling the code a short verification is done. If there are certain libraries that match elements from the dictionary then the owner of the assignment is notified and the code is not compiled. It is clear that this verification will not eliminate the risk of running malicious code and better options can be implemented. One alternative is to ensure that the code is ran on a virtual machine so that no harm can be done. More details about this approach can be found in **Section 5.7 Security**. If the code passes the initial test then the following set of steps is followed:

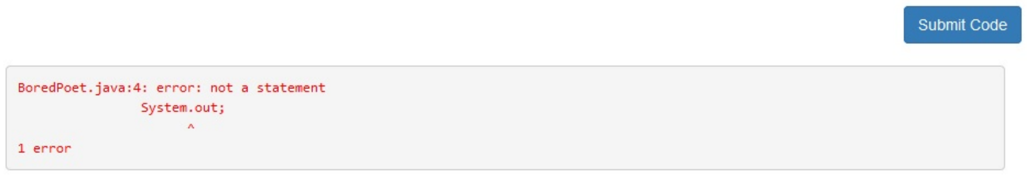
Table 5.1: Evaluating Student's Submission

Step	Command	Arguments	Description
1	mkdir (Linux Command)	-Name of new folder; -Path on System;	A folder is created on the server-side with the username of the person that sent the code.
2	writeFile() (Library method)	-Path on System; -Content;	A java file containing the submitted code will be created.
3	javac (Linux Command)	-Path on System; -Timeout:2000; -Array of Arguments;	The recently file created will be compiled.
4*	java (Linux Command)	-Path on System; -Timeout:2000; -Input; -List of Arguments;	Run the java file. Results of the running code will be stored in an array, in the database on both the Submission Document with complete details and also on the User Document with minimal details.
5	unlinkSync() (Library Method)	-Path on System;	The Folder, ".java" and ".class" files will be permanently removed from the server.
End of Table 5.1			

*Step number 4 is executed as many times as test cases there are for the assignment. *Timeout*: No code will be allowed to run for more than 2 seconds on the system to increase the level of security and to prevent infinite loops;

Input: Is represented by a String which is in fact the Input that the educator provided;

All the above steps might interrupt the whole set if there are any errors. In that case, a special message will be provided to the user with the Error Message and further instructions if necessary. Such an example can be viewed in Figure 5.2 below.



```
BoredPoet.java:4: error: not a statement
                System.out;
                      ^
1 error
```

Figure 5.2

Diagrams regarding how objects are represented and stored can be found in **APPENDIX A** For security reasons, an activity log file is stored outside of the database which is updated with details about the action done by the user such as date of action, result of the action and the user id. The same process applies to Type “E” Users when they intend to create a new assignment.

Creating an Assignment

Type “E” Users have the option of creating a new Assignment. The Action is complex and requires the completion of multiple fields by the user and verifications from the server. In **Section 2.5 - Format of an Assignment** there is a more in depth look of what an assignment is supposed to contain. In this chapter we are only discussing about the functionalities provided by the system.

To increase the percentage of reliable assignments, a verification on the code and set of test cases provided by the user is done. After the completion of all mandatory fields, the following set of actions is completed on both front-end and back-end:

a The content is verified on the front-end. No empty fields or fields filled with spaces are allowed. If a deadline was selected, a verification of today versus the selected date as a deadline is executed to guarantee that the deadline did not already pass.

b Data is sent to the server through an HTTP Request as a JSON Object.

c Once the data reaches the server-side, verification is done on the user. If he/she has no permissions of adding assignments to the selected course then the execution will be stopped with an error message sent back to the user.

d If the user has the necessary permissions then the process detailed above in **Evaluating Submissions** is executed.

e If the code provided by the Type “E” does not obtain 100% then an error message is returned with details about what tests were incorrect.

f If the score is 100% then the user is presented a preview of the assignment and the option of adding the assignment to the course is shown.

g If the user confirms, the assignment will be added to the Assignments Document and minimal details will be added to the User Document and Course Document.

h The Activity Log file is updated with the results of the action.

5.4 Front-end Libraries and Implementation

JAFFA is defined as an educational platform with the only way for users to access and utilize the data contained by the system being through a Web UI. As it was stated in **Chapter 2 - Background**, different perspectives has been taken into consideration to try to achieve a useful and yet simple design. Multiple readings and articles such as *“Don’t Make Me Think”* by Steve Krug repeatedly stated that the most important component of an application is the “face” of it. Any feature or action needs to feel intuitive to the user.

5.4.1 Web UI

The project is designed as a web application with an User Interface built upon various web libraries and components. This was highly prioritized among other components. The Web UI was designed as a single-page web application, written in JavaScript through AngularJS. Until recent years, the common way of developing a web application was done by using multiple HTML pages with each containing different functionalities. This is not an efficient way of doing it because there will be a large amount of components requested multiple times from the server. Lately, SPAs (Single Page Applications) became more and more popular due to multiple reasons:

- **Speed and Efficiency**: By using an SPA the user’s machine does not need to execute multiple queries to the server to download all pages as the application is represented only by one. During the use of the web-site, no reloading or transfer to another page is necessary as only the components to be shown are switched instead of a full page. This highly increases the interactivity and responsiveness of the application.

- **Increased user experience**: If the architecture of an SPA is used correctly, the user experience is highly improved due to the fact that compared to the traditional web applications, an SPA can provide the user with a more fluid experience, closer to desktop applications.

- **Maintainability**: Single-Page Applications are mainly designed on a Model-View-Controller (MVC) structure which helps increasing the level of abstraction. Requests to or from the server contain in most of the cases data in an XML or JSON format. SPAs are using the raw data to transform it in HTML which afterwards is used to update the Document Object Model (DOM).

5.4.2 SPA's Framworks

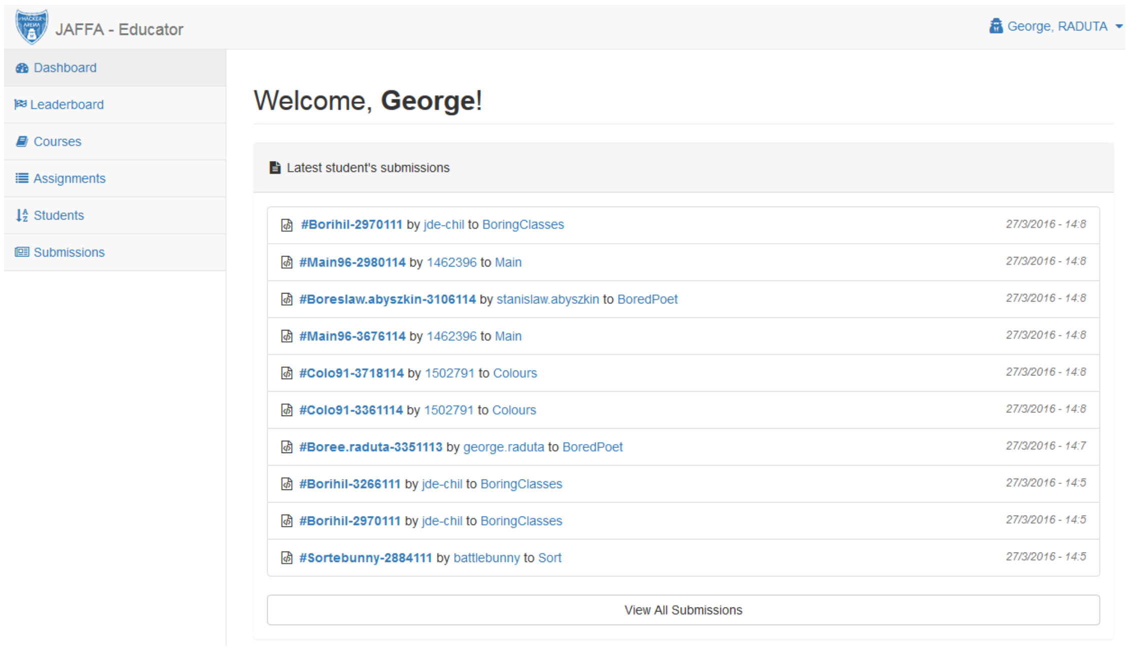
Due to the fact that nowadays, Single Page Applications are so popular, there are different frameworks that can be used for the implementation (React.js, AngularJS, Meteor.js).
AngularJS is a web-framework used for dynamic web applications developed by Google. It allows the use of HTML as a template language and extends HTML's syntax to express the application's components in a clear way. Two of its biggest advantages are data binding allowing to use HTML syntax to add, change or loop through objects and arrays and dependency injection which permit the elimination of duplicated code.
As most of the frameworks, AngularJS uses the MVC implementation by splitting the application in multiple components and serves as the bridge that connects them. Moreover, being a popular framework developed by Google, there are a lots of public components and documentations which help integrate already developed functionalities.
"Java Automated Feedback for Assignments" front-end was built using AngularJS mainly because of the data binding and MVC pattern. One of the main functionality of the platform is to show each assignment to the user. In order to do that, JSON Objects need to be passed from the server to the user-side and presented in an organized manner. AngularJS deals well with such requests, allowing easy implementation of HTML Templates. JAFFA was designed as a single web-page application. Each user case component was implemented based on a fix structure which contains:

- One module which is needed to specify the necessary injections with other modules.

- One controller which implements the requested functionalities of the component.

5.4.3 Features

Type ”E” User’s Dashboard



Welcome, George!

Latest student's submissions

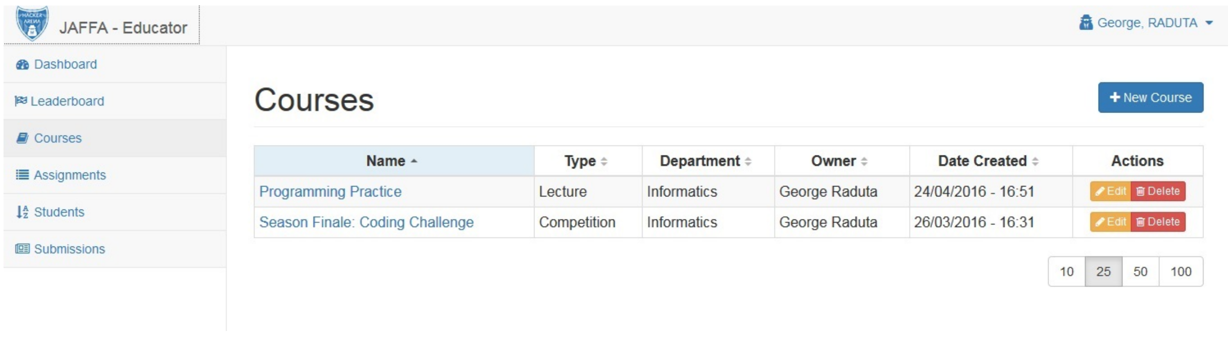
- #Borihil-2970111 by jde-chil to BoringClasses *27/3/2016 - 14:8*
- #Main96-2980114 by 1462396 to Main *27/3/2016 - 14:8*
- #Boreslaw.abyszkin-3106114 by stanislaw.abyszkin to BoredPoet *27/3/2016 - 14:8*
- #Main96-3676114 by 1462396 to Main *27/3/2016 - 14:8*
- #Colo91-3718114 by 1502791 to Colours *27/3/2016 - 14:8*
- #Colo91-3361114 by 1502791 to Colours *27/3/2016 - 14:8*
- #Boree.raduta-3351113 by george.raduta to BoredPoet *27/3/2016 - 14:7*
- #Borihil-3266111 by jde-chil to BoringClasses *27/3/2016 - 14:5*
- #Borihil-2970111 by jde-chil to BoringClasses *27/3/2016 - 14:5*
- #Sortebunny-2884111 by battlebunny to Sort *27/3/2016 - 14:5*

Figure 5.3

The Dashboard provides information regarding the last submissions made by the Type “S” Users. The last 10 submissions are shown in a list format through the use of AngularJS data binding; this allows a loop through an array by using HTML Syntax. By using the “ng-repeat” syntax, it will automatically iterate through the ‘lastSubmissions’ JSON Object Array and print its components in the specified format. As it can be seen, it also allows us to save space by not rewriteing every component each time we need it.

```html
<ul class="list-group clearfix">
   <li href="" class="list-group-item" ng-repeat="submission in lastSubmissions">
      <i class="fa fa-file-code-o"></i>
      <a href="">&nbsp;<b>#{{submission.subId}}</b></a> by
      <a href="">{{submission.knumber}}</a> to
      <a href="">{{submission.challengename}}</a>
      <span class="pull-right text-muted   small"><em>{{submission.date}}</em>
      </span>
   </li>
</ul>
```

Courses



Courses

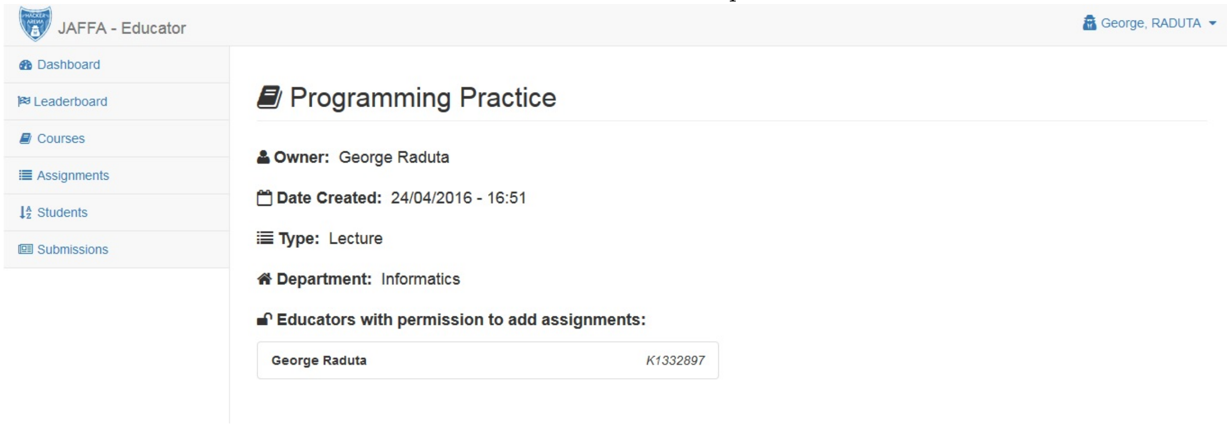
Name	Type	Department	Owner	Date Created	Actions
Programming Practice	Lecture	Informatics	George Raduta	24/04/2016 - 16:51	Edit Delete
Season Finale: Coding Challenge	Competition	Informatics	George Raduta	26/03/2016 - 16:31	Edit Delete

Figure 5.4

The Courses section provides minimal information about each course allowing at the same time the user to sort the table based on the following columns:

- Name
- Type
- Department
- Owner
- Date Created

In order to be able to retrieve more details about the course, the Type "E" user has to click on the name of the course and it will be transferred to the next component with full details about it.



Programming Practice

Owner: George Raduta

Date Created: 24/04/2016 - 16:51

Type: Lecture

Department: Informatics

Educators with permission to add assignments:

George Raduta *K1332897*

Figure 5.5

5.5 Difficulties during implementation

Due to the complexity of the project and the amount of work needed by the implementation of the platform different difficulties were encountered.

In order to evaluate and provide feedback to each user, a set of functions needs to be executed. One step compiles the Java File while the next one assumes the compilation was already done and is trying to run the code. As described in **Section 5.3.2 NodeJS** one of the most important advantages of this server-side library was the asynchronous execution of the functions which in this case was not a suitable option. To guarantee that *'java'* command was properly running, the execution of *'javac'* needed to be finished.

Unfortunately, the platform was trying to run both commands at the same time which would then cause: "ERROR: Could not find or load main class".

Initially a 'hard coded' option was implemented. Before running the *'java'* command, the system would wait 2 seconds, time in which the execution of *'javac'* would have been completed. Usually, compiling a java file does not need that amount of time but there are worst case scenarios where more time would actually be needed. This would mean that the user would have to wait another 2 seconds beside the already existing time and thus create an inconvenience. Fortunately, in January 2016 a new version of NodeJS was released, v.12, in which the current library "Child Process" integrated the execution of Synchronously Functions. In this way, the system would not have to wait 2 seconds to run the next step and it would to it by itself once the prior step was completed.

5.6 CAKE (Collecting Additional Knowledge)

The main objective of the platform is to allow students to test their code. In this way they can get automated feedback and improve their solutions. To help in that matter a new core functionality was added: CAKE

The Collecting additional knowledge feature on each student's submission is implemented by keeping all data of a student's submission and then allowing both the owner of the assignment and the owner of the solution to visualize it. There are 2 main advantages that led towards implementing it.

On one hand, educator's will be able to analyse the submitted code by running it through a plagiarism system and keep track of students that are not playing fair and try to copy others students' work. Moreover, by seeing students' files they have the possibility to look for logic

errors and try to explain it in particular or during lectures and tutorials.
On the other hand, students have the opportunity to revisit assignments after a period of time and try to improve their past submissions by having a look at them.

5.7 Security

The aim of the platform is to be installed and integrated into King's College London servers which are expected to contain sensitive data. Thus, it is important that certain levels of security are implemented. This objective was labelled as 'high priority' in the design of the project in both front-end and back-end.

Front-End

The project involves various interactions between the user and the system. The most common one is filling in various fields with data. Each field is verified not to be empty or filled with spaces before the execution of any commands. In this way it is guaranteed that the system will not receive **null** values which might produce errors in the back-end server. There is also an extra verification during login-in with the user field being checked to match the King's College London 'k number' format.

SSL Certificates

Communication between the system server and the user-side is done via HTTP Requests. It is well known that a common problem with such implementations is the possibility for a third party to intercept the data. To prevent this, an SSL Certificate has been acquired which means requests are now HTTPS. If a malicious system would try to intercept or redirect the traffic through another network, it would not be able to read it unless it has the server's private key.

Testing Code

Blindly testing a user's code would represent a big risk. While it is true that data kept on the system is minimal when we refer to the user there are users that might try to reduce the functionalities of the system by submitting malicious code. In order to prevent that, a small dictionary which contains names of libraries and practices that are considered to be harmful was created. Before compiling the code a short verification is done. If there are certain words that match elements from the dictionary then the owner of the assignment is notified and the code is not compiled. It is clear that this verification will not eliminate the risk of running malicious code and better options can be implemented. A better option would be to test each submission on a virtual machine instead of doing it on the server. In this way, if the code would be malicious it would not harm the system. In the worst-case scenario the virtual-machine would break, but not the system. It can be argued that there are ways of reaching the system throughout a VM, although in the past couple of years there has been a change in their implementation. Instead of running the VM on top of the operating system of the server, the VM is installed and ran on the Kernel level, guaranteeing no access to the system itself. An example of such an implementation is done by "Bromium". To increase the chances of not running a malicious code for too long, a time-out argument has been added when the code is tested. If the code enters an infinite loop or has malicious purpose, it will be automatically stopped after 2000ms.

Activity Log Files

It is always important to know which actions are executed on the platform and what users are doing them. Thus, as another security level, I added Activity Log files which are storing data in the following format about the user's actions. These files are not stored in the database and can be accessed only by the admin of the server. Separate files are stored for Type "E" and "S" users. An example of how data is stored in the activity log files can be seen below.

```
!-> NEW_USER: k1332897 at 3/2/2016 - 18:13 <-!
!-> LOGIN: k1332897 at 3/2/2016 - 18:14 <-!
!-> SUBMISSION: k1332897 on challenge : WelcomeHackers at 3/2/2016 - 18:15 <-!
!-> SUBMISSION: k1332897 on challenge : WelcomeHackers at 3/2/2016 - 18:16 <-!
!-> SUBMISSION: k1332897 on challenge : WelcomeHackers at 3/2/2016 - 18:17 <-!
!-> SUBMISSION: k1332897 on challenge : WelcomeHackers at 3/2/2016 - 18:17 <-!
!-> LOGOUT: k1332897 at 3/2/2016 - 18:19 <-!
```

Chapter 6

Professional and Ethical Issues

”JAFFA” was designed and implemented for use in the professional environment, thus a various amount of issues have been taken into consideration.

Firstly, as it is stated in **Code of Conduct & Code of Good Practice** and well known by the developers community, using material without authorisation is considered illegal and unethical. A great attention was paid to the use of external libraries. Each of them was carefully checked to ensure their use complies with their Policy and Terms and conditions. All Open Source Code or Libraries user are stated in the Reference Section. By using public libraries, the implementation process has hugely benefited as multiple functionalities were not developed from scratch, thus increasing the pace of the development and the quality of the project.

Secondly, security and privacy were highly prioritized in the development of the project and careful attention was given were needed.

6.1 Research Ethics

A good implementation of a project cannot be achieved without properly testing. That is why the platform was tested multiple times by users which were King’s College London students. For this purpose, a course was run by me through KCL Tech Society regarding Algorithms and Data Structures in Computer Science.

Furthermore, in order to be allowed to get feedback from students, an application was done through REMAS – Research Ethics Management Application System. The approval was obtained and it granted me the following benefits:

- Ethical Approval is granted for a period of time of 1 year.
- Feedback from users in questionnaires format.

Feedback it is considered one of the most important tools when developing a project. By getting real feedback from students, not only did it improve by expressing the need of new functionalities but also it got people to understand how to take full advantage of the project ‘s features.

Chapter 7

Testing

To guarantee a quality product with great functionalities, testing was imperative to the project. The implementation involves various components which needed to be tested separately or as a whole and in this chapter, different types of testing will be discussed.

7.1 Real World Usage

In January 2016, which marked the half of my project allocated time, a student side platform with the core functionalities (SF) was released and so, I decided to run my own course through KCL Tech Society.

Build X: Algorithms

Based on the principle that at the core of every achievement or skill there is practice and perseverance, my project seeks to provide students with the necessary framework and support to spark their curiosity and enrich their programming experience. Thus, over a period of 10 weeks, I have been organizing weekly workshops for King's College London students across all departments in which I am teaching them various Algorithms and Data Structures. I booked a room in KCL Strand Campus and each Monday starting from 06:00PM until 08:00PM, I have been teaching students from various years and different degrees about Algorithms. As this was mostly a theoretical program and for students not to lose interest, I decided to integrate my project within the course. Students were assigned 2 challenges each week and the student with the most points obtained was given an award. It was a great surprise to see that in the first week since the platform was announced more than 30 people created an account and started coding. Different educators and teaching assistants were also approached in order to help teaching students about algorithms such as: Dr. Andrew Coles, Prof. Luca Vigano, Lecturer Martin Chapman and Postgraduate Teaching Assistant Christopher Hampson. By launching the platform to real world usage, I was able to gather data in the activity log files with errors that users had encountered. After analysing it, the errors were addressed and fixed.

Season Finale: Coding Challenge

Weekly testing the platform was of great help and a lot of the Type “S” Users’ functionalities were assessed . Unfortunately, performance, a non-functional requirement, was not tested up to the limit as the platform was not intensively used by more than 5 users at a time. That is why I decided to organize an end-term competition on which students would be able to compete by solving challenges based on Algorithms and Data Structures. It was at this point, when I also collected feedback from students by using Google Docs.

By this time, the educator side was also complete and so I asked 5 committee members from KCL Tech Society to test it, to add assignments for the competition and to provide feedback. There were more than 40 students that participated in the competition and the platform ran with no errors during it.

Some of the feedback received during the event included:

- This was a great initiative and the platform works great.
- These assignments really make us think about algorithms.
- I came to your course for the whole term because it was a course on which I was actually learning to code.
- Great platform, but would it be available in more programming languages?

Results of the testing revealed no major issues encountered by both types of users.

7.2 UNIT Testing

Test Driven Development, also known as TDD, is one of the testing techniques most used in agile implementations. It offers various advantages throughout the development of a project including:

- Ensuring certain parts of the system are working as expected. In multiple cases, methods implemented depend on other parts of the system and that is why it is imperative to make sure that the data that is passing through the system is the one expected by each component.

- Make sure that data is remaining in the system and it is not modified. By running tests on the format of data that is received would guarantee that third parties did not interact with it and remained in the system.

- Ensuring that functions are working properly as expected to.

It is well known that Web Applications, especially SPA's have been difficult to unit test and developers actually had to use real-world testing quite a lot. Fortunately, some components such as HTTP Requests can be tested. In order to test this particular components, ”Mocha”, a simple and flexible JavaScript Test Framework' have been used. At the end of each new package of functionalities added, both unit and real world testing were applied to guarantee that the results were as expected. For example, when running ‘java’ files, the character encoding was different compared to the one supported by the browser's. By running tests, a different ASCII Code representation was discovered. Had this not been discovered, submissions would have obtained different results and would lead to marking no points for it.

Chapter 8

Evaluation

The following chapter will provide an analysis of the final product by comparing the functional and non-functional requirements stated in **Chapter 3 - Requirements and Specifications** with the implementation and testing obtained.

8.1 Functional Requirements

- **SF1** – The user should be able to login on the system.

  This requirement was partially completed. In the initial request, users should use King's College London credentials to log in. The verification should have been done through SSH channels which would allow testing the username and password on the university's systems. Unfortunately, due to limitations of the chosen server-side library, opening children processes were not allowed and an alternative solution was chosen. Users needed to create accounts stored on the project's server.

- **SF2** – View all the Courses he/she is part of.

  Requirement was accomplished by providing all courses in a list. Each element can be expanded to see the assignments contained by that course.

- **SF3** – View and access all the Assignments he was assigned to.

  Assignments can be accessed through the Course Section.

- **SF4** – Submit solutions to Assignments.

  Requirement was fulfilled with users being able to submit solutions in Java through the special Java Text Area.

- **SF5** – Receive personal, automated and instant feedback.

  Type “S” Users are receiving automated feedback in a matter of seconds within the format “Correct/Incorrect Answer” to each test case and detailed explanations if provided by the user.

- **SF6** – Access all past submissions of the student.

  Submissions can be accessed via a table which contains minimal details on each row about one assignment.

- **SF7** – View all the relevant details of each submission.

  More details are available to users with a single click on the Submission ID from the table specified above.

- **SF8** – View a basic profile of other users.

  Users can access their profile from the top corner of the application.

- **SF9** – View the Ranking System.

  The Ranking System is available to all users revealing only the username of users.

- **EF1** – View and access a dashboard.

  A dashboard is provided to each Type “E” User. This contains the last 10 submissions of the Type “S” Students.

- **EF2** – View all the Courses created by users with such permissions.

  The Courses can be visualized in a table format with minimal details on each row. More details can be seen by clicking on the name of the course.

- **EF3** – Create a new Course and assign permissions to other Educators.

  Only Type “E” Users are allowed to create courses with verifications implemented on both front and back end sides.

- **EF4** – Edit/Delete Courses that he is owner of.

  Only the owner of a course is allowed to edit/delete the course with verifications added.

- **EF5** – View the Leaderboard ranking system.

  Compared to the Type “S” User, the “E” type is allowed to view details about the user that is in the ranking system by simply clicking on the user ID.

- **EF6** – View all the Assignments.

  All assignments are visible in a table format.

- **EF7** – Edit/Delete Assignments the he is owner of or has permissions to do.

  Only deletion of an Assignment is allowed. The delete function was not implemented due to complex modifications that were needed.

- **EF8** – View all Students enrolled on the platform.

  A special section was created so that Students could be organized based by different criteria such as First Name, Last Name, Number of Assignments Attempted, Total Score, Number of Submissions and Rating (A result based on a formula of how many assignments were solved in a certain number of submissions).

- **EF9** – View all students data.

- **EF10** – Access all students' submissions.

  All Type "E" Users are allowed to view all users' submission with a fully detailed profile on each of them.

- **EF11** – Access others educators profiles.

  Requirement was not implemented.

8.2 Project Evaluation

In this section, non-functional requirements will be discussed based on data collected during the "Season Finale: Coding Challenge".

For this project, as it was described in **Section 5.3.1 Digital Ocean** machines were chosen for the server-side. The minimal size was chosen with the following specifications:

- 512MB / 1 CPU
- 20GB SSD Disk
- 1000GB Transfer

During the competition, the system proved high performance. For a period of time of 3 hours, more than 40 Type "S" Users were submitting solutions to the challenges, with 5 Type "E" users watching and analysing live their results. Even though the specifications for the server were minimal, the platform worked well with no slowdowns despite us receiving more than 820 submissions.

The scalability of the platform has high expectation based on the tests results. It was proven that the project does not consume lots of resources when a large number of users are accessing it. Based on the hardware components, the scalability will improve.

During the course of the term, the platform was maintained and continuously developed with no consequences on the student's work. This proves high maintainability with new functionalities being easy to be added.

Performance and Scalability are represented in the figures below with peak values representing a number of aproximately 45 users using the system at the same time:

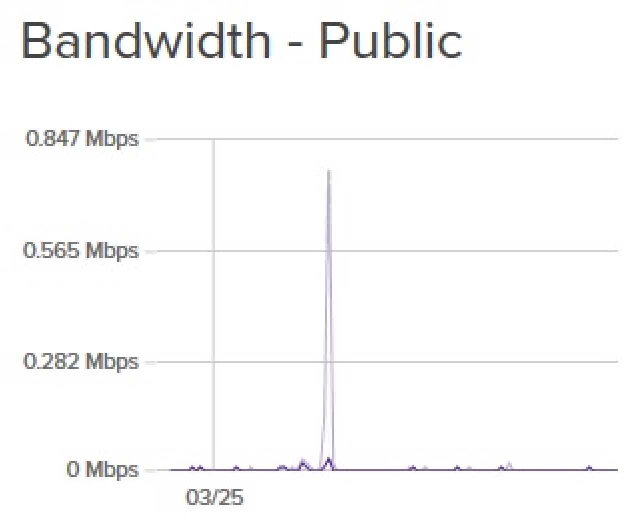


Figure 8.1

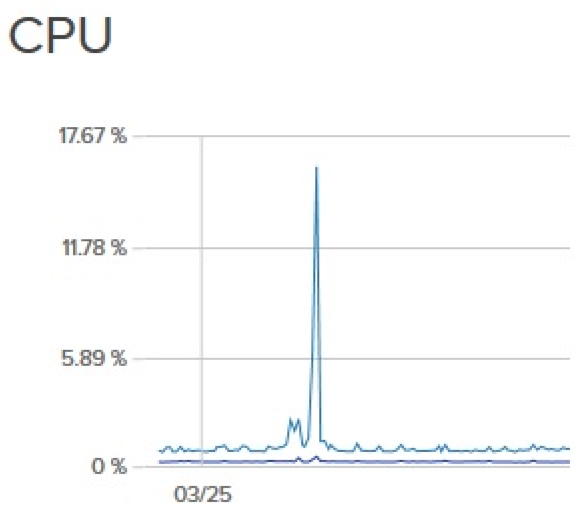


Figure 8.2

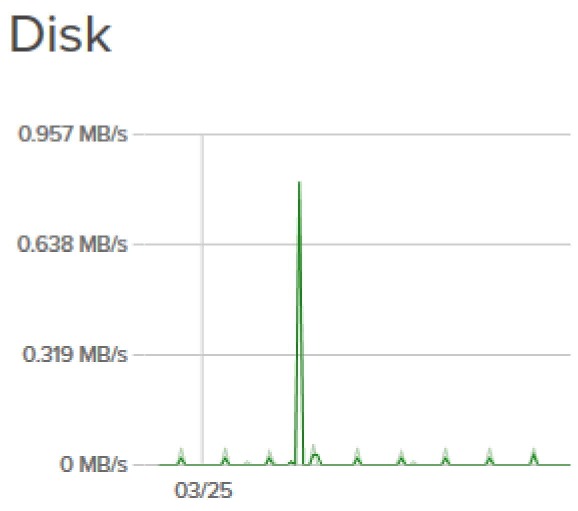
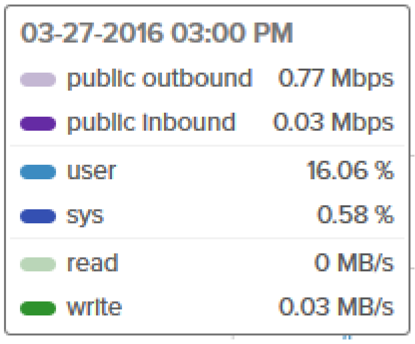


Figure 8.3



03-27-2016 03:00 PM	
● public outbound	0.77 Mbps
● public inbound	0.03 Mbps
● user	16.06 %
● sys	0.58 %
● read	0 MB/s
● write	0.03 MB/s

Figure 8.4

Chapter 9

Conclusion and Future Work

9.1 Conclusions

While this project started with an extensive research and analysis on the theory behind learning techniques and web applications approaches, it soon became a blueprint of real life issues, of bridges between different perspectives and various requirements. This materialised in the end into the implementation of a customized, complex platform that suits both educators and students.

With the advantage of an insight in a student's needs and a supportive lecturer who offered me a glimpse into an educator's perspective, the start of the project revolved around different papers on new technologies and general feature coding. All throughout the project I have discovered time and time again the importance of genuine user input. Starting with the educators, I have understood the role that time plays in their academic life.

Acknowledging the constant balance they need to maintain between research, teaching and administrative tasks I fabricated their side of the platform as an efficient way for them to offer the same qualitative feedback to the constantly increasing number of students. This had greatly impacted the existing approaches on teaching as several barriers have been crossed, such as the physical one (where an available university learning environment is no longer required in order to explain a task).

As a two faced coined, the student's component came with its own challenges. And as they became more and more divergent, the solution I found to cover them all was to teach a course myself. Over the course of this, I benefited from live user testing and rating, thus I managed to gain a deeper understanding in what other platforms lack and what this one should excel

in. By being able to provide reliable, approved material, the course was a stepping stone in observing user behaviour and plan for a future machine learning feature.
Mirroring the interaction between students and their professors, “Java Automated Feedback for Assignments” overcomes the difficulties in the mentioned relationship and triumphs in its usefulness and applicability with an enormous impact on both types of users. In the course of the years, this will be replicated to other subjects, thus leading a creative shift in the academic experience.

9.2 Future Work

The project has brought into attention the diversity in teaching and learning approaches found nowadays in academia. Out of all the features the platform disposes at the moment, the most interesting part to be further developed, from my point of view is represented by the feedback algorithms. A great deal of complexity lies in the mechanism of these algorithms, that can be addressed by implementing machine learning structures. Furthermore, to add to this, the tests and descriptions provided by the educators have an incredible impact on the learning outcomes.
JAFFA represents just a small part of a fully functional platform across all the university’s modules. Developing a customized interface for each and every module will for sure be a great challenge. This will greatly increase the number of users (both educators and students) and will bring along the challenge of finding a common ground between such a varied pallet of options. Furthermore, as complex as it will be to be developed, as much benefits it will bring to multidisciplinary students. Creating an online platform which would integrate multiple subjects and on which students would be able to solve homework, assignments and challenges provided by a reliable and certified source such as their educator, will most definitely increase students productivity, passion for learning and feed their vision and hunger for discovering
Moreover, turning back to educators and their requirements and expectations from this platform, future work should take into account the amount of time they put into this platform. The current project aim is to automatize the repetitive part of their job, while not depriving students of an insightful feedback. As new technologies and approaches are discovered everyday, the educator’s solution which feeds the machine learning algorithm could be furthermore tailored in an efficient way for the educator and a comprehensive way for the student.

Bibliography

[1] Jeremy Myers *A Short History of the Computer* Cited, March 25, 2000.

[2] Bergin, Thomas J. and Richard G. Gibson *History of Programming Languages-II* New York: ACM Press, 1996.

[3] McGraw-Hill *Programming Languages* New York: McGraw-Hill, 1997

[4] *A Brief History of Programming Languages* Cited, March 25, 2000. www.byte.com/art/9509/se7/artl9.htm

[5] *Java History* http://ils.unc.edu/blaze/java/javahist.html. Cited, March 29, 2000.

[6] Wexelblat, Richard L. *History of Programming Languages* New York: Academic Press, 1981.

[7] Jason Summerfield *Mobile Website vs Mobile App: Which is best for your Organization?* http://www.hswsolutions.com/services/mobile-web-development/mobile-website-vs-apps/

[8] Cyprien Lomas University of British Columbia, Diana G *Learning Spaces* http://net.educause.edu/ir/library/pdf/PUB7102.pdf

[9] Charlotte Danielson *Enhancing Student Achievement* http://www.ascd.org/publications/books/102109.aspx

[10] Arthur W. Chickering and Zelda F. Gamson *Seven Principles for Good Practice in Undergraduate Education* http://www.lonestar.edu/multimedia/SevenPrinciples.pdf

[11] Thomas H. Cormen , Charles E. Leiserson *Introduction to Algorithms-Second Edition*

[12] Dr. Axel Rauschmayer *Speaking JavaScript*

[13] *Bromium-Micro-vizualization* https://www.bromium.com/advanced-endpoint-security/our-technology.html

[14] Susan McKenney *Educational Design Research*

[15] JACKSON, D. and USHER, M. *Grading Student Programs* ASSYST, SIGCSE 1997.

[16] Mara SAELI, Jacob PERRENET, Wim M.G. JOCHEMS, Bert ZWANEVELD *Teaching Programming in Secondary School: A Pedagogical Content Knowledge Perspective* Informatics in Education, 2011, Vol. 10, No. 1, 73–88

[17] Alina Butko, Martin Molin *Intuitive Design Principles* Göteborg, Sweden, June 2012

[18] Michael O. Leavitt, Ben Sheniderman *Research-Based Web-Design & Usability Guidlines*

[19] Yury Puzis, Yvgen Borodin, Faisal Ahmed, I.V. Ramarkrishnan An Intuitive Accessible Web Automation User Interface

[20] https://www.cmu.edu/teaching/resources/PublicationsArchives/InternalReports/BestPractices-1stYears.pdf

[21] Carnegie Mellon *Best practices for Teaching First-Year Undegraduates* Eberly Center for Teaching Excellence, Carnegie Mellon University

[22] Stinson, D.R. *Cryptography: Theory and Practice* CRC Press, 1955

[23] Tarjan, R.E. *Data Structures and network algorithms*, vol. 44 of CBMS-NSF Reg. Conf.Ser.Appl.Math. SIAM, 1983

[24] J.D. Ullman, A.V. Aho, J.E. Hopcroft *The Design and Analysis of Computer Algorithms*, Addison-Wesley, 1974

[25] J. Van Leeuwen *Handbook of Theoretical Computer Science A: Algorithms and Complexity*, Elsevier, 1990

[26] J.D. Ullman, A.V. Aho, J.E. Hopcroft *Data Structures and Algorithms*, Addison-Wesley, 1986