**Acknowledgements:**

We would like to acknowledge our supervisor Martin Hynes for all the support and GMIT staff for their help throughout the last four years.

**Abstract:**

In this project, we will be building a web application for students which will act as a student portal. We used SpringBoot, ReactJS, MongoDB and Google Firebase as the main frameworks for our web application, user authentication and data storage. We used AWS and DockerHub as our cloud platforms to deploy the Web Application, GitHub for controlling the source code and saving our work. The project will be written mainly in java and the react framework and a few other languages such as Javascript and CSS.

Currently with the times we are in, studying and keeping up with our work is very challenging. We cannot get access to course materials due to several reasons, such as little to no face to face contact with lecturers, and college work is being done all online. A lot of students miss the online lectures due to bad internet and because of this they may fall behind on their work. Other students do not even have an opportunity to visit their institute/university and are not able see any of their peers in person throughout the year. This is what sparked us and made us think that by making a student portal we can enhance and help these students with their college work and allow them to participate with each other.

This project is a research project as it will allow us to get a better understanding of the work that goes into making a high-end web application. It would also allow us to get a better comprehension of the different languages and frameworks that we will be using.

The overall objective of this project is to assist students with their online learning and to make online learning more enjoyable. It will also allow students to reach out to other students through the messaging part of the web application. First year students who have not been to college do not know who is in their class or what they look like. The web app will act as a social media platform which will allow the students to talk to each and get to know each other.

This project supports further expansion and insertion of new code and components such as a user profile page and a timetable to manage classes and modules. The proposed solution will be comprised of a high-end web application that allows the user to log in, talk to their fellow students and save notes in a notepad or as sticky notes. This will be done through CRUD functionalities of the app.

Chapter one:

Introduction:

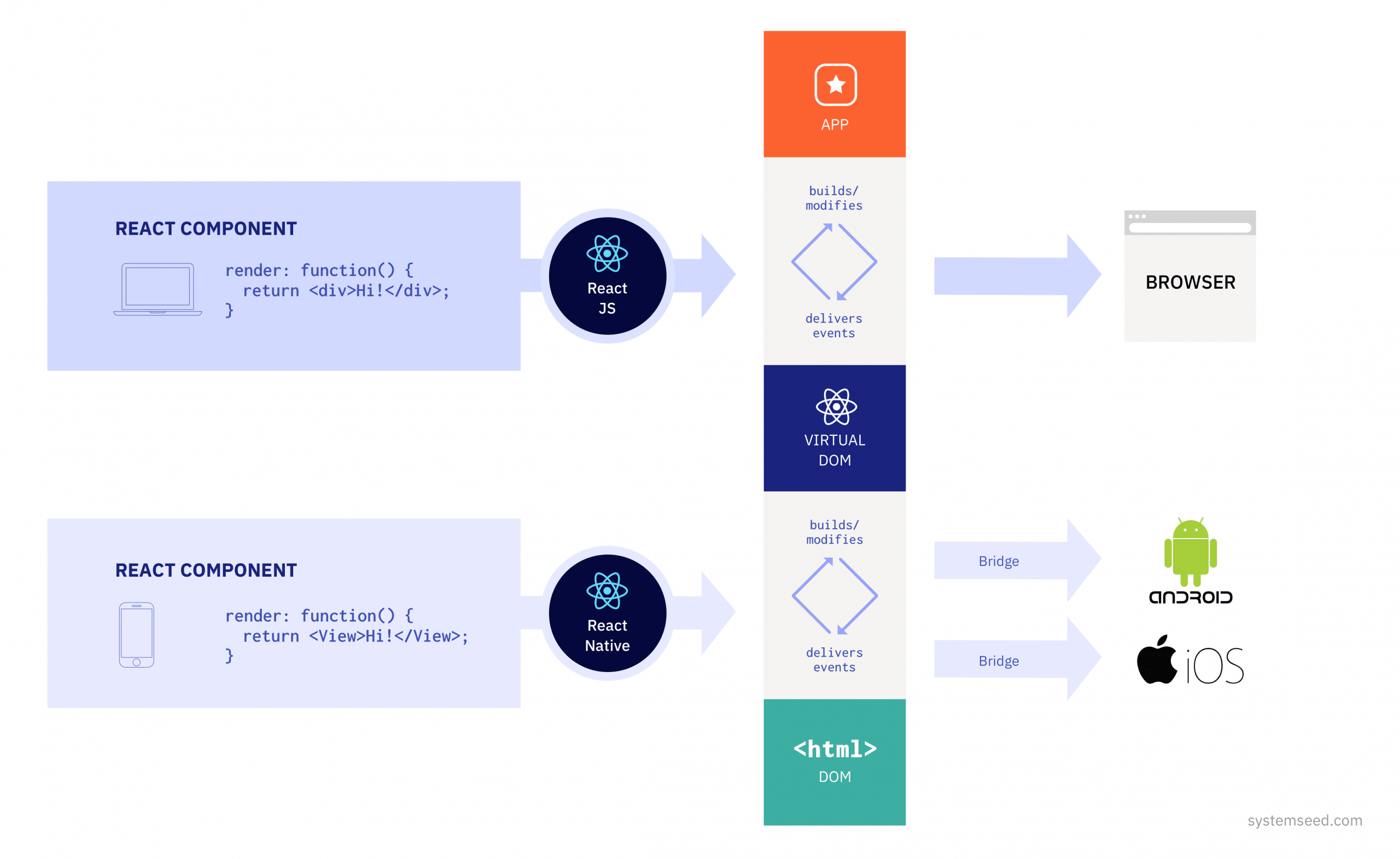
Initially our first idea for our final year project was to create a web application but we did not know what direction to go in. We thought of different ideas such as a business website for buying and selling products or a blog website. While discussing our ideas together, we realized that due to the global pandemic happening around us, our college life will be affected. We will not be able to see each other and work on this project due to lockdown. We thought that if we are going to have this problem, then there will be other students who will be faced with the same issues. We decided that we would build a web app that will help us with this problem. This web app will allow students to get in contact with other students and share their notes and resources in a global notepad or talk to each other privately and share documents. The intention of this project is solely research based with the purpose of helping students.

Developing our Design:

At the beginning of this project it was challenging to select the language and the framework we wanted to work with. There were a lot of options that we could have gone with such as Python and Django or IonicJS. We decided not to use these languages and frameworks because we had previously used these, and we wanted to challenge ourselves with something new. While researching about different companies in our field, we discovered the most widely used frameworks that a lot of the companies use are SpringBoot and ReactJS. We thought this would be a great opportunity to learn new frameworks and enhance our skills further in Java.

This project uses the ReactJS framework to build the front end of the web app, and it uses the SpringBoot framework for the back end of the application. The ReactJS framework uses Javascript as the main language while the SpringBoot framework uses Java as the main language. We wanted to use MongoDB and Google firebase as the databases for the project because they are the most popular databases in the industry. We also wanted to run this project on the cloud so that it could be accessible from anywhere. We had numerous options when it came to cloud services, such as Google Cloud, Amazon Web Services and Microsoft Azure. We decided to use Amazon Web Services (AWS) and DockerHub for this part as they are industry standards.

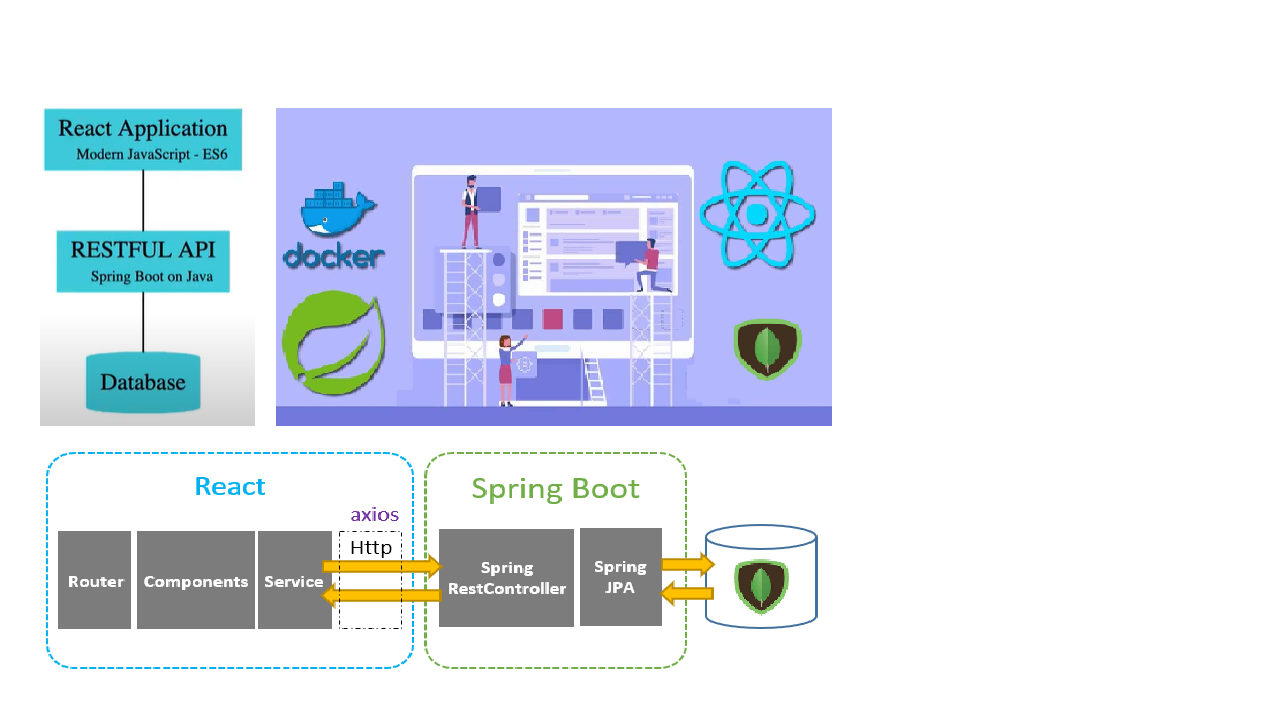
The figures below indicate how both of these frameworks work. We will address these frameworks in depth further in the dissertation.





Final Design:

Student Hub, the title of our project, is a web application project that involves SpringBoot, ReactJS, MongoDB, FireBase and AWS cloud technology. The end product of this project will be a web application that will be running on aws and can be accessed anywhere and on any device. The user will have access to several services to help them get through each semester of college and have a pleasant experience throughout. There will be a facility that will authenticate the user and give them access to these services. The users will be saved in MongoDB, running on the cloud. The user will also be able to contact us regarding any queries that they may have through our live contact page. Below is a schematic diagram that we came up with at the start of the project to give us an idea of the end product.



Objectives:

In order to provide a high-end web application that is also high performing, multiple goals need to be accomplished during the course of this project.

* The first objective was to create a SpringBoot and ReactJS application that could talk to each other and save details to the Mongo database. We also wanted to have a basic web application set up i.e. home page, navbar and footer.
* With the initial web app configured, we want to work on the common pages that usually appear on a website such as a contact page and an about us page. In order to achieve this, we will be running a node server that will detect when the user wants to contact us and will send us an email.
* Subsequently, we will create pages for the user authentication. Once we have designed the register and login page, we will connect them to the back end where the data will be saved in our database. The password will be hashed using the hash algorithm for extra security. We will implement a recaptcha on the login page to provide further security against bots and androids. The mongo database will be setup on the cloud so that it can be accessed from anywhere.
* After the user has successfully logged in they will be able to access a number of features such as sticky notes, notepad, messenger and a student forum page. All these features will be connected to the firebase database to allow for quick access. The messenger chat feature will be using an external API known as chat engine. We will edit this API and add our changes to the chat page.
* Lastly, the next step would be to deploy our project to the cloud. This will be completed through the use of Docker files and Docker-compose. The docker files will then be run on an Instance that we create in AWS.
* If we have extra time, we would also like to include a user profile page where the user can customise their profile and it can be viewed by other students.

Overview:

As this is our final year project, we knew that we would have to challenge ourselves to work on a project that will allow for independent learning. We wanted to make sure that we learned something new through each stage of development. This dissertation will be laid into different chapters, these chapters will explain our thought process and the different aspects of the project. This section contains a small description of each of the chapters that we will be covering.

Methodology:

This chapter of the dissertation will be focusing on the steps that we took throughout the development stage to make sure we have a successful project. We will discuss the reasons for various technologies we used such as, SpringBoot, ReactJS and mention the problems that we encountered during the initial set up and the development stages of the project. We mainly used the agile methodology, and we will further provide diagrams where needed to show off the methodology used in our final project design.

Technology Review

This chapter will be looking at all the different technologies that we have encountered during our research. We will go into further details on how to set up various technologies, for example Docker, Robo3T, Postman, and their key functionalities and advantages.

System Design

This chapter will focus on the different components and how they were implemented. Snippets of our code and images will be provided when explaining components to give a better understanding of the project. Each component will be discussed in great detail and we will explain how each part of the project contributes to the final product.

System Evaluation

This chapter of our project will discuss how our project performs, how scalable it is and its robustness. We will also mention any limitations that we faced such as, latency issues and cloud issues.

Conclusion

This chapter will be going over our goals and challenges that we set ourselves with. It will also cover if we accomplished these challenges and goals. The end result of the project will be discussed alongside all the difficulties that were experienced. These difficulties then will be discussed to show how we overcame our problems.

Project Links:

Link to Repository

* <https://github.com/sagheerahmadGmit/FinalYearProject>

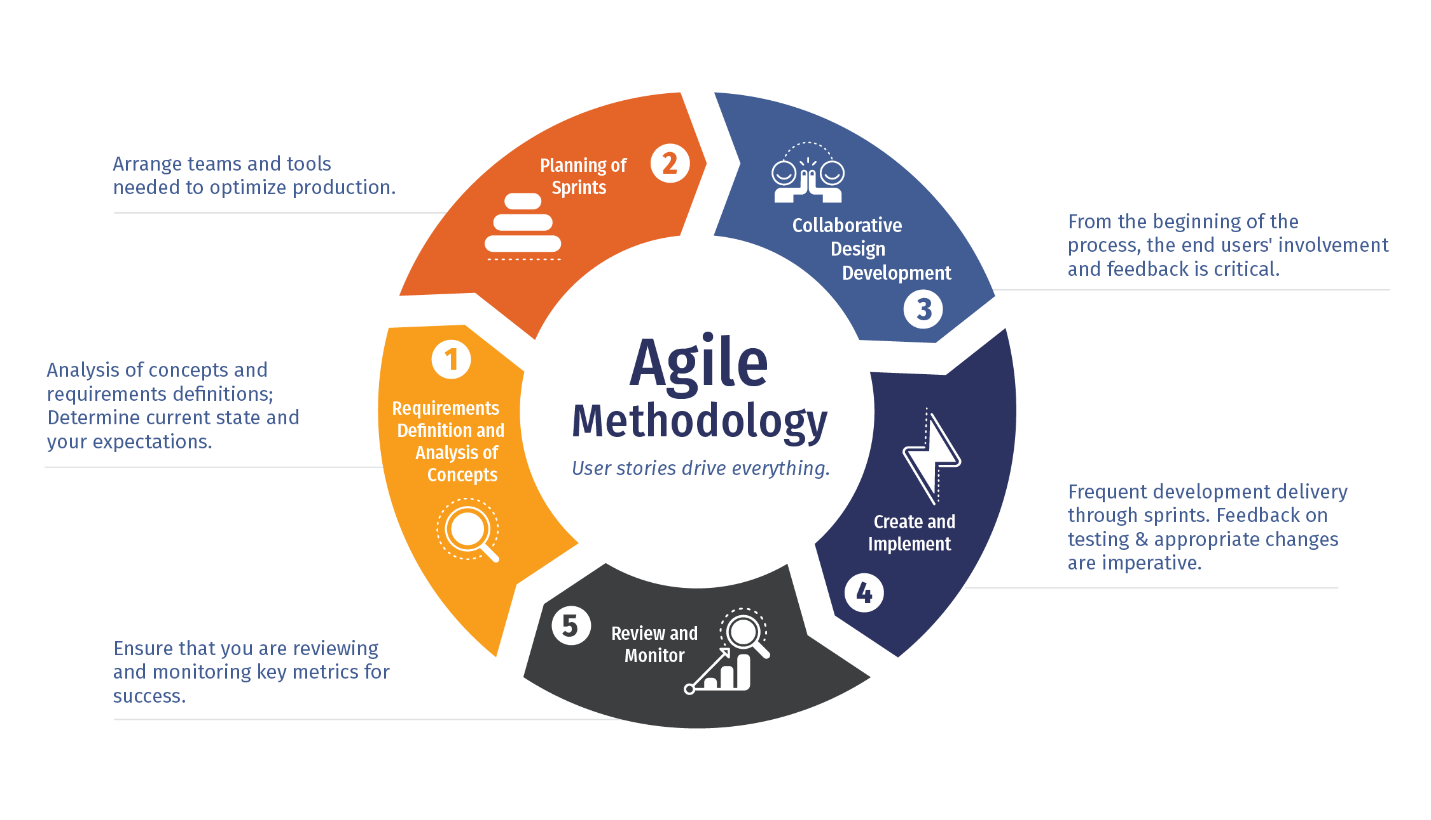
Chapter two:

Methodology

As previously said, this section would focus on the various methodologies used to deter possible disasters. It was important to schedule and monitor the production stage ahead of time, so that any anticipated issues or glitches could be addressed. This enables us to make smarter plans, resulting in a faster planning phase and less lost time. We can also write simpler, more readable code by planning ahead. If this project is not properly prepared, it will result in events such as missed deadlines or the need to cut back on the project, resulting in the goals not being achieved.

Planning Phase

We defined the project's scope during the initial planning process. We concluded that Agile would be the best methodology to use. We investigated other methodologies, such as waterfall, but after comparing them to agile, we opted to stick with agile. We were able to divide the project into multiple stages, thanks to the Agile approach. It allowed us to make incremental improvements to the project at each point. Agile approach allows for greater flexibility in project planning and execution. We could also modify the project's various elements as it progressed. After discussing the different methodologies, we had to decide on the technologies to use. We researched into several technologies and there were a lot of good options to go with, but we concluded that Java, SpringBoot, ReactJS and MongoDB would be the main technology for developing this web application. In addition to the technologies, we had to choose an integrated development environment (IDE). We knew we had to use IntelliJ for the backend, we had previously used it in one of our modules and wanted to expand our skills and knowledge on it. We decided to use Visual Studio Code for the frontend. This way it would be easier to navigate through the code and see where the backend and frontend connect to each other.



We developed a few diagrams during this process to help us better visualize the work that will be needed. The diagrams looked a lot like the schematic diagram described earlier. Since we couldn't meet in person to collaborate on the idea, we had to think about how we would approach everything. When doing research for this initiative, we came across a programming technique called mob programming. After doing more research, we decided to tackle the project with a mob programming approach. Mob programming is a method of software development in which a group or team of people work on the same project element at the same time. This way, we could both work on the project at the same time and know what was going on with the code. We made the decision to call a few days a week to progress on the application.

Requirements Analysis

One of the most important facets of this project was defining the project's criteria, as it would help us to understand what we need to do and split each phase down into steps. These phases will then assist us in completing various activities based on priority and importance. We had to explore and research what is needed for a viable web app, as well as the components that are required. We needed our web app to be able to meet these standards as efficiently as possible. These requirements are as follows:

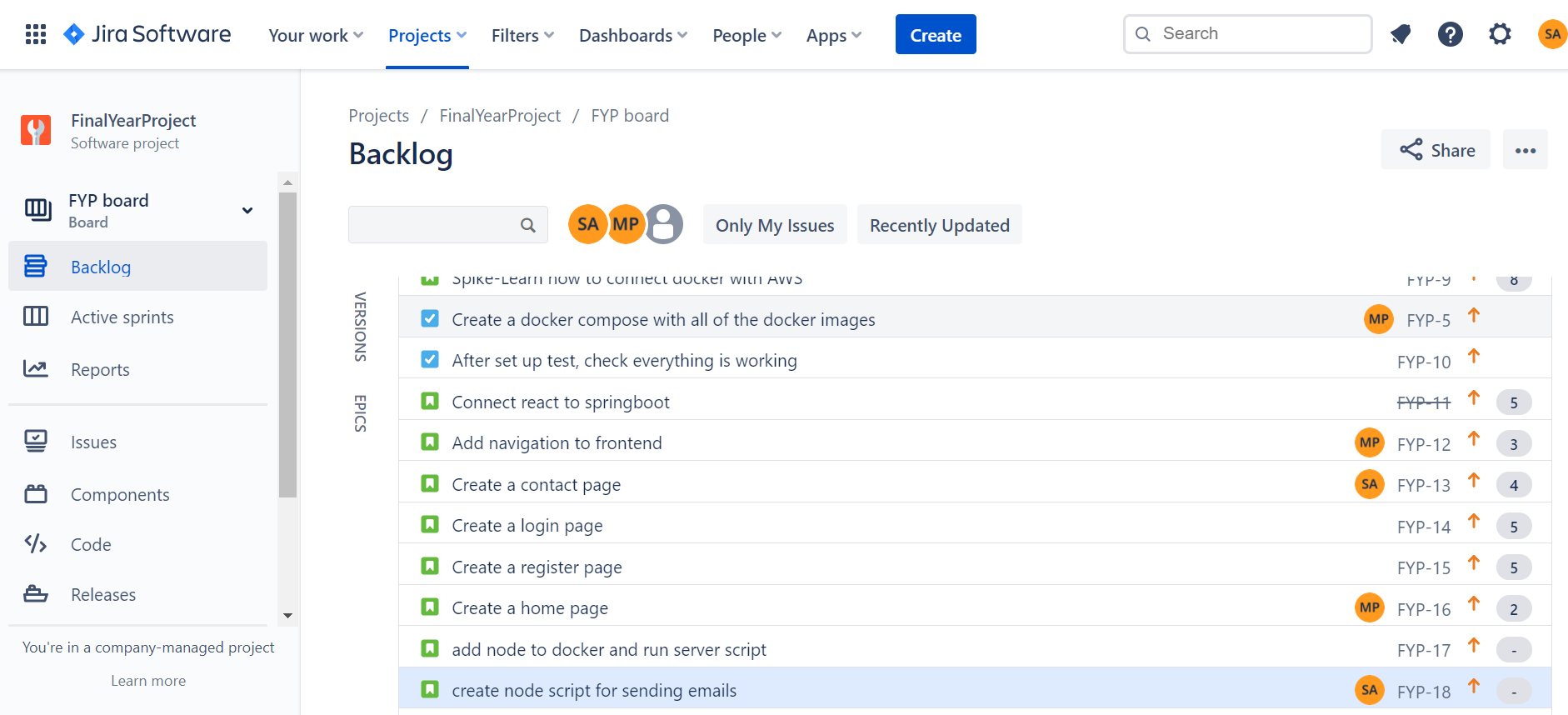
* To begin, we must have a functioning web application that operates correctly and enables the user to perform the function for which it was designed.
* Second, provide a completely functioning and stable register and login framework that saves user data in a database. To add to the encryption, the password must be hashed.
* Next, the user should then be able to log in and communicate with other students, as well as save notes and documents to their account. They should also have access to the student forum, where they can post questions and get answers.
* After that, we would like to have the project running on the cloud through docker hub and AWS.
* Finally, if we have enough time, we'd like to convert our web app into a mobile app so that users can access all of their information on the go.

Meetings

Since we were a two-person team, we had to organize and arrange our meetings to collaborate on the project on specific days and times. We had to schedule it around other classes and assignments. We agreed that we will work on the project every other day and make a few commits. Since we had other assignments to complete during the year, this was a little more complicated at times. We had talks with our supervisor as well. Every week, either on Monday or Thursday, we met with our supervisor. This was to provide a report on our development, get input on any prior work, and address potential future proposals. The minutes of the meeting will be emailed to our supervisor, and we hoped to complete the recommended conditions for the next meeting. We used Microsoft Teams as the main tool for communication and meetings.

Development

We divided our work into smaller tasks and kept track of all these tasks on Atlassian Jira. We registered any bugs or concerns on Jira so that if we have these problems in the future, we can report back to this documentation. To ensure that we were on track with the project, we reviewed each feature separately and made regular GitHub commits.

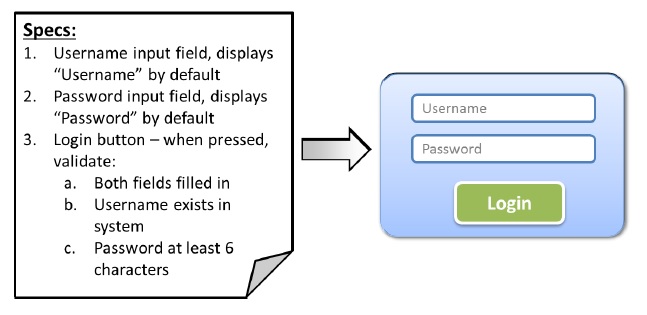


This is an example of how we used Atlassian Jira. We gave each task a story point and that dictated the amount of time that we could devote to that specific task.

Testing and Validation

Testing was a critical component of the project, as well as software in general. Testing can save us a lot of time by preventing any future catastrophes in the code and project. It also reassures us that the software can fit in with other features and elements of the web application. To test the project, we used Robo3T and Postman, among other tools. We carried out black box testing on the project by allowing other students to register and access the website in order to assess the various functionality and components. Black box testing is a form of software testing that focuses solely on the website's front end. It has no understanding of the website's back end or the code that makes it work. This was extremely beneficial because we received direct feedback from students. We asked them if we could strengthen and change things, and we tried to make the necessary improvements.

Allowing a client to log in with incorrect information and watching what happens or attempting to enter a website function without being logged in, are examples of black box testing.



Problems Encountered

We ran into a lot of issues and failures during the early stages of development, including docker not running, the user being unable to save their credentials while registering, and GitHub failing to push our code to the correct branch. We had to look at these mistakes and come up with suitable alternatives. We made a note of the solution after we had resolved these problems. This was so that if the same issue arose again, we would know how to correct the mistakes and issues.

Version Control Manager

We wanted to select a version control manager before we began working on the actual code component of the project. There were a couple of great options, including Bitbucket, an Atlassian version control repository hosting site, and AWS CodeCommit. We concluded that using GitHub, which is more well-known and widely used in the tech industry than any other version control manager, would be more useful to us. We had been using GitHub for our work and projects since our first year of college, so it made the decision easier. GitHub has a lot of features that we could take advantage of, the most important of which is the ability to work on various branches to finish different parts of the project. It also allowed us to keep track of our issues inside the project repository. If we make an error or accidentally insert the incorrect file, GitHub helps us to revert our commits and return to the original code. We knew we wouldn't lose any of our code this way. Since GitHub helps us to see where in the code, we make modifications and how they affect the code, we were able to quickly see where we had glitches and errors. We decided to commit to GitHub on a regular basis to encourage us to work on the project more often. We made small commits rather than huge commits or one final commit at the end. This is because it demonstrates our progress and thought process in the development stage. It further demonstrates the many issues we encountered and how we addressed them. In our view, GitHub was by far the best version control manager.

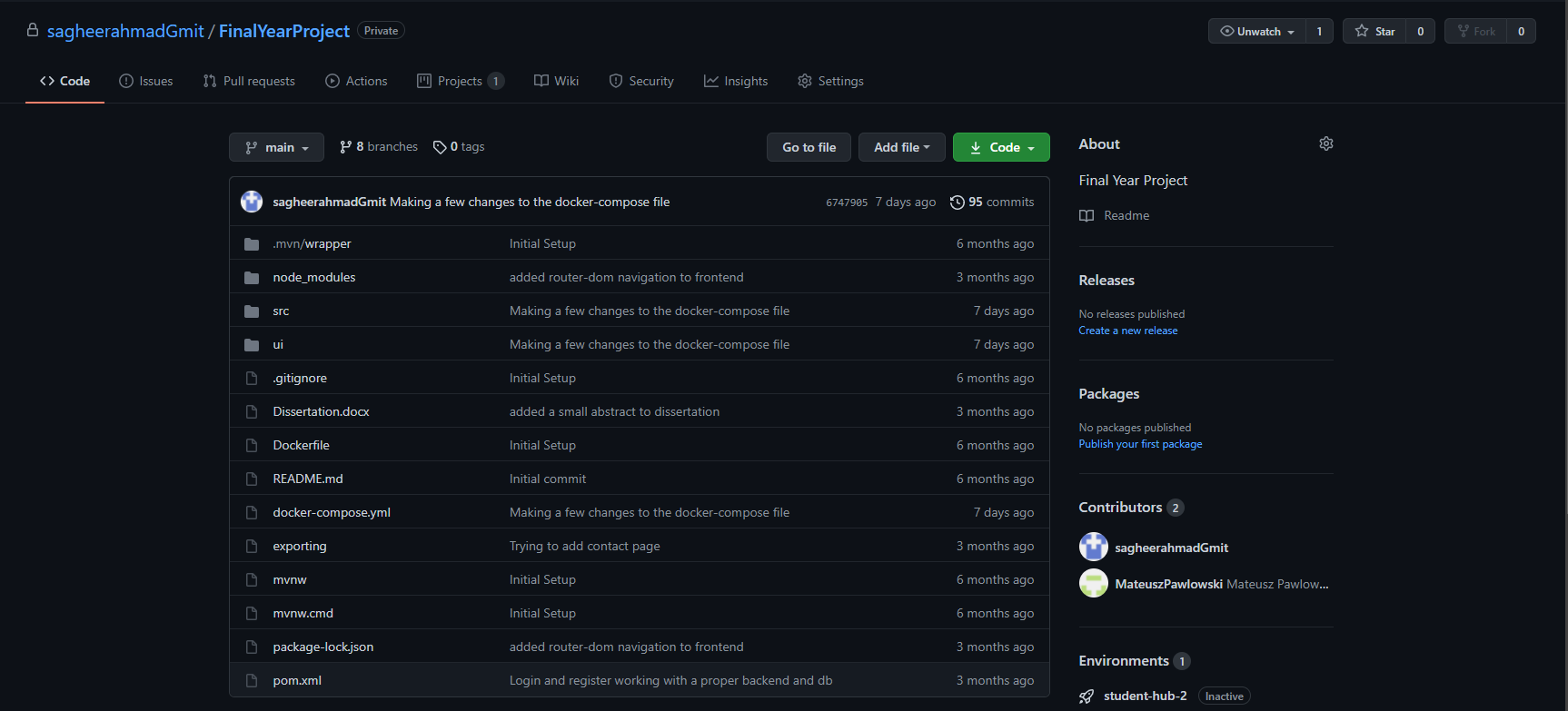


Figure: GitHub

Chapter Three

Technology Review

This segment discusses the various technology that we used to construct our web application. We will use code samples and visual aids to help you visualize how the technologies work. The technologies we will be talking about are:

* MongoDB
* Firebase
* NodeJS
* ReactJS
* Spring Boot
* Postman
* Robo3T
* Atlassian Jira
* GitHub
* GitHub Desktop
* Docker
* Amazon Web Services (AWS)
* Visual Studio Code
* IntelliJ

**Database**

The data tier refers to the processing of data that have been stored in the database. The user will be able to perform CRUD (create, read, update, delete) operations on the application. We have two databases setup for our project. Our account information is kept in MongoDB, while our website data is kept on Firebase.

MongoDB



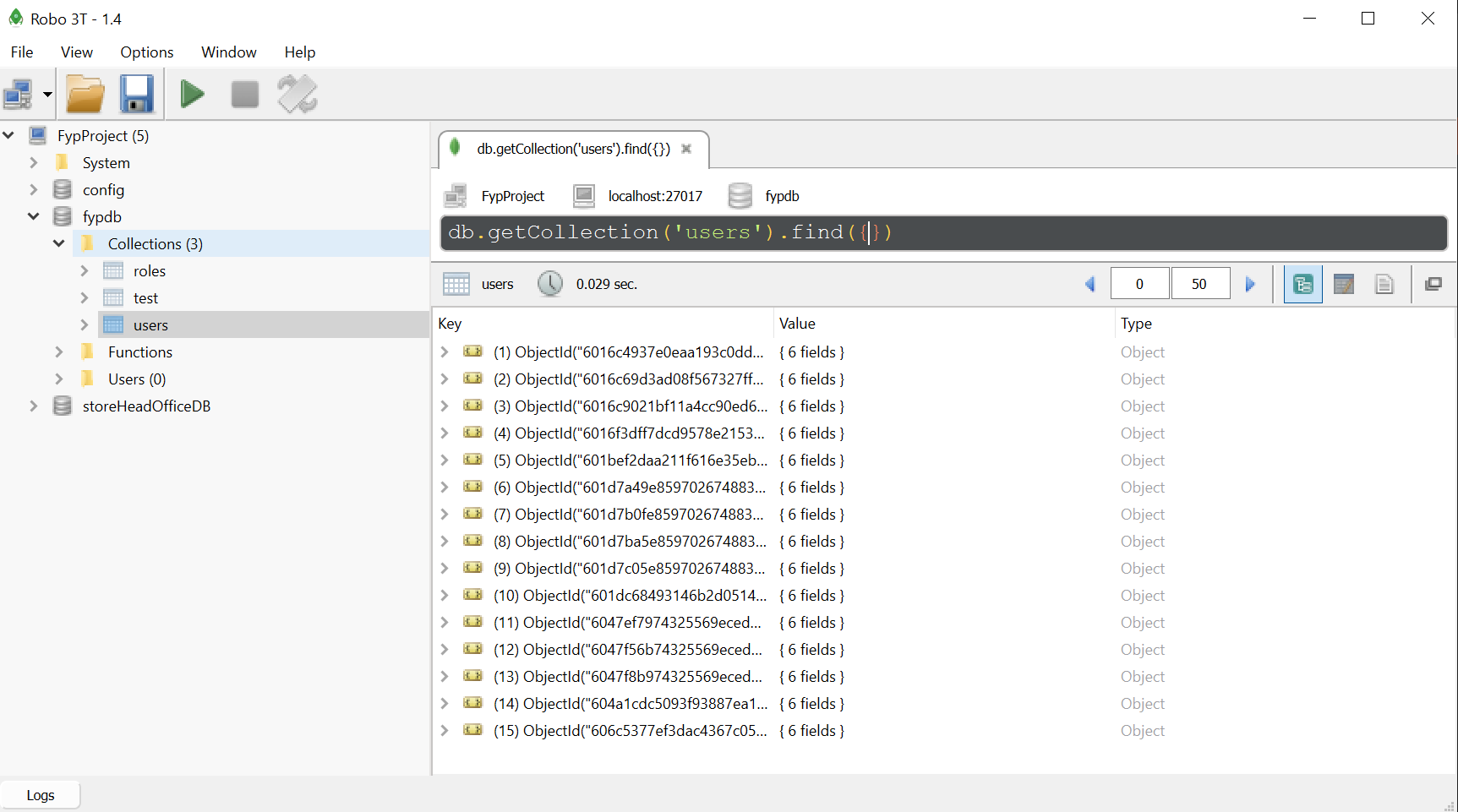
MongoDB is a document database, meaning it contains information in JSON like formats. According to MongoDB, “We believe this is the most natural way to think about data and is much more expressive and powerful than the traditional row/column model”. MongoDB's goal is to store and recover data easily so that developers can switch and program quickly. MongoDB is a NoSQL database, which means it isn't tabular and doesn't store data in the same way as relational tables do. MongoDB has several features, including horizontal scaling (distribution of data across multiple devices allowing for a better distribution system) and load balancing. It can support modular schemas and is readily elastic when dealing with massive volumes of data and heavy user traffic. It allows data to be viewed instantly after being inserted into the database. MongoDB allowed us to change the structure of the database as the project progressed. Ad-hoc queries can be used by developers to obtain real-time metrics. Ad-hoc is a short-term command that is dependent on the value of a variable. With these features MongoDB is a very efficient software that helps us to view and analyse our data.

As previously said, MongoDB stores all of its data in JSON format. JSON uses fewer data in general, lowering costs and reducing storage requirements. As a result, JSON parsing is faster than other formats such as CSV and XML. No matter what programming language you use, JSON is easier to read and map to domain properties. The mongo database was originally set up locally to validate the project, but after the initial implementation stage, we migrated the database to AWS cloud using a docker images.

Robo3T

Robo3T (originally known as Robomongo) helps users to communicate with MongoDB data using visual indications rather than a text-based interface. It has a graphical user interface that is like that of a desktop graphical user interface. It supports cross-platform applications, which means it can incorporate the mongo shell into its user interface for text-based or graphical interaction.

While testing the project, we used Robo3T to keep track of our data. We had to make sure that the users were saved to the database and that the passwords of the users were hashed. We have used Robo3T to retrieve data and build new columns and tables. In the figure below we can see that there are several users saved in the users section of the database

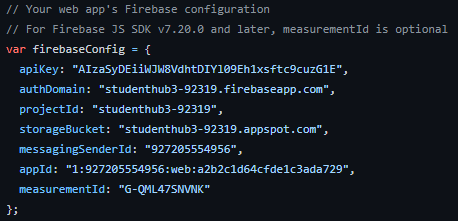


Firebase



We chose firebase to handle all the data on our web app such as the student forum page or the student notepad element. Firebase is a database provided by Google to enable developers to create high-quality web apps easily. It has a variety of tools and services that aid the developers to grow their user base and create quality apps. Just like MongoDB, Firebase is also a NoSQL database and stores data in JSON format. Firebase has a lot of useful features that make developing more fun. The data is synced in real time and remains available even when the app goes offline. It can provide fast hosting for a web app and makes the whole hosting process much easier.

Software development kits (SDK) are used by Firebase to provide methods for generating and handling data. These SDKs will submit information to Google's cloud database, which can then be shared with other web users. All users share a single database instance and are immediately updated with the most recent information. If the user loses connection, the data will still be transmitted to the database. Below are details of the app that we created in firebase and the SDK to that app.



Version Control Manager

GitHub



GitHub is a collaborative code hosting site. It allows you and others to collaborate on projects from any place at any time. For Git commands, GitHub provides a graphical user interface (GUI). It is used as a version control manager, as previously mentioned. Java, Docker, Python are a handful of the languages that can be hosted on GitHub. Some of the features available to GitHub include, following other users, forking other people's repositories, subscribing to other’s projects. Since it is widely used in the industry, GitHub is an excellent skill to acquire. Cloning other people's repositories is a fantastic feature of GitHub; it allows you to get another person's code and make improvements to it in your repository.

These are some of the most commonly used GitHub commands when working on a project:

* Git add .

Adds all the files that are not already on the GitHub Repository.

* Git commit -m ”First Commit”

Commit all the changes and give the commit a message.

* git push

Push all the changes to your repository on GitHub

GitHub Desktop



Along with all of this, GitHub has created GitHub Desktop, a software program that works in the same manner as their website. It's available as an interface which gives you easy access to GitHub repositories. We decided to focus on the desktop edition because it's easier to read and doesn't require the use of the command line for any inserting, committing, pushing, or pulling. It also enables users to clone any directory onto their machine and begin working on it right away. We may even inspect all of the modifications we made to the project before committing them using GitHub Desktop. If we feel that these modifications aren't what we want, we can roll back directly from GitHub Desktop, saving us the trouble of getting into the code and removing any changes we created. Many of these features are accessible via GitHub Desktop's user-friendly graphical user interface.

Integrated Development Environment (IDE)

We used two different IDEs to work on our project. We used Visual Studio Code as the main IDE for the front end and IntelliJ as the main IDE as the main IDE for the backend.

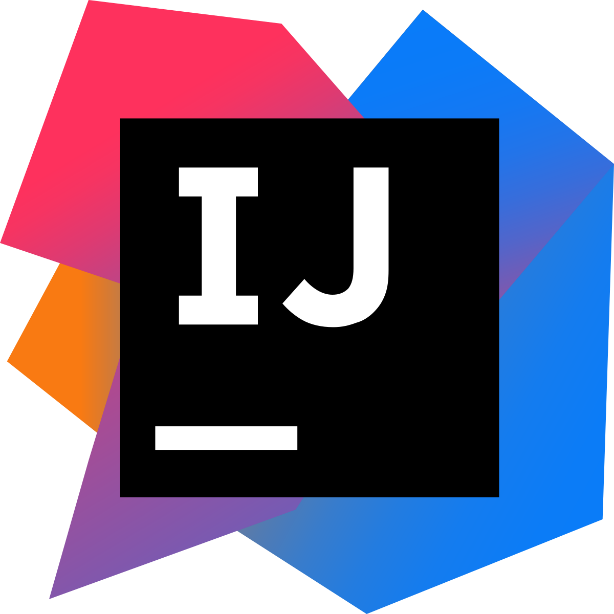
Visual Studio Code



There are a lot of great IDEs that we could have chosen, such as Atom or NetBeans but we decided that we wanted to go ahead with Visual Studio Code as our development tool when building our front end. It is a Microsoft developed source editor that has the benefit of being compatible with the most popular operating systems such as Windows, Linux, and MacOS. Visual Studio Code is very popular in the outer world as it is very programmer friendly and compared to editors like Visual Studio it is small in size and fast. Depending on your needs, the editor allows you to download several extensions. Many languages are supported by Visual Studio Code, including JavaScript, C#, Dockerfile, and CSS. Visual Studio Code can also be used to NodeJS projects as well as ASP.NET It is highly customisable, with users being able to add their own shortcuts, it is very simple to navigate and has excellent code refactoring which was very useful for us especially with larger classes.

We both have familiarity with Visual Studio Code, having used it for over two years on a variety of projects. We had tried a few extensions in VS Code and concluded that it was the better option. We also used Visual Studio Code's IntelliSense function, which allowed us to type Ctrl + Space to get code suggestions instead of typing it character by character.

IntelliJ



As our backend development tool, we went ahead with IntelliJ. It is a Jet Brains-developed source editor that is compatible with most operating systems, much like Visual Studio Code. We did not have much experience with IntelliJ before, but it is a very simple and enjoyable editor to use. It has a fantastic smart completion feature that provides you with a list of the most important symbols for the current situation. IntelliJ has dedicated keyboard shortcuts for the most part, which helped us save time during the development stage. You are provided with a built-in terminal and can use command line, bash, or PowerShell depending on the project. Another great feature of IntelliJ is its inline debugger. When debugging the code, IntelliJ IDEA displays variable values alongside their usages in the source code. Every time a variable's value varies, the IDE shows it in a different colour so you can see how the condition changes in the code. This came in very handy as we were checking unique features within our project.

Errors and misunderstandings with the coding are likely to happen with large projects like this. With their fast fixes, IntelliJ provided us with a solution that was really successful. A lightbulb can appear if you are about to make an error and clicking on it will bring up a series of steps you may take to correct the mistake. When we were researching various IDEs, we discovered that IntelliJ has a separate Docker plugin, which was one of the main reasons we picked IntelliJ for backend. The plugin connects to a running Docker machine on your local network and handles its core services including images, containers, and most importantly Docker compose.

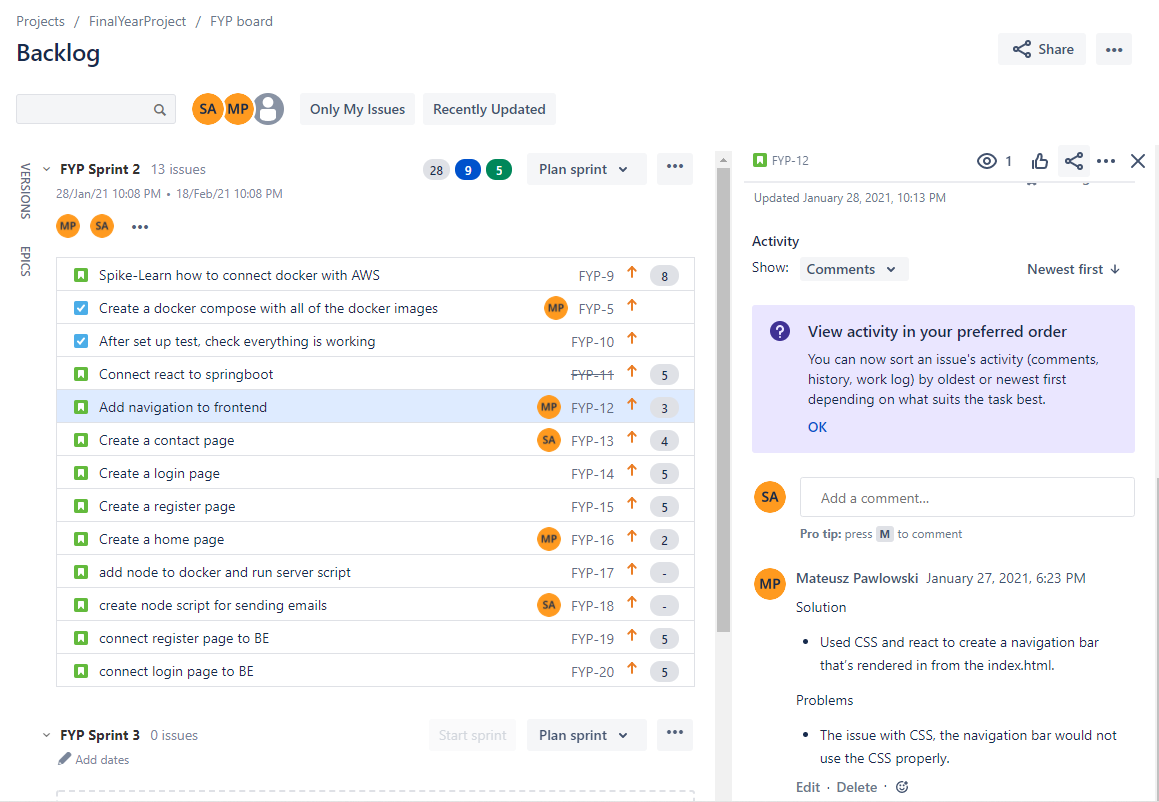
**Others**

Jira Software



Atlassian's Jira is a project management platform that has a problem and error monitoring framework. It assists teams of all sorts in managing their workload and staying on top of things. Jira can be used by developers to create specifications, handle test cases, and automate testing. It's ideal for agile growth. Jira has a number of quality assurance add-ons, including problem monitoring, Kanban forums, scrum boards, and progress reporting.

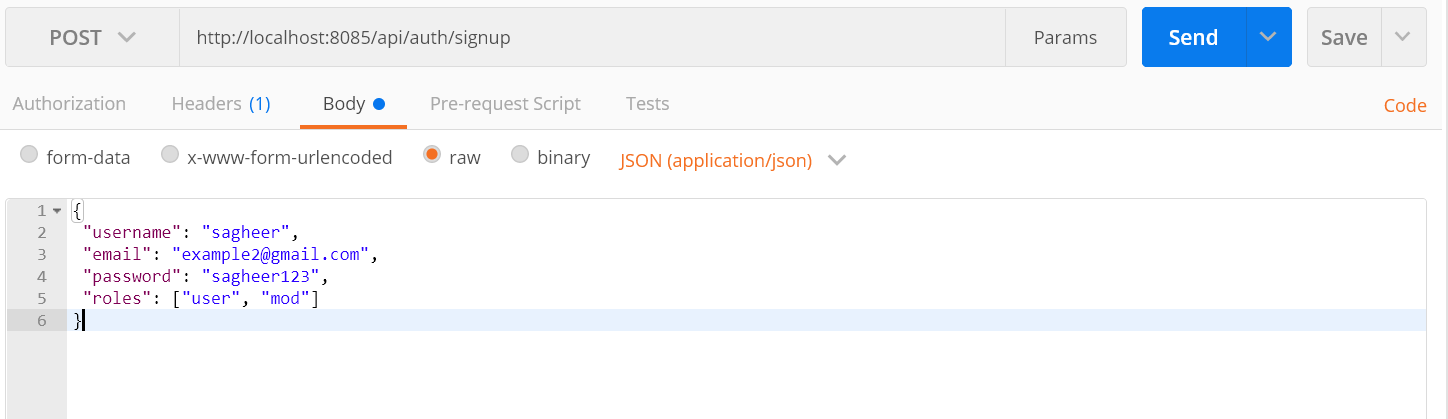
From the beginning of our project, we used Jira to handle our specifications. We maintained meticulous records of everything we did, noting which challenges we encountered and how we resolved them. We divided the project into a number of smaller tasks. The task was then given a story point. The story point indicates how much time we can spend on each task; for example, if a task has a story point of 1, it should only take us a day to finish it, while a task with a story point of 3 should take us a week to complete. It also encouraged us to divide the work between ourselves so that we knew exactly what we needed to do, and it could help us keep a good workflow.



Postman



Postman is a collaboration platform for API development. We've been using Postman since the beginning of development. We initially used it to verify whether our mongo database was linked to our back end by sending Get and Post requests. The next step was to see how we could submit user information in JSON format to register a user and then use the same information to login. We gave Postman the link to our back end and sent a request to that link to see if we can get a response.



**Cloud Hosting**

We had to use two separate platforms to host our web application in the cloud. We were able to host our website and enable users to access it from anywhere, at any time, thanks to these technologies. These two technologies were Docker and AWS.

**Docker**



We have to run each part separately because our project uses several functionalities such as ReactJS, MongoDB, and Spring Boot. This became a very tedious and time-consuming procedure as the project grew in size. We looked into this issue and discovered Docker. Docker is a popular open-source project written in Go and created by Dotcloud. It is essentially a container engine that creates containers on top of an operating system using Linux Kernel features such as namespaces and control groups. This was the ideal solution to our dilemma because it saved us time from having to execute each function separately.

In order to use Docker, we had to first generate images for the backend and frontend, which were then uploaded and stored onto our Docker Hub account. In order to create the images, we had to run the following commands:

* docker build -t sagheergmit/frontendfyp:latest .
* docker build -f Dockerfile -t sagheergmit/final-year-project .

These commands generated images for the project's frontend and backend. We didn't need to build a mongo image because one is already available for public use by default. We then generated a new file on our local machine called docker-compose.yml that would use these images to execute the web application as a single file. This was ideal because we didn't have to run each project part individually.

Originally, we were running the project locally, but as it progressed, we began to use the cloud. We had to make a few adjustments to our code and remake the images which would be pushed up to Docker Hub. To run the application on cloud, we had to create a new docker-compose file on the aws instance. The images were pulled from our Docker Hub account using the command “docker-compose pull”, this pulled all the required images and saved them to the instance. To run the images on the designated port we had to use the command “docker-compose up”.

We had never used Docker before this project, but after doing some research, we discovered that most businesses use it to create images and run them in the cloud. As soon, we will be applying for jobs in this industry, we decided it would be a great idea to have Docker covered. Docker has a number of benefits, the most important of which is that it speeds up the whole web hosting process, allowing you to devote more time to other activities. It has a quick and simple setup for cloud deployment, a secure and remote environment, it is very scalable, and has a rollback option if ever in need, among other features.

**Amazon Web Services (AWS)**



Amazon Web Services is a pay-as-you-go cloud computing network that allows customers to host web applications. Millions of people depend on it to fuel networks and applications. AWS enables developers to cut prices, improve agility, and innovate more quickly. Moving apps to the cloud becomes faster and more cost-effective as a result of this. This online portal for cloud computing provides a diverse set of tools and modules. Two of these platforms are Amazon Elastic Compute Cloud (EC2) and Elastic Beanstalk (EB). Customers may use these programs to build a virtual cluster of computers in the cloud that is always available to them.

We looked at all of these services to see which will be the most appropriate for our project. We used EC2 to set up an instance on which we could run our web application. We had to first consider what kind of instance we wanted to create. Either a Windows or a Linux instance had to be chosen. We chose Linux over Windows because we believed the instance would run more smoothly on Linux. After selecting the instance type, we had to choose between the free and paying tiers. The free tier was our first option. The instance was set up and configured to run NodeJS and Docker Compose. To set these up, we needed to run a few commands:

* sudo apt-get update
* sudo apt-get install \ apt-transport-https \ ca-certificates \ curl \ gnupg-agent \ software-properties-common
* curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
* sudo apt-get install docker-ce docker-ce-cli containerd.io
* apt-cache madison docker-ce
* sudo apt-get install docker-ce docker-ce-cli containerd.io
* sudo apt install docker.io
* sudo apt install docker-compose
* npm install nodejs

We were able to customize the instance using these commands. Then we attempted to run the docker compose file on the newly generated instance. The web application turned out to be far too large for the free tier instance. The instance was extremely slow, failing to even load the home page, much less process any requests.

After that, we decided to start over and build a new instance. We chose the medium tier this time, which would cost us around 20c per hour. We chose a Linux instance once more and installed it with the same configurations as before. The instance was then used to build and run a Docker Compose file. We were able to navigate the website from the cloud after the Docker Compose file started running, without a hitch.

**Frameworks**

We used a few different libraries and frameworks while working on this project. These are Spring Boot, ReactJS and NodeJS.

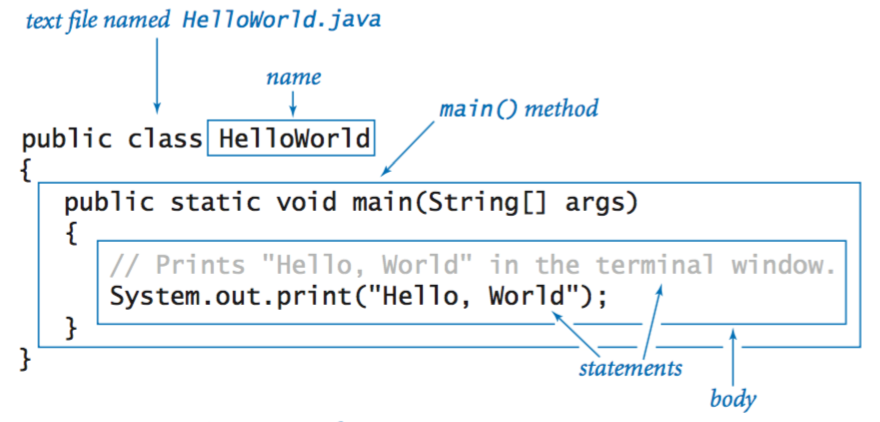
**Spring Boot**



Spring Boot is a java-based open source framework for developing microservices. It's also used to make stand-alone, industrial-grade spring applications. The majority of Spring Boot applications only require a few configurations to setup and run. Spring Boot has a variety of features. You don't need to deploy any WAR files because Spring Boot will embed Tomcat directly into the program. Tomcat is a Java application server that runs servlets and renders web pages using java server page coding. Spring Boot also comes with starter dependencies, which will make building applications easier. It will also dynamically configure any third-party libraries, stopping the program from crashing due to errors. Spring Boot also doesn't require any code generation or XML setup.

**Java**

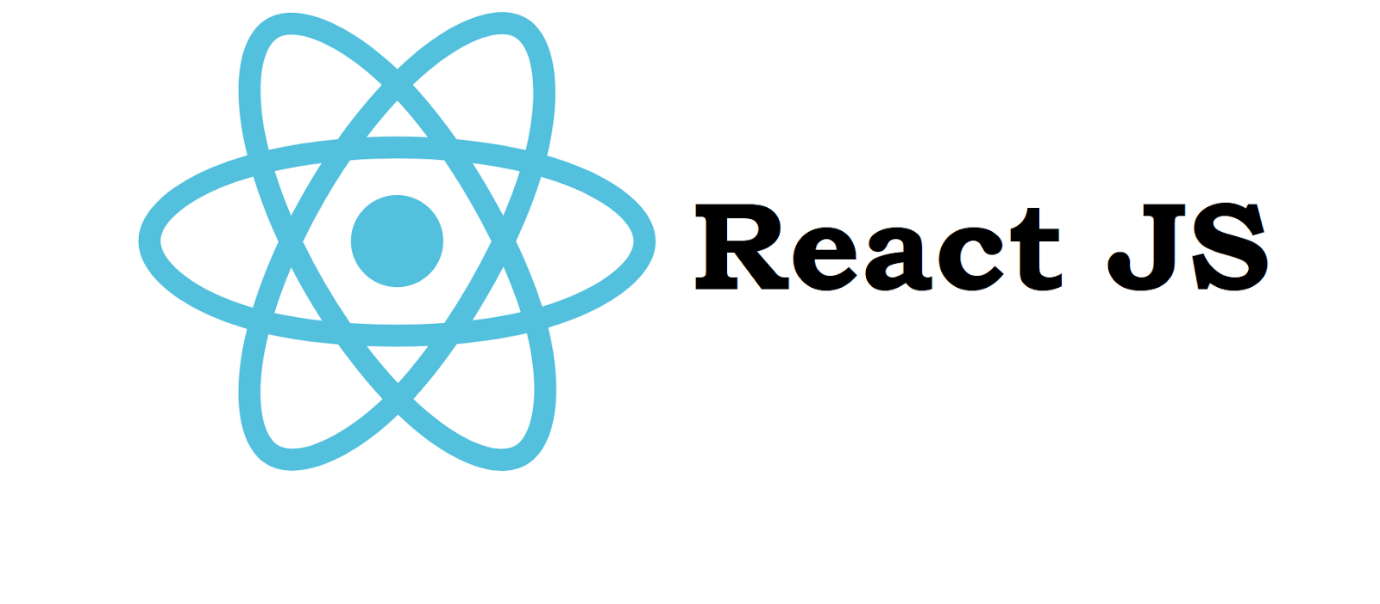
Spring Boot is written in Java. Java is a concurrency-oriented, object-oriented, and class-based programming language. Sun Microsystems first published it in 1995. There are several technologies that we use on a daily basis that would not function without Java, ranging from mobile phones to game consoles to supercomputers, and more are being developed every day. Since then, Java has risen in popularity and is now one of, if not the most common, programming languages.



For the back end of the project, we're using Java and Spring Boot. Both the front end and the mongo database are connected to our Spring Boot application. The Spring Boot program receives a register request from the front end and transfers it to the back end. An authentication controller is used by the Spring Boot application to process this incoming request. The authentication controllers would then submit this information to the user repository to see if the user's information is valid. If the user continues, the controller will respond with a message informing them that they have already signed up with these credentials. If the information is unique, the password is hashed, and the information is stored in the Mongo database. As a default, the user is given the role of user.

If the user sends a login request, the authentication controller will manage it once more. It will submit this request to the user repository to see whether the specifics match any past users; if they do, the user will be able to log in successfully; if they don't, the user will get a "incorrect details" response.

**ReactJS**

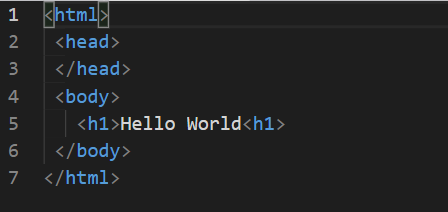


React is a JavaScript library for building user interfaces and UI modules that is open source and component oriented. Jordan Walke, a Facebook software developer, developed React. Facebook and a consortium of small developers and companies are now in charge of it. React is mainly used to develop single page websites and smartphone apps. It is arguably the most popular framework in the world for web application, development which means there are a plethora of tutorials and guides available. Netflix, Instagram, and Airbnb are only a few examples of major existing businesses that use React. The Model View Controller architecture is used by React, which ensures that the app's view layer is in charge of how it looks and feels.

We used a variety of languages and styling sheets during our React development stage, including HTML, JavaScript, and CSS/SASS/SCSS.

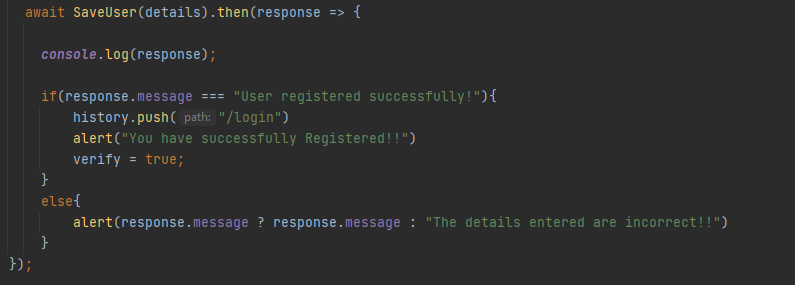
HTML

HTML stands for hypertext mark-up language, and it is used to create documents that are meant to be viewed through a web browser. It can be aided by technologies like CSS for styling and JavaScript for complex functionality implementation. HTML includes all of the required elements for anybody to create tables, lists, add pictures/videos, simple bold or italic text, buttons, check boxes, and so on. It contains the fundamentals of any programming language, is extremely simple to learn, and is supported by all web browsers. HTML 5 is the most recent version, which was first released in 2014.



JavaScript

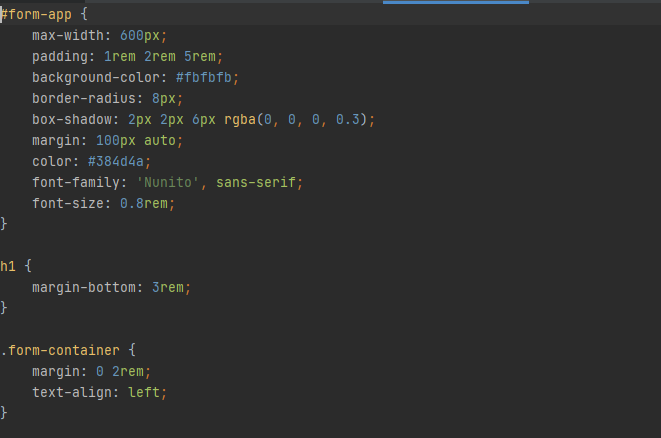
JavaScript, or JS for short, is an interpreted programming language that adheres to the ECMAScript principles. It is the world's most commonly used programming language. Though HTML and CSS give a web page layout and style, JavaScript allows you to add complex interactive elements that are appealing to the user while being simple to manipulate and use. Importing JavaScript into your code greatly increases the user experience of the web page through translating a static web page into an interactive one. Javascript is a scripting language that is both lightweight and complex. It supports the use of an object-oriented programming approach. One of the core languages used by the ReactJS library is Javascript. It is more widely used than HTML and CSS by ReactJS.



CSS/ SASS/ SCSS

Within our project, we employ a variety of styling techniques; the following is a brief description of some of the styles we implement:

* CSS, or cascading style sheets, is a term for describing the presentation of text written in a mark-up language like HTML. CSS is not limited to HTML and can be used in any XML-based markup language. CSS is used to adjust the look of a page's colors, backgrounds, and fonts, among other things.
* SASS (Syntactically Awesome Style Sheets) is a pre-processor scripting language with more capabilities than regular CSS. It offers a more elegant CSS syntax. SASS creates CSS after it has been compiled.
* Sassy Cascading Style Sheets, or SCSS, is a newer CSS syntax and extension. Unlike other styling sheets, it supports nesting rules, inline imports, and variables. It also aids in keeping things more organized and speeds up the development of style sheets.



**NodeJS**

****

NodeJS is a backend Javascript runtime server that was built on Google Chrome’s V8 Javascript Engine. It was developed by Ryan Dahl in 2009. It is mainly used for event driven servers because of its single threaded nature. We used NodeJS to create and run a server that would allow us to create a fully functional contact page. The server uses external libraries such express and NodeMailer to send an email. It runs on port 5000. When the user fills in the detail on the contact page and presses send, it sends a request to this server, which in return sends an email to our Student Hub Gmail account.

Resources

<https://www.mongodb.com/what-is-mongodb/features>

<https://www.mongodb.com/nosql-explained>

<https://blog.cloud-elements.com/using-json-over-xml#:~:text=JSON%20uses%20less%20data%20overall,language%20you're%20working%20with>.

<https://firebase.google.com/>

<https://www.educative.io/edpresso/what-is-firebase>

<https://firebase.google.com/docs/database>

<https://scalegrid.io/blog/how-to-connect-your-mongodb-deployments-to-robo-3t-gui-at-scalegrid/#:~:text=Robo%203T%20(formerly%20Robomongo)%20is,of%20a%20text%2Dbased%20interface>.

<https://www.atlassian.com/software/jira/guides/use-cases/what-is-jira-used-for>

<https://www.postman.com/>

<https://spring.io/projects/spring-boot>

<https://java.com/en/download/help/whatis_java.html>

<https://www.simplilearn.com/tutorials/reactjs-tutorial/what-is-reactjs>

<https://alistapart.com/article/why-sass/>

<https://www.w3.org/standards/webdesign/htmlcss>