

Creating a web application in Java Spring: “StudyBuddy”

Louis Jordan (6488798)

Karthikan Murugathas (6480737)

Nithesh Koneswaran (6474079)

A coursework submitted in partial fulfilment of the requirements of the module:

COM3014 (Advanced challenges in web technologies)

Dec 2019

Department of Computing

University of Surrey

Guildford GU2 7XH

Module leaders: Dr. Bogdan Vrusias

i. Requirements & Objectives (SMART)	3
Objectives for StudyBuddy:	3
General requirements:	3
Requirements for Mentor:	3
Requirements for Mentee:	3
Requirements for Admin:	3
ii. Description of the system architecture and UML diagrams	4
Use Case Diagram & Behavioural diagram of the three most important functions	4
Structural diagram	5
During the development process we ensured that our web application was separated and organized into distinct and coherent layers where each layer depends upon the next. It all starts off with the client making a request to the server to a predefined GET or POST mapping. The controller then communicates through a model to the service which in turn instructs the repository to make a call to the database. The views are subsequently rendered server side using JSP with the objects retrieved from the controller through the ModelAndView spring framework.	5
Behavioural view diagrams	6
iii. Description on the choice of technologies and patterns employed	7
iv. Supporting documentation	8
User Manual	8
Deployment manual	10
Group structure	11
Git repository	11
Individual Contribution	11
i. Louis Jordan (6488798)	11
ii. Karthikan Murugathas (6480737)	11
iii. Nithesh Koneswaran	12
Contribution weighting	12
References	13
APPENDIX-A (Dummy Accounts)	14
APPENDIX-B (Github activity)	15
APPENDIX-C(Planning)	16
APPENDIX-D(Coding standards compliance)	17
APPENDIX-E (Screenshot of mentee filter page)	18

i. Requirements & Objectives (SMART)

Objectives for StudyBuddy:

- A. To provide an open platform which will allow students to learn from other students who may be more knowledgeable in a given area, in exchange for monetary compensation.
- B. To allow three kinds of users (Admin, Mentee, Mentor) with different user privileges.
- C. Deploy a hosted MongoDB Atlas database to store user and booking details.
- D. Use Bootstrap and [SB Admin 2](#) (open source theme) to enhance the UI/UX.
- E. Use Spring security to provide authentication and authorisation.
- F. Adopt technologies such as JQuery, Ajax, and JSP to enhance the overall user experience, usability, stability and most importantly security.

General requirements:

- A. There must be 3 types of users: **Mentor**, **Mentee** and **Admin**.
- B. The payment of a booking must be trackable to ensure payments are made.
- C. A booking must hold a date, time and duration.
- D. Total booking price must be handled server side.

Requirements for **Mentor**:

- A. A mentor must be able to Login/Sign up, using their university email address.
- B. A mentor must fill in personal details in order to complete a profile.
- C. A mentor must be able to accept session bookings
- D. A mentor must be able to confirm payment has been received.
- E. A mentor must be able to see all upcoming/previous bookings.

Requirements for **Mentee**:

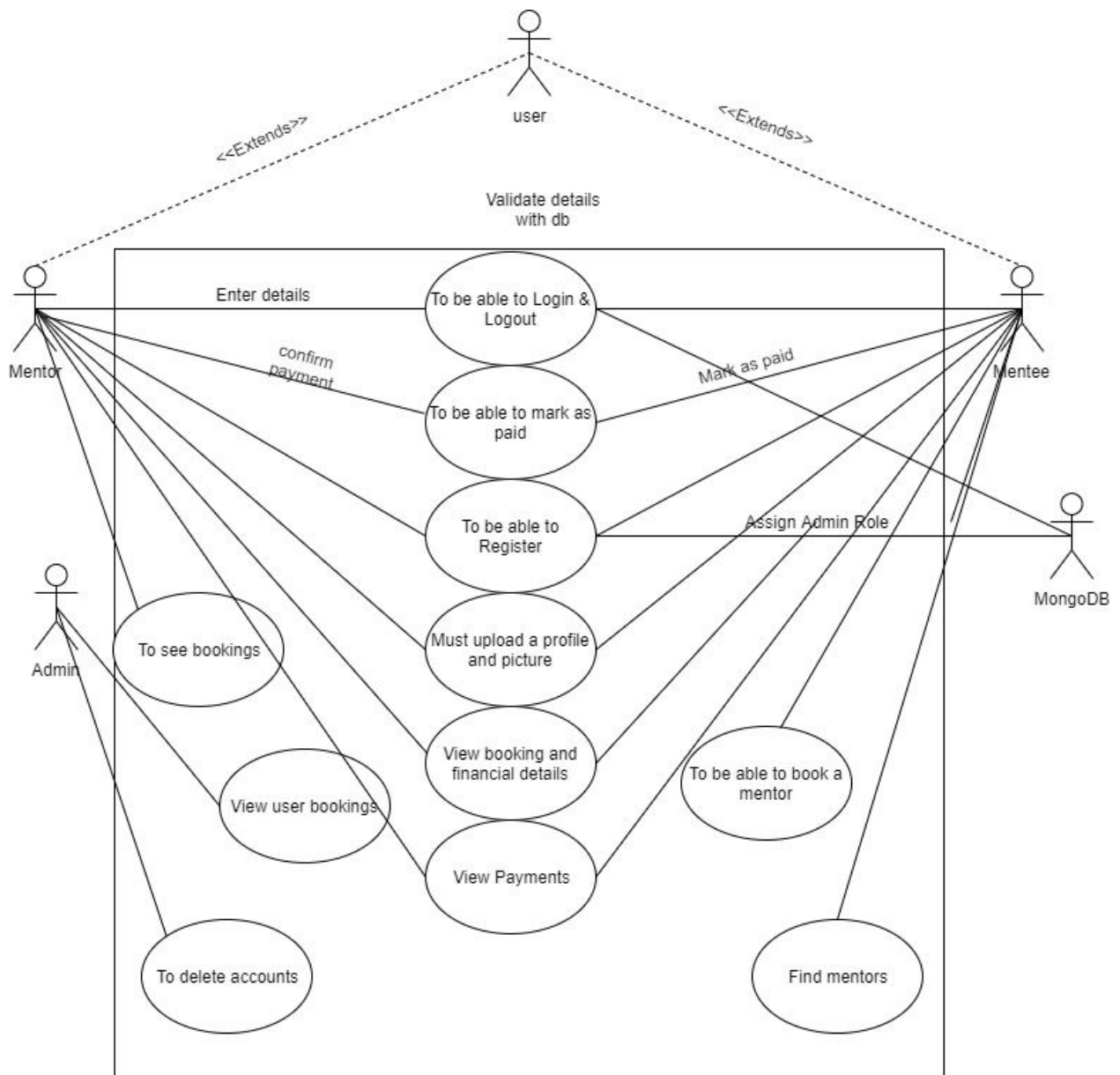
- A. A mentee must be able to Login/Sign up, using their university email address.
- B. A mentee must have the ability to fill in personal details.
- C. A mentee must be able to browse and filter through a catalogue of mentors.
- D. A mentee must be able to book a session with a mentor at a given time and date.
- E. A mentee must be able to confirm payment has been handed over (cash) and track when the payment has been confirmed by the mentor.
- F. A mentee must be able to see all upcoming/previous bookings.

Requirements for **Admin**:

- A. The admin must be able to Login/Sign up, using their university email address, into the secured overview environment.
- B. The admin overview environment should allow the admin to oversee important data and should act as a 'console' to the web app. Some of these functions include:
 - a. Deleting users manually.
 - b. Viewing important financial data.
 - c. View the relationship between Mentors and Mentees.
 - d. View bookings made for an individual user

ii. Description of the system architecture and UML diagrams

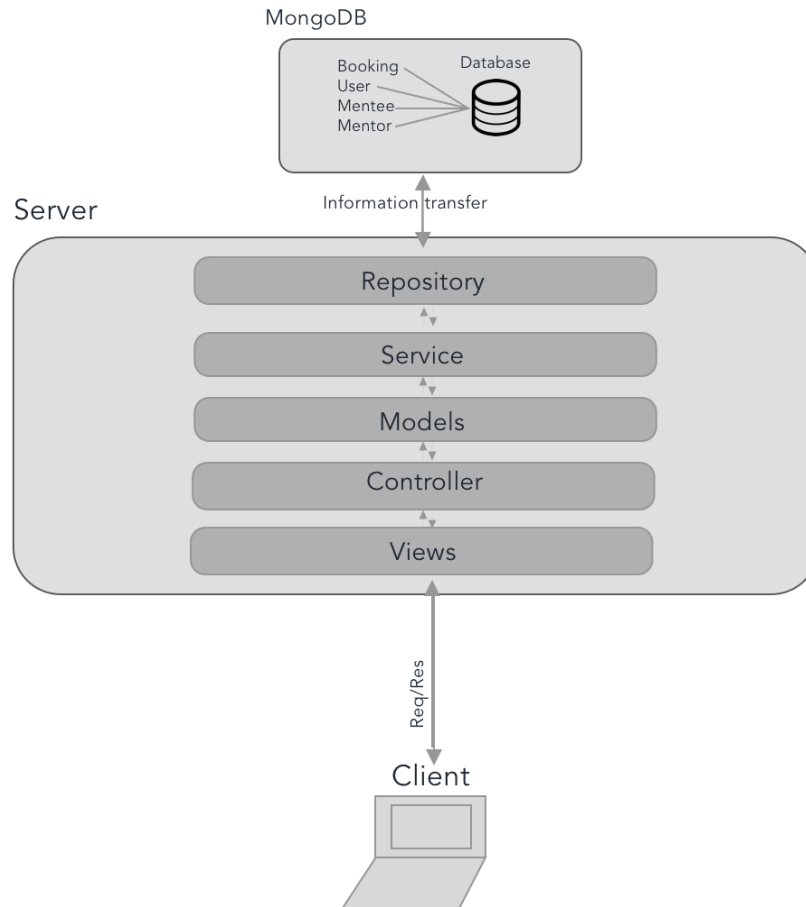
1. Use Case Diagram & Behavioural diagram of the three most important functions



The above use case diagram shows four actors. The Mentee and Mentor class inherit from the user as they have similar attributes. The database is also an actor, as it is external and accessible([online] Stack Overflow (2019)) in multiple systems. As you can see above, both mentee and mentor are linked when performing some tasks such as when the mentee wants to mark as 'paid' it would be required for mentor to confirm the payment. Both Mentee and

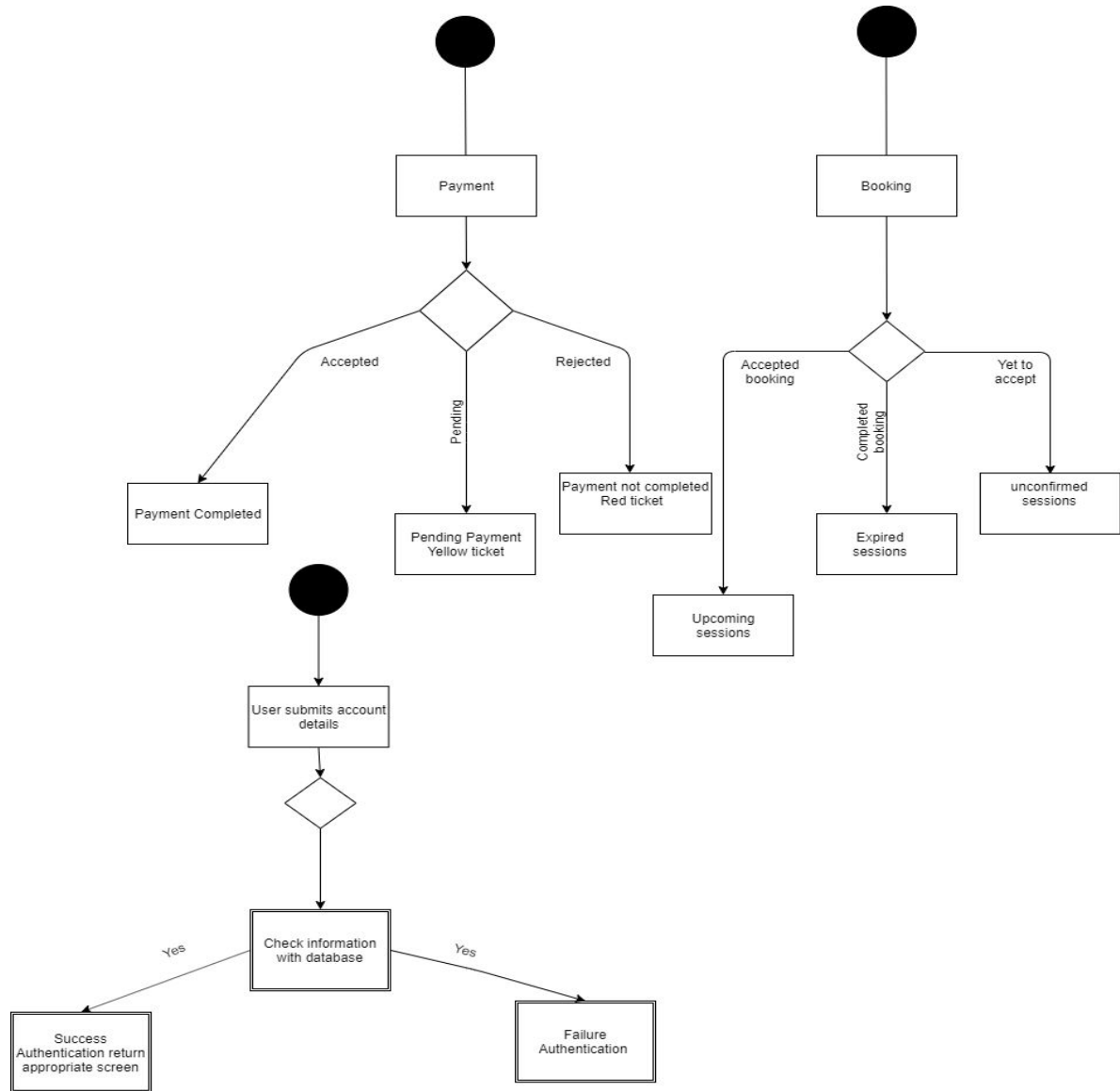
Mentor have different functions as they will be given different permissions. The admin user does not inherit from the user as they do not have similar attributes, and have completely different functionalities.

2. Structural diagram



During the development process we ensured that our web application was separated and organized into distinct and coherent layers where each layer depends upon the next. It all starts off with the client making a request to the server to a predefined GET or POST mapping. The controller then communicates through a model to the service which in turn instructs the repository to make a call to the database. The views are subsequently rendered server side using JSP with the objects retrieved from the controller through the ModelAndView spring framework.

3. Behavioural view diagrams



This behavioural view diagram, shows the three key functionalities of our web application. The three core functionalities of our web application is the booking, payment and login functionality. The booking functionality allows the mentee to see the current bookings, which is split into three categories: upcoming bookings, expired bookings and confirmed bookings.

The payment functionality has been implemented in a way which ensures full transparency and allows the administrator of the web app to track payments. A 2 step payment process was devised to A. Tell the mentor payment was made and B. Inform the mentee that payment was received. This process is trackable by both the mentor and mentee.

iii. Description on the choice of technologies and patterns employed

Technologies employed:

- Bootstrap (Front-end Framework)
A set of pre-defined CSS classes which simplify styling. This was used alongside a bootstrap admin panel theme: ([SB admin 2](#)).
- Java Spring (Back-end Framework)
Java framework which drastically simplified the web development process. Dependency injection and CoC (Convention over configuration) make it lightweight and easy to configure. Validation was implemented in the back-end.
- JSP (Server-side language)
Use to build html pages server side, of which the result is sent directly as a response. As opposed to making database calls directly from the client, by doing this server-side it obscures a lot of the database access mechanisms.
- HTML/CSS/JQUERY (Client-side languages)
These are the fundamental languages used to create structured and dynamic web pages. JQUERY is a library for javascript and simplifies the DOM manipulation process. See appendix-D for proof of standards.
- AJAX (Technique)
Asynchronous JavaScript and XML is a powerful tool that can be very impressive when used in the right way. It enabled us to make a call to our server and load contents without reloading the page. This was implemented in various form retrievals for StudyBuddy and for retrieving the list of mentors with a given filter.
- MVC (Pattern)
A fairly thin-client approach was adopted as we placed much of the Model, View and Controller logic on the server. The various layers were: database layer, service layer, Object Models, Controller and Views.
- MongoDB (NoSql Database)
MongoDB was found to be the most appropriate for our DB, given that it is schema-less and easily scalable. Data transfer objects were used to carry and update various data fields. Database Sharding was employed to split the dataset across multiple mongo instances. This optimizes
- MongoDB Atlas
An Atlas Cluster was deployed to store our collection of Bookings, Mentors and Mentees on a hosted server, rather than solely on our local machines.
- Spring Authorisation and authentication
Spring authentication was used to verify the user attempting to access the website. Authorisation was used to ensure certain get mappings were protected. For example the Mentee/Mentor would be stopped from accessing any admin routes. Alongside this, 3 different access levels (Mentor, Mentee and Admin) were defined with Spring Auth.
- Spring Boot
Spring boot enabled us to get up and running very quickly. It essentially automatically configured a lot of libraries we would have otherwise needed to add manually, with little setup required.

iv. Supporting documentation

1. User Manual

StudyBuddy is a web application created with the intend for students to mentor other students. A website which was built to be easy to use. This user manual explains how the user can get the most out of the web application.

The core functionalities of our web application include

- Ability to filter mentors available based on course ticked and level of study.
- Bookings page which shows 'Upcoming Sessions' and 'Pending sessions' and 'Expired Sessions':
- As a mentee user, the user can access a page called the 'Payments' page to show the Unconfirmed Payments, Processing Payment and Paid.
- As a mentee you may pick the day, time and duration of the session. It will automatically calculate the cost based on the duration of session.
- Mentor has a tab in the side-toolbar called "bookings", where they can see all their bookings

Minimum system requirements

- Java 7 and Spring Framework 4.3.24 release and above.
- 4GB RAM (Docs.spring.io, 2019).

Stage - Logged out

Study Buddy offers a simple front page with simple headings. The headings of StudyBuddy include:

- About → Introduction about StudyBuddy.
- Services → All the services provided by StudyBuddy.
- Portfolio → Displaying images linked to the mentors available at the courses.
- Contact → How to contact the owners of the website
- Sign up → A form will pop up, and once the required fields are filled in, it will create a Mentee/Mentor account
- Login → A form to pop up to allow the user to enter their registered information in.

Stage - Logged in

Both types of users (Mentee and Mentor) will be introduced to a responsive and interactive dashboard to navigate through to the pages. Both the Mentee and Mentor will be asked to build a profile to perform any other tasks. This is to get more information from the user and help to link the two users together.

The features available once logged in as a Mentee user:

- Before building profile:
 - Dashboard → The dashboard will prompt the user to build their profile
 - Build profile: The user will be asked to fill in all their personal information, so they can access all the other features.

- After building profile:
 - Dashboard → The dashboard will list the number of different mentors, unsettled bills, upcoming sessions, and number of pending payments to be confirmed.
 - Profile → The mentee will be able to see their profile and make changes if necessary.
 - Find Mentors → This page will allow the Mentee to find Mentors based on picked categories. Initially it will show all the mentors available in card form; each mentor will have a short introduction and have their 'level of study' and 'course' label attached to them. You may also click the cards to view the selected mentor's profile.
 - Booking a mentor → On the left hand side side bar, the user can book to arrange a booking by picking the date, time and duration (which will list the total cost based on the duration).
 - Bookings → will show your booking tickets and show whether it is confirmed by the mentor. Any expired tickets will also be displayed.
 - Payments → This page will show all the booking events associated with the mentee. It consists of bookings which has been paid for, awaiting confirmation of payment or if it has been paid.
 - To logout, the user needs to click their profile picture and drop down list should appear with one of the items as the logout button. Once pressed they will be sent back to the home page.

The features available once logged in as a Mentor user:

- Before building profile:
 - Dashboard → The dashboard will prompt the user to build their profile
 - Build profile: The user will be asked to fill in all their personal information, so they can access all the other features.
- After building profile:
 - Dashboard → The dashboard will prompt the user to build profile
 - Bookings → will show your booking tickets for you to confirm, this will show when the mentee user has booked you.
 - Payments → This page will show whether the user has paid, or payment is processing or has not been paid. If the mentee has paid you can mark as paid by pressing the paid button
- The user will be asked to fill in all their personal information, so they can access all the other features.

The features available once logged in as an Admin user:

- Dashboard - In the left toolbar, it will display Mentees, Mentors and Users. Mentees will display all Mentee users and will also allow the admin user to display bookings and delete the accounts. Likewise, with the Mentor users, filtering users by role 'Mentor' and allowing the user to view the bookings and delete the accounts, if the admin wishes to. Users, on the other hand will display both role types, and will show the bookings for both role types and the permission of deleting accounts.

- The user will be asked to fill in all their personal information, so they can access all the other features.

2. Deployment manual

StudyBuddy is very easy to set up and get running, however there are some prerequisites required to start up the web application.

Prerequisites:

IDE

Download an IDE which supports Java EE. Few IDEs that support Java EE are Eclipse, Netbeans and IntelliJ. If the client wants to use NetBeans, one thing to ensure is that all the necessary plugins are installed and enabled. You can check that under the Tools/Plugins menu. The most important plugin is the “Java Web and EE” plugin. I recommend using eclipse, as StudyBuddy was built using eclipse.

Downloading Eclipse

→ Go to <http://www.eclipse.org/downloads/packages/> and look for Java EE Developers.

Tomcat

StudyBuddy uses ‘Tomcat’ which is an open-source web server. Tomcat scans the webapp, (a folder within our deployed resources) and decides which JSP need to be deployed and available when run.[Kattan, S(2019)]

→ To install Tomcat, you will need to download a stable version of tomcat (version 7 or newer). If the user has Java 7, then they would need to tomcat 8 minimum for it work.

Potential errors and solutions of Tomcat

Error: Tomcat server not found

Solution: To run ‘StudyBuddy’ the user needs to make sure Tomcat is checked in ‘Runtimes’. To do this, right click the project(StudyBuddy) and look for Project Facets and click ‘Runtimes’, if no Apache Tomcat v.xx is shown, then click ‘new’ and pick tomcat v.8.0+.

Error: Port conflicts

Solution: Tomcat is already running. Close/stop the server and click run again. Otherwise, make sure something else is not running on port :8080.

Running ‘StudyBuddy’

To run our web application, right click the project folder and click ‘maven build’. Right click the project folder one more time and run as, ‘Run on Server’, it will take some time to build, and then it will load the site. Note, the site is loaded as localhost:8080. Dummy accounts can be found in Appendix-A.

v. Group/Individual Performance

1. Group structure

Being in a group of only 3 members proved to work surprisingly well. We were able to divide the work-load fairly equally, at each stage evaluating what had been accomplished and subsequently assigning new tasks.

In the beginning, one team member defined the initial front end structure and then we moved on to another member implementing Spring security. Once the User, Mentee/Mentor and Booking objects were defined each team member was assigned to work on the view of either the Mentor, Mentee or Admin. Towards the end the whole team worked together fixing bugs and adding innovative new features and updating the entire theme. Trello and whatsapp were used consistently to plan and coordinate work (See Appendix-C).

2. Git repository

With over 200+ github pushes (See Appendix-B), it is safe to say that there was consistent use of version control conducted by all team members throughout the duration of the project. Each major task had its own branch and on completion of the task, it got merged with the master branch.

3. Individual Contribution

i. Louis Jordan (6488798)

“Having had experience developing web applications with other frameworks, it was most certainly interesting seeing what Java Spring had to offer. Defining the overall structure, defining get/post mappings, formatting and parsing data between the front and back end, screen layout design, handling the booking process, form design and DOM manipulation with the mentee/mentor/booking objects were all tasks that I completed. My primary focus being the Mentee and Admin panel.

A number of hurdles I encountered were most notably throughout the design of the booking process: having to parse the Date object between type string/Date. Design-wise, the grid layout was rather confusing to begin with but became clearer as time went on. Parsing data between the front and back-end was confusing at first but when done once was easily replicated. “

ii. Karthikan Murugathas (6480737)

“The only experience in web development was during one of my second year module, ‘web development’ where we were working with Ruby to create a simple website. It was amazing to know that you can use Java as a back-end to perform tasks. I came up with the idea as it would benefit struggling university students with the exchange of cash. I attempted validating user input and helped pick the design with bootstrap.

The number of problems I came across was getting used to the MVC structure and getting used to the new technologies such as JSP. However, the labs helped a lot and the various amounts of research, made this is an intense and an amazing learning experience.”

iii. Nithesh Koneswaran (6474079)

I was in charge of managing the back-end of our project as well as organising weekly meetings to make sure that everyone was up-to-date on the project. In the initial meetings I created a trello board dedicated in tracking progress and managing sprint tasks. As a group we iteratively designed the requirements and individually worked over each other's work improving code design as well as peer-assessing and testing code functionality. Other contributions include setting up the repository & services for each entity, redesigning the whole theme of the project, setting up the security config and web config files, fixing any bugs/errors to do with the backend. My biggest achievement was the design of the mentor filter page and the use of ajax to filter the list of mentees based on the options the user selects (see Appendix-E). Overall the back-end role was simple since the repository provided pre-existing simple methods to access the database without any hassle. Some stuff that I struggled on was trying to call back specific error messages from validation and serialising an image and storing it into mongoDB. These features were pushed into a back-log to tackle later.

The biggest hurdle was separating the system's logic to meet the requirement of having 3 access levels. As the back-end developer I researched the different databases compatible with spring and concluded with using the simple and scalable, MongoDB. Installing the local version of MongoDB was quite the hassle, after switching to MongoDB Atlas our experience has been a very smooth process. After that I spent a full week configuring spring authentication to separate the access levels. The most difficult task that we faced as a group was setting up the project and configuring the application so that the resources were linked and spring was able to run without issues. Switching from intellij to eclipse solved most of our issues. But we still had a lot of dependency problems during the initial construction of the project, I solved this by converting our project to use spring boot. This made it far easier to get the dependencies we need for our requirements and solved any dependency errors (like missing dependency jar error).

Contribution weighting

As a group we can confirm that the workload was distributed, accomplished and delivered equally.

References

1. Docs.spring.io. (2019). *9. System Requirements*. [online] Available at: <https://docs.spring.io/spring-boot/docs/1.5.21.RELEASE/reference/html/getting-started-system-requirements.html> [Accessed 17 Dec. 2019].
2. Kattan, S. (2019). [online] Available at: <https://www.quora.com/How-does-Apache-Tomcat-work-internally> [Accessed 17 Dec. 2019].
3. Stack Overflow (2019) [online]. Available at: <https://stackoverflow.com/questions/49172461/can-database-be-count-as-actor-in-use-case-diagram> [Accessed 17 Dec. 2019].

APPENDIX-A (Dummy Accounts)

Account Type	Email	Password
ADMIN	admin@surrey.ac.uk	password
MENTEE	mentee@surrey.ac.uk	password
MENTOR	mentor@surrey.ac.uk	password

APPENDIX-B (Github activity)

louisjoejordan / **Web-Application-Project-Front-End** Private

Unwatch ▾

1

★ Star

0

Fork

0

<> Code

🔔 Issues 0

🔗 Pull requests 0

▶ Actions

📁 Projects 0

🛡 Security

📊 Insights

⚙ Settings

Web application group project.

Edit

[Manage topics](#)

📄 220 commits

🌿 13 branches

📦 0 packages

📦 0 releases

Branch: master ▾

New pull request

Create new file

Upload files

Find file

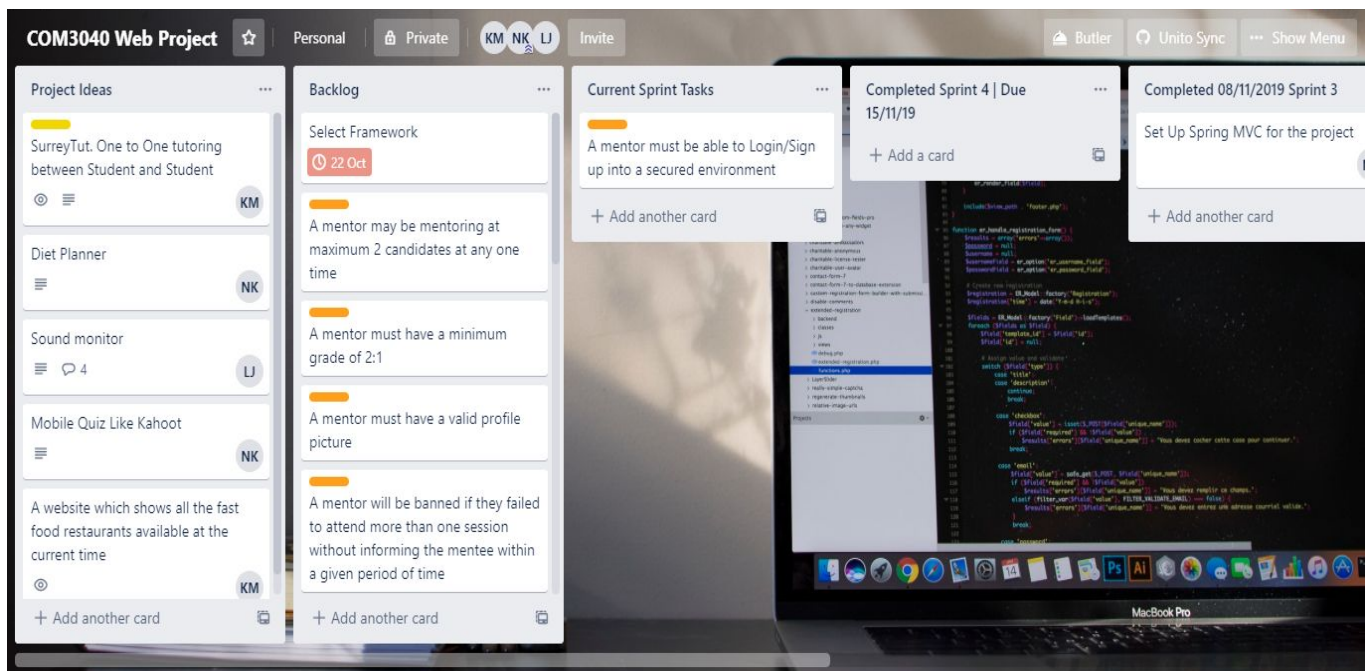
Clone or download ▾

nk00374 updated payment check

Latest commit fea2476 22 seconds ago

📁 .settings	Added views, controller, repository, services that allow for the user...	22 days ago
📁 WebContent	Reorganised jsp views	9 days ago
📁 src	updated payment check	22 seconds ago
📄 .DS_Store	Added Mentee View page	7 days ago
📄 .classpath	Added views, controller, repository, services that allow for the user...	22 days ago
📄 .gitignore	Updated Project Structure, fixed link between resources and jsp files	last month
📄 .project	Added views, controller, repository, services that allow for the user...	22 days ago
📄 pom.xml	Added styling and scripts for the new home page	8 days ago

APPENDIX-C(Planning)



“Trello” Used to organise sprints, TODO tasks and assigning tasks to members.

APPENDIX-D(Coding standards compliance)

Nu Html Checker

This tool is an ongoing experiment in better HTML checking, and its behavior remains subject to change.

Showing results for contents of text-input area

Checker Input

Show

☒ source

☐ outline

☐ image report

Options...

Check by

text input

css

```
<!DOCTYPE html>
<html>
  <head>
    <link href="../../css/error.css" rel="stylesheet">
    <link href="../../css/creative.css" rel="stylesheet">
    <title>error 401</title>
  </head>
  <body>
```


Check

Use the Message Filtering button below to display options for hiding/showing particular messages, and to see total counts of errors and warnings.

Message Filtering

<http://validator.w3.org> was used to ensure our HTML files were meeting coding standards. On a few of our pages warnings and errors were displayed, however given the fact that our html was mainly derived from bootstrap and SB-2 admin we were confident HTML coding standards had been met.

APPENDIX-E (Screenshot of mentee filter page)

 STUDYBUDDY


[Dashboard](#)


[Find Mentors](#)

[Bookings](#)

[Payments](#)

[←](#)



James May 

Find Mentors

Subject and Level of Education Filter

☐ Level 5 : Foundations Degree

☐ Level 6 : Bachelor's Degree

☐ Level 7 : Master's Degree

☐ Level 8 : PhD

Level of education subject to change in the future.

☐ Maths

☐ English

☐ Chemistry

☐ Physics

☐ Computer Science

☐ Law

☐ Biology

☐ Economics

☐ Accounting and Finance

☐ Art

☐ Design

☐ Aerospace Engineering

☐ Mechanical Engineering

☐ Electrical and electronic engineering

☐ Civil Engineering

☐ Psychology

☐ Criminology

☐ Business

☐ English literature

☐ Medicine

☐ Geography

☐ Chemical Engineering

☐ History

More subjects to be added soon, for now pick the subject area that works best for you.

1 Available Mentors

Searched for 'Anything ' with level categories: [ALL] and with subject categories: [ALL]

