National College of Ireland

Software Project

Business Information Systems

2020/2021

Nathan Hodgkiss

x17381176

x17381176@student.ncirl.ie

Smart Coding Journal

Technical Report

Table of Contents

Executive Summary 3

1.0 Introduction 4

1.1. Background 4

1.2. Aims 5

1.3. Technology 6

1.4. Structure 7

2.0 System 7

2.1. Requirements 7

2.1.1. Functional Requirements 8

2.1.1.1. Use Case Diagram 8

2.1.1.2. Requirement 1: Programming Journal 9

2.1.1.3. Description & Priority 9

2.1.1.4. Use Case (1) 9

2.1.1.5. Requirement 2: Resources 11

2.1.1.6. Description & Priority 11

2.1.1.7. Use Case (2) 11

2.1.1.8. Requirement 3: Create A Chat Room 13

2.1.1.9. Description & Priority 13

2.1.1.10. Use Case (3) 13

2.1.1.11. Requirement 4: Join A Chat Room 15

2.1.1.12. Description & Priority 15

2.1.1.13. Use Case (3) 15

2.1.1.14. Requirement 5: Notes 17

2.1.1.15. Description & Priority 17

2.1.1.16. Use Case (3) 17

2.1.2. Data Requirements 19

2.1.3. User Requirements 19

2.1.4. Usability Requirements 19

2.2. Design & Architecture 20

2.3. Implementation 22

2.4. Graphical User Interface (GUI) 26

2.5. Testing 30

2.6. Evaluation 38

3.0 Conclusions 42

4.0 Further Development or Research 43

5.0 References 43
6.0 Appendices 44
6.1. Project Plan 44
6.2. Project Proposal 45
7.0 Objectives 47
8.0 Background 48
9.0 Technical Approach 50
10.0 Project Plan 51
11.0 Technical Details 52
12.0 Evaluation 52
12.1 Declaration of Ethics Consideration 53
12.2 Reflective Journals 59

Executive Summary

This report will highlight the lifecycle of the project ‘Smart Programming Journal’, including an introduction into the project which will give details on the technologies used in order to achieve the functionality set out in the project requirements. The report will then continue to give an overview on the system developed, by outlining key use cases, non-functional requirements, implementation details, all forms of testing utilised, and then an evaluation of the systems performance. The implementation details that the report covers include how the natural language processing aspect of the programming journal was implemented, and also how the AI voice assistant was implemented into the web application. The report concludes with a summary of the strengths and weaknesses off the developed system, with the strengths including a complete application according to the initial requirements, with additional functionality added on top, other strengths include unique functionality such as the smart programming journal and the custom voice assistant which has great accessibility uses. The third strength listed in this report is that the web application has been developed according to best practices, which has resulted in high-performance and secure application. The report then concludes with where I feel the application could do with further development in order to add value to the system.

1.0 Introduction

1.1. Background

The idea of this application came to fruition when thinking of what kind of application could be useful for people who are interested in learning to code in these heavily digital times. As I continue to learn to code in my spare time as I am a Business Information Systems student and not a pure Computing student, often I do so when discussing with my friends, I felt an application that provided me with resources relating to coding and also allowed me to chat with my friends could be of great use. There are two other primary competitors for this type of application, Slack & Discord. Slack targets business users, allowing for third party applications to be integrated, to enhance the user's business workflow. Discord generally targets users who are gamers, who are part of large communities. Similarly, to Slack, it offers both text and voice chat to users. One of the key aspects of both applications is that they offer clean minimalistic user interfaces, my application will take inspiration from this aspect.

Slack(L)/Discord(R)

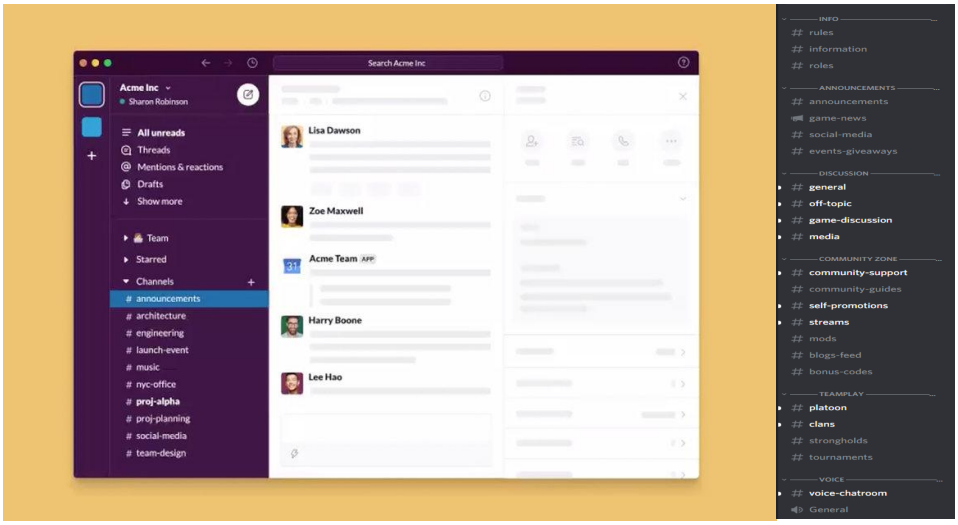


Figure 1

I decided to undertake this project as I was interested in learning new technologies, primarily React. The idea of building a web application came from the fact that I am targeting a career as a software developer/web developer after graduating from college. My interest in these technologies and this career path stemmed from the fact that I completed a six-month web development internship during my third year in college, which exposed me to modern web development, and allowed me to work with experienced software developers/web developers. To conclude, the background off this application comes from my personal interest in coding and learning new technologies, combined with the experience from my internship.

1.2. Aims

The objective of this application is to develop a high-quality accessibility focused piece of software that allows users to access a range of materials relating to learning to code, while also allowing users to access personal management tools and chat rooms. The application strategy is to focus on a very specific target audience, that will allow me to develop specific functionality, therefore differentiating my application. The target audience for this application is anyone who is learning to code with a group of friends/people they know. The chat room aspect could be used in a class environment, or it could simply be used amongst a group of friends working on a project, or who have a shared interest in learning to code.

As briefly mentioned, I intend to develop this application with an emphasis on accessibility, as it is quickly becoming increasingly more important, and will allow me to practice best practices. The technology that I will use to give my application a differentiating feature, is called AlanAI. This piece of technology will allow me to implement a voice assistant feature, that can allow the user to navigate parts of the application and ask some pre-defined questions. This will result in a better user experience and help with the usability off my application to a larger audience.

Another high-level objective is to successfully implement new technologies, such as React, AlanAI etc into my application that I have not previously worked with, to further expand my skillset and enhance the richness off my application. By using new technologies, it ensures that my project will be more unique than my previous projects and will also allow me to advertise my skills to better effect after graduation. I intend to implement modern React to great effect, including hooks, custom hooks etc. The implementation off innovative JavaScript libraries such as Compromise, will allow me to implement key functionality such as the analyses of users input in the programming journal section.

1.3. Technology

Front-End: React, Redux, Compromise.

Back-End: Firebase, AlanAI.

Testing: Cypress, Jest.

Other: Material UI, React Bootstrap, SCSS, HTML, JSX, Balsamiq, Google Books API

The application presented is a web application, and therefore the technologies used are primarily web based. The wireframe mock-ups for this application were created using Balsamiq, in order to gain feedback from my academic supervisor.

The user interface of the application has been built using the React framework. React allows the developer to programmatically update the state of the application, which is a powerful tool for more advanced applications. I have also made use of React Routing in order to dynamically update what is displayed on the screen. I will also have used Redux for managing the routing within my application. I have also used of the JavaScript library 'Compromise', which is a natural language processing library. I have implemented compromise in my programming journal section, as a means off analysing user input and giving a suggestion through a tooltip if possible.

The back end of the application consists of Firebase and AlanAI. Firebase allowed for the authentication aspect of the application, as they offer an excellent set of integrated tools. For the chat room aspect of the application, I have used Firebase Cloud Firestore, which is a NoSQL database, the structure of this database works well for a chat like functionality, as data is stored in a collection of 'documents', as such, chat rooms will be 'documents', which will have a collection of 'documents' for the messages in each chat room. Each document then stores information. Another technology that I will be using on the back end is AlanAI. AlanAI allows developers to create their own custom voice assistant for their applications.

For testing my application, I will be using Cypress and Jest. Cypress is an end-to-end testing framework, that runs in a browser, which makes testing far easier as you do not need to write implicit wait times etc. Jest is a JavaScript testing framework; I will be using Jest for my unit and integration testing with my React components.

1.4. Structure

Section 1.0 outlines the introduction for the project name, detailing the background and aims of the project, along with what technologies are intended to be used and how they will fulfil the relevant functionality. Section 2.0 outlines details on the system, including requirements gathering, which range from functional requirements to usability requirements, along with all the associated use cases. This section also includes details on the implementation of two key functionalities, which are the natural language processing implementation in the programming journal, and also the custom voice assistant that I implemented into the application. Also included in this section is screenshots of the application's graphical user interface, along with a brief description of each of the main user interfaces. The system section also includes details on the testing methods used in the application, which I have sectioned into unit testing using Jest, and integration/end to end testing using Cypress. Finally, this section also includes an evaluation off the system which has been sectioned into performance, security, SEO and mobile. Section 3.0 contains the conclusion for the report, which outlines both the strengths and weaknesses of the application. Section 4.0 details what further development I would complete on the application, in order to increase its value and performance.

2.0 System

2.1. Requirements

Programming Journal: Users should be able to access a programming journal section, where they can input errors that they encountered, and how they fixed them. The system should then give suggestions if no solution is inputted using AI.

Resources: Users should be able to access a resources section, where users can search a book and gain access to that book without formal training.

Notes: Users should be able to access a note section, where users can create and modify notes.

Chat: Users should be able to input text into a chat in a chatroom.

2.1.1. Functional Requirements

Programming Journal: System should take user input, store user input, analyse that user input, then output a suggestion.

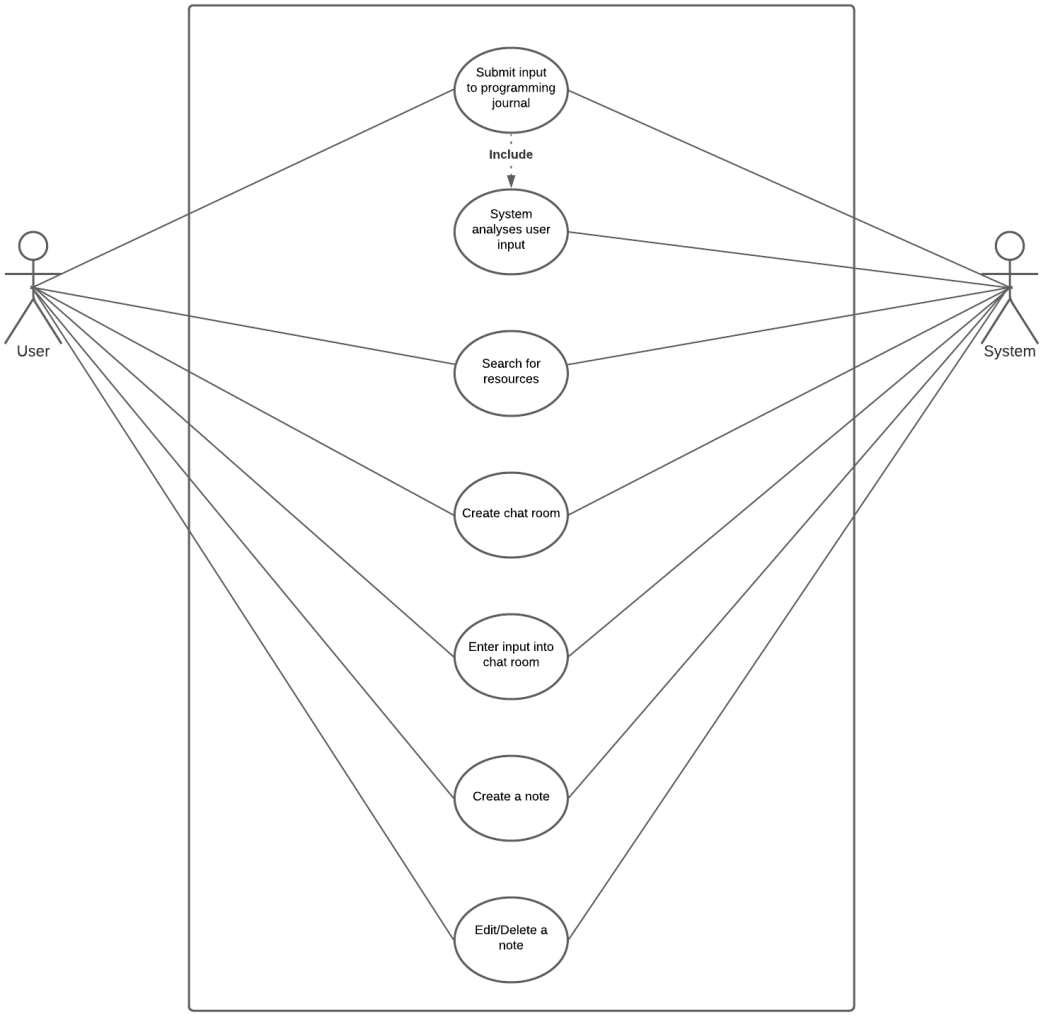
Resources: System should take user input, then make an API call to Google's API Library, return a result.

- When user selects book from returned API call, book is embedded on the page.

Chat: System should allow users to create chat rooms, then allow users to input text into that chat room. This will be achieved through CRUD functionality to the database.

Notes: System should allow users to create, edit and delete notes. This will be achieved through a library called React Quill and CRUD functionality to the database.

2.1.1.1. Use Case Diagram



2.1.1.2. Requirement 1: Programming Journal

2.1.1.3. Description & Priority

The purpose of this system is to take user input, store user input, and complete an analysis on this input using AI. High priority for the overall system.

2.1.1.4. Use Case (1)

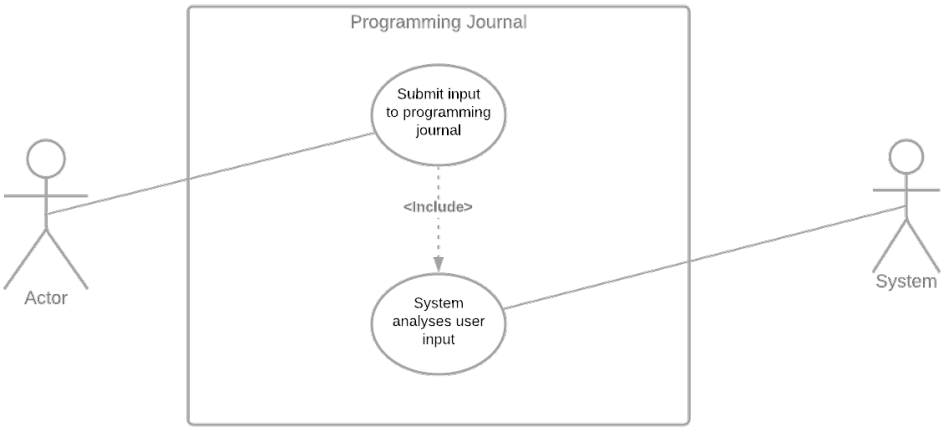
Scope

The scope of this use case is to accept user input, store user input, analyse user input. User input will be errors that the users has encountered while programming.

Description

This use case describes the flow of the user from inputting text, to storing that text, to completing an analysis on that text using AI.

Use Case Diagram



Flow Description

Precondition

The user is registered and logged in.

Activation

This use case starts when a user selects the 'Programming Journal' section.

Main flow

1. The system displays a UI.
2. The user enters error encountered.
3. The user enters their solution. (See A1).

4. The system then stores the journal entry, with a logged date.

Alternate flow

A1: <User does not enter a solution>

1. The user has no solution to enter.
2. The system then analyses the error and suggests a fix. (See E1).

Exceptional flow

E1: <System cannot suggest a fix>

3. The system analyses input but cannot suggest a fix.
4. System outputs a default message
5. The system then stores the journal entry, with a logged date.

Termination

The system presents the original UI.

Post condition

The system goes into a wait state.

2.1.1.5. Requirement 2: Resources

2.1.1.6. Description & Priority

The purpose of this requirement is to allow the user to search for resources through user input. Medium priority for the system.

2.1.1.7. Use Case (2)

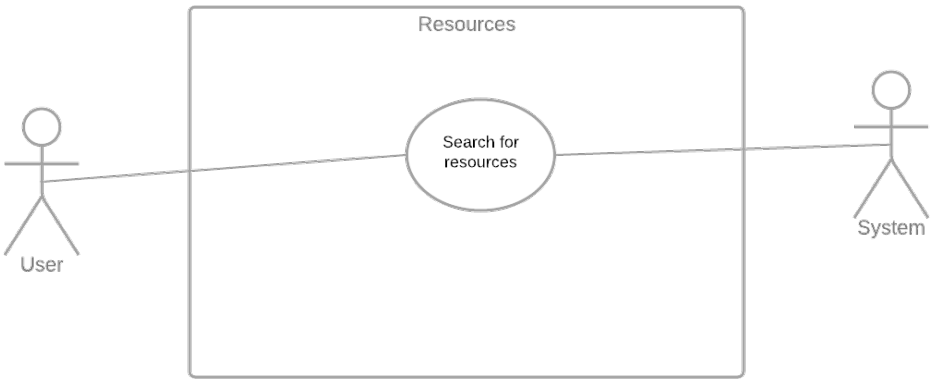
Scope

The scope of this use case is to allow users to enter input, which is then used in an API call to search for resources.

Description

This use case describes the flow from user input to API call, to then the result being shown on screen.

Use Case Diagram



Flow Description

Precondition

The user is registered and logged in.

Activation

This use case starts when a user selects the 'Resources' section.

Main flow

1. The system presents a UI.
2. The user enters input into the search bar.
3. The system returns a list of results (See A1).
4. User selects the appropriate resource.

Alternate flow

A1: <System does not return a result.>

6. The system cannot return a list of results.
7. The system displays an error message.
8. System returns to original UI.

Exceptional flow

-

Termination

The system presents the list of results.

Post condition

The system goes into a wait state.

2.1.1.8. Requirement 3: Create A Chat Room

2.1.1.9. Description & Priority

The purpose of this requirement is to allow users to create chat rooms. High priority for the system

2.1.1.10. Use Case (3)

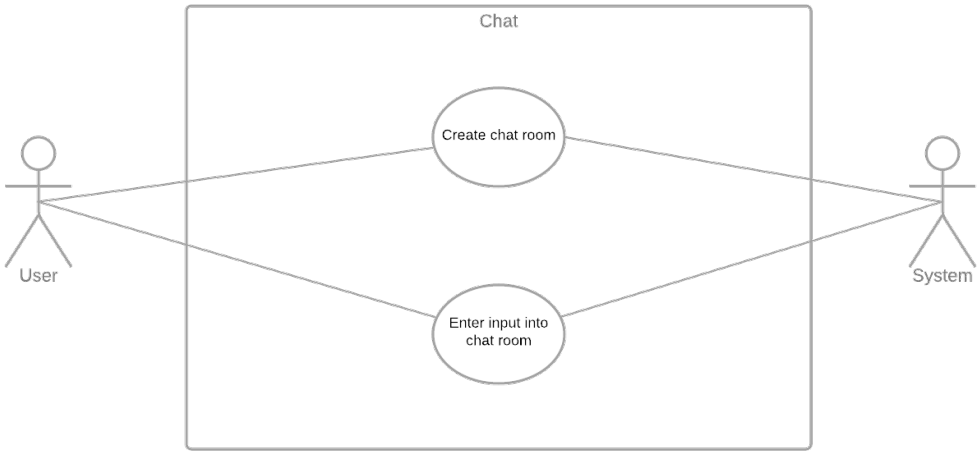
Scope

The scope of this use case is to allow users to create and view chat rooms.

Description

This use case describes the flow from creating a room/entering text into a room.

Use Case Diagram



Flow Description

Precondition

The user is registered and logged in.

Activation

This use case starts when a user creates/joins a chat room.

Main flow

1. The system presents a UI.
2. User creates a chat room (See A1).
3. User enters details.
4. System presents UI of chat room.

Alternate flow

A1: < Chat room is already created >
1. The user creates a chat room.
2. System presents an error message with ‘chat room already exists’
3. Return to stage 1 main flow.

Exceptional flow

E1: <No chat room is available with that name>
1. The user attempts to create a chat room.
2. System displays an error message.
3. Return to original UI.

Termination

The system presents the chat room.

Post condition

The system goes into a wait state

2.1.1.11. Requirement 4: Join A Chat Room

2.1.1.12. Description & Priority

The purpose of this requirement is to allow users to join chat rooms and enter input. High priority for the system.

2.1.1.13. Use Case (3)

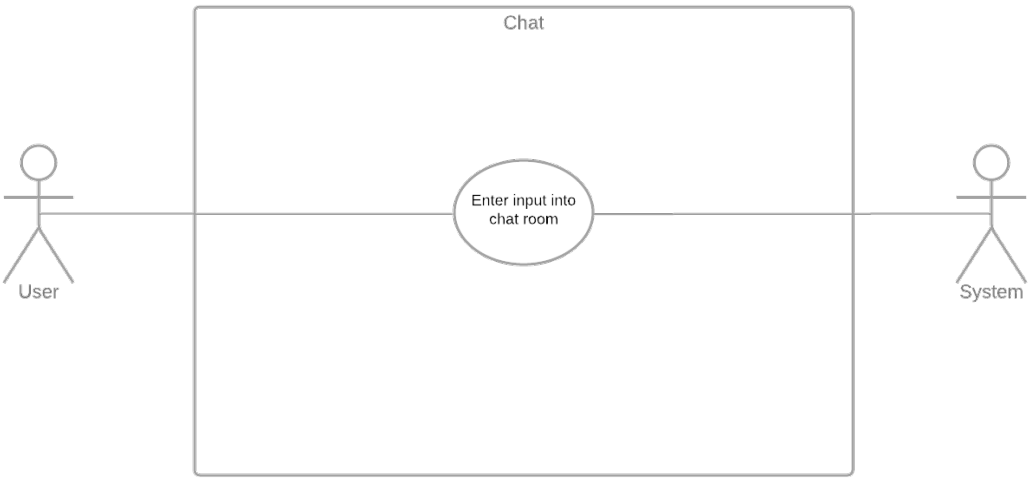
Scope

The scope of this use case is to allow users to join, view and enter input into chat rooms.

Description

This use case describes the flow from entering text into a chat room.

Use Case Diagram



Flow Description

Precondition

The user is registered and logged in.

Activation

This use case starts when a user creates/joins a chat room.

Main flow

1. The system presents a UI.
2. User joins a chat room (See A1).
3. User presented with UI of chat room.
4. User enters input into chat room.

Alternate flow

-

Exceptional flow

E1: <No chat room is available with that name>
1. The user attempts to join a chat room.
2. System displays an error message.
3. Return to original UI.

Termination

The system presents the chat room.

Post condition

The system goes into a wait state.

2.1.1.14. Requirement 5: Notes

2.1.1.15. Description & Priority

The purpose of this requirement is to allow users to use a note taking functionality.

2.1.1.16. Use Case (3)

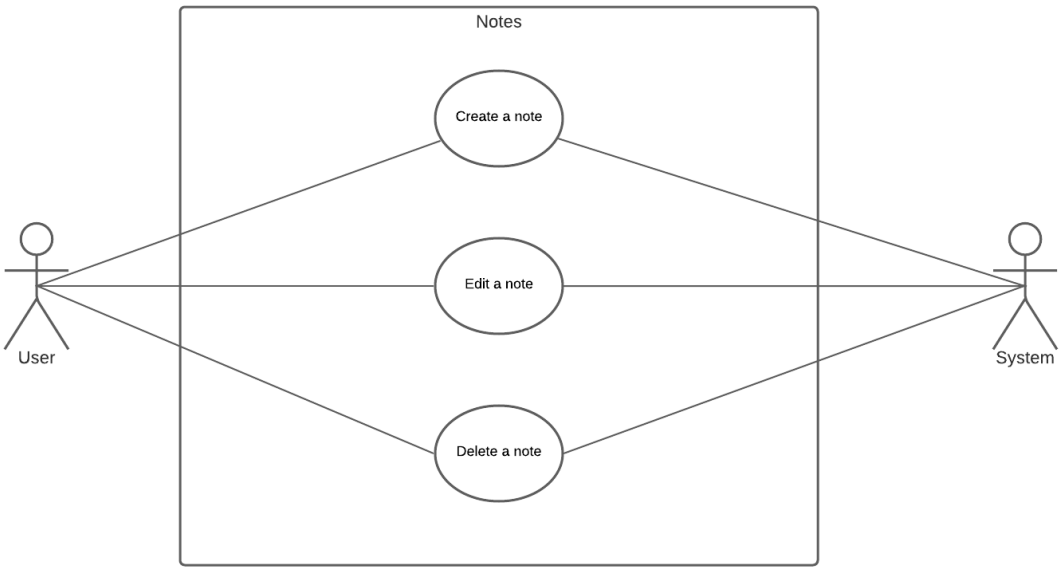
Scope

The scope of this use case is to allow users to create, edit and delete notes.

Description

This use case describes the flow from creating/editing/deleting notes.

Use Case Diagram



Flow Description

Precondition

The user is registered and logged in.

Activation

This use case starts when a user creates/joins a chat room.

Main flow

1. The system presents a UI.
2. User creates a note (See A1) / (See A2).
3. User presented with UI of note editor.
4. User enters input texts into editor.

Alternate flow

A1: User selects existing note

1. User selects existing note.
2. User presented with UI of note editor.
3. User modifies content.

A2: User deletes existing note

1. User selects existing note.
2. User selects delete option.
3. System deletes note.

Exceptional flow

-

Termination

The system presents finished note.

Post condition

The system goes into a wait state.

2.1.2. Data Requirements

Data Integrity: A data requirement for this project is to ensure that all data sent by the user has a high level of integrity.

Data Security: Another data requirement for this project is to ensure that all user details are appropriately stored.

API: This application will receive data through an API call to the Google Books API.

2.1.3. User Requirements

Programming Journal: Users should be able to access a programming journal section, where they can input errors that they encountered, and how they fixed them. The system should then give suggestions if no solution is inputted using AI. The user should also be able to access previously logged entries to their journal.

Resources: Users should be able to access a resources section, where users can search a book and gain access to that book without formal training. Users should be able to click on the card of a book and have a modal with a link to the book pop-up on the page.

Chat: Users should be able to input text into a chat in a chatroom. Users should also have the ability to create chat rooms. The ability to create these chat rooms should be obvious and intuitive. Users should also be able to switch between the various chat rooms quickly and easily.

Notes: Users should be to modify the content of a note. Users should also have the ability to delete notes. The ability to create the notes should be intuitive and accessible. Users should also be able to read all their notes easily.

2.1.4. Usability Requirements

This application should meet the following criteria with regards to usability:

- *Learnability:* The system should be easy to use without formal instruction.
- *Efficiency:* The system should allow users to quickly access the functions of the application.
- *Memorability:* The systems user interface is memorable and clean
- *Satisfaction:* The user should be satisfied when using various parts of the application.
- *Accessibility:* The system should be accessibility focused, to ensure access to the widest audience possible. This is also important for SEO based metrics. The implementation of a voice assistant will allow for a richer user experience and will also act as a major accessibility feature. Also included in the accessibility experience is an appropriate colour scheme in order to have an appropriate contrast ration on text.

2.2. Design & Architecture

Figure 2

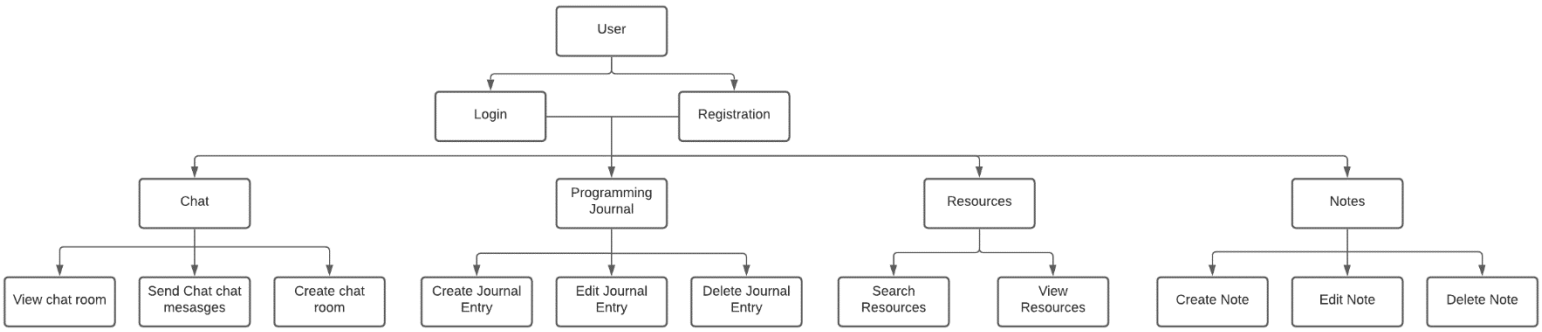


Figure 3

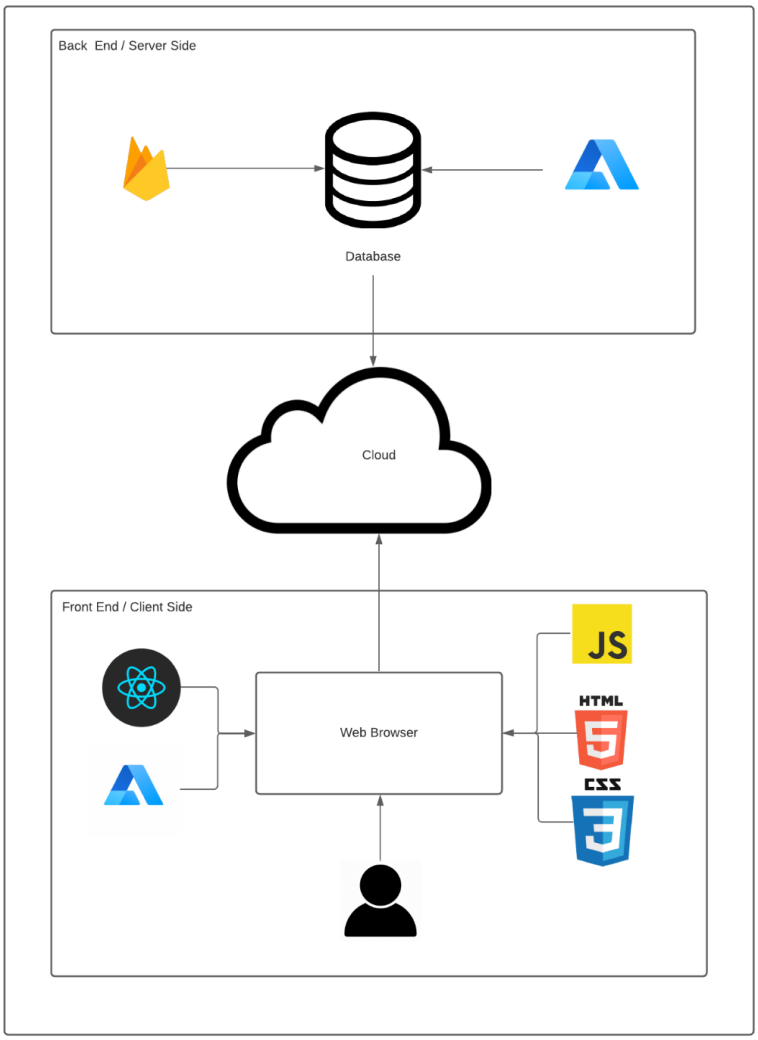


Figure 1:

This diagram describes the flow the users can follow, along with the available actions assigned to each requirement.

Example Flow 1:

Login -> Chat => Join a chat room.

Example Flow 2:

Registration => Programming Journal => Create Journal Entry.

Example Flow 3:

A third example of a flow would be Login => Resources => Search Resources.

Figure 2:

This diagram offers a high-level overview of the technologies used in both the front-end server side.

Front End technologies:

- React.
- AlanAI.
- Material UI.
- Compromise
- HTML/SCSS/JavaScript.

Back End technologies:

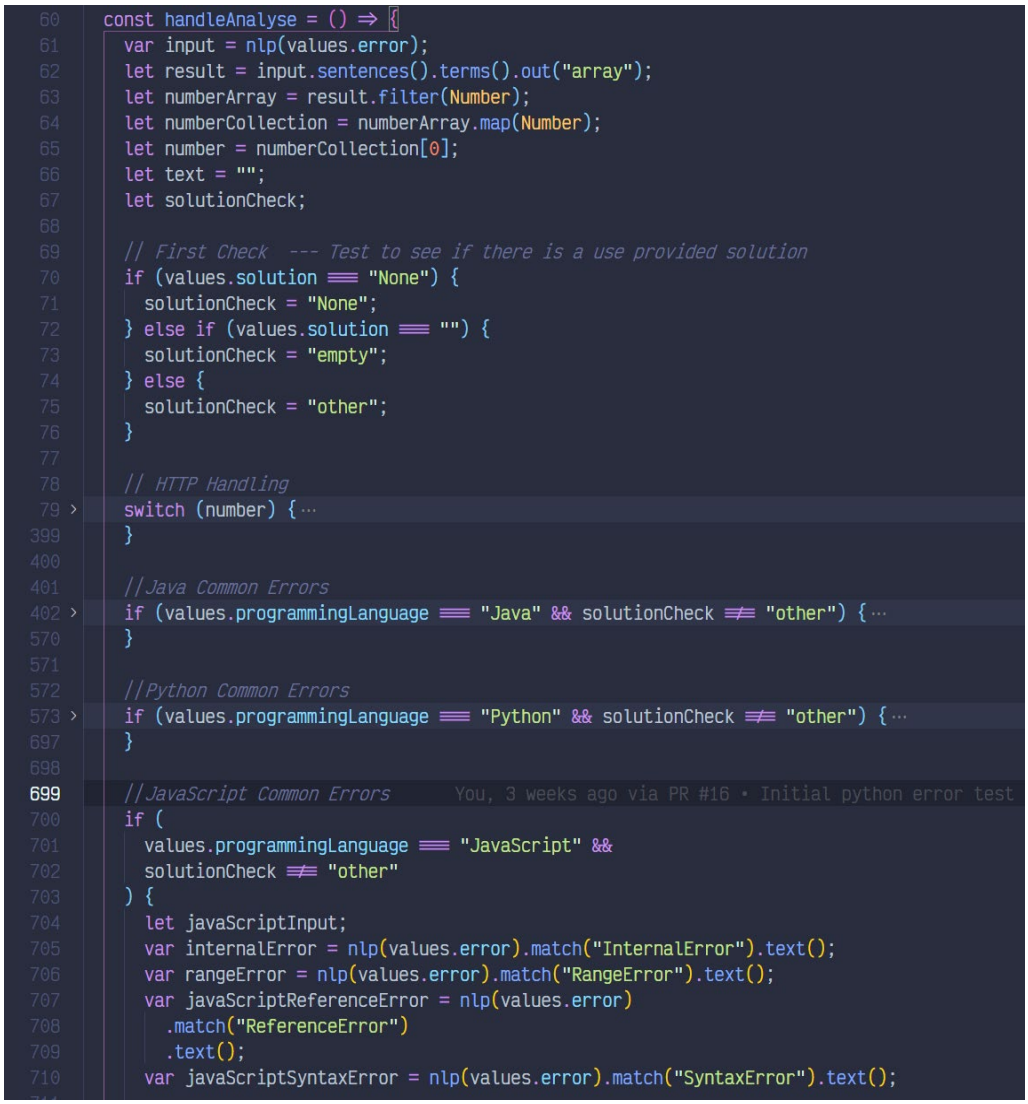
- Firebase.
- AlanAI.

The backend technologies include Firebase & AlanAI. My application will be deployed to the cloud through Netlify.

2.3. Implementation

2.3.1. Programming Journal - Natural Language Processing

The first key piece of functionality that I will walk through is the programming journal part of the project. The project consists of two components, one of which is the form that the user types into, this component is responsible for handling user input and passing it to the database. In this component the user input is also analysed in order to give a suggestion. The second component is responsible for displaying the users' previous submissions, and also handles updating and deleting the records. It is worth noting that the "nlp" method that I make use off in the code snippets provided come from the natural language processing library called "Compromise".



```
60  const handleAnalyse = () => {
61    var input = nlp(values.error);
62    let result = input.sentences().terms().out("array");
63    let numberArray = result.filter(Number);
64    let numberCollection = numberArray.map(Number);
65    let number = numberCollection[0];
66    let text = "";
67    let solutionCheck;
68
69    // First Check  --- Test to see if there is a use provided solution
70    if (values.solution === "None") {
71      solutionCheck = "None";
72    } else if (values.solution === "") {
73      solutionCheck = "empty";
74    } else {
75      solutionCheck = "other";
76    }
77
78    // HTTP Handling
79 ›  switch (number) {···
399   }
400
401   //Java Common Errors
402 ›  if (values.programmingLanguage === "Java" && solutionCheck !== "other") {···
570   }
571
572   //Python Common Errors
573 ›  if (values.programmingLanguage === "Python" && solutionCheck !== "other") {···
697   }
698
699   //JavaScript Common Errors      You, 3 weeks ago via PR #16 • Initial python error test
700   if (
701     values.programmingLanguage === "JavaScript" &&
702     solutionCheck !== "other"
703   ) {
704     let javaScriptInput;
705     var internalError = nlp(values.error).match("InternalError").text();
706     var rangeError = nlp(values.error).match("RangeError").text();
707     var javaScriptReferenceError = nlp(values.error)
708       .match("ReferenceError")
709       .text();
710     var javaScriptSyntaxError = nlp(values.error).match("SyntaxError").text();
```

Figure 4

The code above is a snippet of the function that is responsible for taking the user input and then giving a suggestion based off the input using natural language processing. The first piece of the function above is to take the user input and store it in an object that has the natural language processing methods in its prototype chain. This is done in line 61, where I make use of the "nlp" method. From this point I can then proceed to check if the values

provided are relevant, but first I need to check if the user has provided a solution or not. This is done on line 70 -> 76, from this point I then check to see what programming language the user has entered, then based on that I can perform a check on the input.

```
699   //JavaScript Common Errors
700   if (
701     values.programmingLanguage === "JavaScript" &&
702     solutionCheck !== "other"
703   ) {
704     let javaScriptInput;
705     var internalError = nlp(values.error).match("InternalError").text();
706     var rangeError = nlp(values.error).match("RangeError").text();
707     var javaScriptReferenceError = nlp(values.error)
708       .match("ReferenceError")
709       .text();
710     var javaScriptSyntaxError = nlp(values.error).match("SyntaxError").text();
711
712     //Checking for which error has occured
713     if (internalError === "InternalError") {
714       javaScriptInput = internalError;
715     } else if (rangeError === "RangeError") {
716       javaScriptInput = rangeError;
717     } else if (javaScriptReferenceError === "ReferenceError") {
718       javaScriptInput = javaScriptReferenceError;
719     } else {
720       javaScriptInput = javaScriptSyntaxError;
721     }
722
723     //Switch statement to set the tooltip
724     switch (javaScriptInput) {
725       default:
726         setTooltip(
727           "No suggestion available - Ensure any errors are in their original form provided by IDE/console."
728         );
729         break;
730       case "InternalError":
731         setTooltip(
732           "JavaScript Error: The internal error often occurs when something is too large, EG: too many switch cases, too much recursion, too many parantheses in regular expressior
733         );
734         break;
735       case "RangeError":
736         setTooltip(
737           "JavaScript Error: A Range Error is thrown when trying to pass a value as an argument to a function that does not allow a range that includes the value."
738         );
739         break;
740       case "ReferenceError":
741         setTooltip(
742           "JavaScript Error: A ReferenceError object represents an error when a non-existent variable is referenced, make sure to check your variable names. "
743         );
```

Figure 5

As can be seen in the code above, this is the part of the function responsible for dealing with JavaScript errors. The first step is to make a variable for each of the potential errors that the project supports, by using the .match() method and passing it as a string. The logic to the application is that a check is performed on which variable has become populated, when a populated variable is found, a separate variable called “javaScriptInput” is assigned to that value. The code for this logic is on line 713 => 721, this “javaScriptInput” variable is then passed as the argument to a switch statement. This switch statement checks for which error is stored in the argument, and then sets the state off the tooltip, which is what the user will read. The code snippet for this can be seen on line 724 => 743.

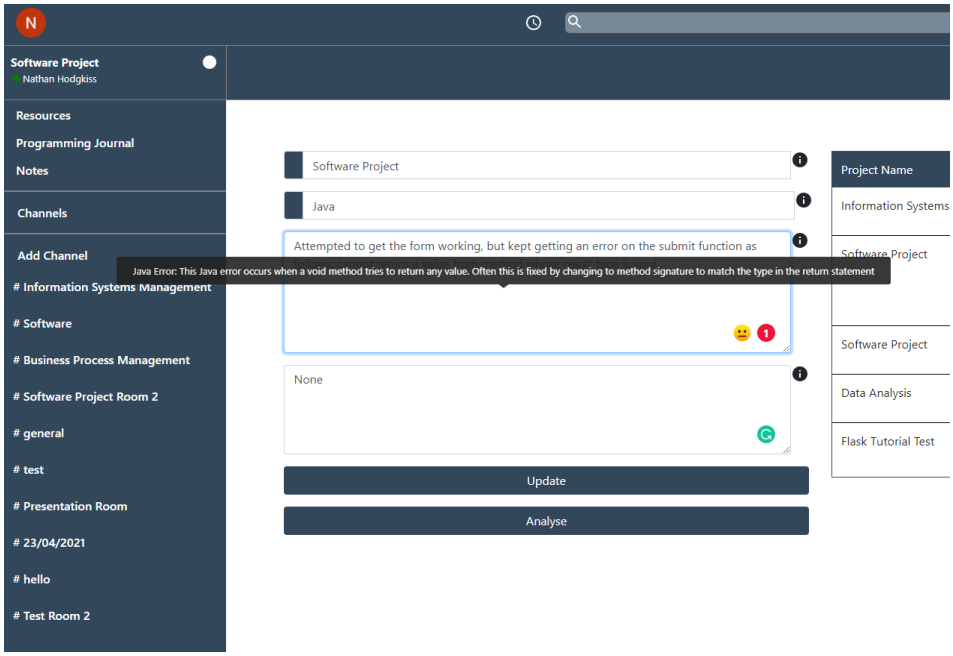


Figure 6

As can be seen in the screenshot above, the state off the tooltip is set in the switch statement, which then appears on screen when the user hovers their cursor over the “Error” text area.

2.3.2. AlanAI Voice Assistant

The second key piece of functionality that differentiates my application is the implementation of a voice assistant. This voice assistant has been programmed in both the client and the back end, which are connected through an API key. I will first walk through the back end which was programmed in the Alan Studios IDE.

```
// Questions
intent('What does this app do?', "What is this project?", reply('This is a smart programming journal. You can use it to track your programming progress and find useful resources.'));

intent("What can I do with this application?", "What can I do here?", reply("You can submit journal entries, write notes, look up books, and chat with your friends."))

intent('How do I use the programming journal', "How do I use the programming journal", reply("Enter your project name, programming language used, the error you encountered and your solution i

intent('How do I use the notes section', 'How do I create notes', reply("Click the notes section on the left side of the screen, and then click new note to start taking notes. You can also te

intent('How do I use the resource section', reply("Click on the resource section on the left side of the screen, then enter your required book in the search bar and then click enter or search

// Commands
intent("Bring me to the resource section", "Show me the resource section", "Bring me to the home screen", (p) => {
    p.play({command: 'navigationResource'});
});

intent("Bring me to the journal section", "Show me my journal", "Show me the journal section", (p) => {
    p.play({command: 'navigationJournal'});
});

intent("Bring me to the notes section", "Show me my notes", "Show me the notes section", (p) => {
    p.play({command: 'navigationNotes'});
});

intent("Bring me to the general chat", "Show me the general chat", (p) => {
    p.play({command: 'generalNavigation'});
});
```

Figure 7

As can be seen in the code above, it is separated into two parts, the first part handles the logic surrounding questions. The “intent” keyword is used, it first takes the user’s input, such as

“What does this app do”, after which a reply can be stated, such as “This is a smart programming journal. You can use it to track your programming progress and find useful resources”, on line 2. This logic is then repeated for different questions, can be seen on line 4, line 6, line 8, and line 10.

The second part of this code is the commands section, which contains four commands. The logic for each command is as follows, the intent keyword is used to capture the users input, then based on this a callback function is used, which makes use of the p.play() method that is built into AlanAI. Within the p.play() method a command is passed to the client. Based off this command, the client will then execute code.

Figure 8

The code above is the client-side code, this code is contained in a Use Effect, which is called once when the component App.js loads. The first piece of the code is “key: alanKey” on line 32, which is passing the AlanAI API key, this allows the client and the back end to communicate. The second part of the code is the method “onCommand”, which receives the command passed from the p.play() method as an argument. Based on this command, the code in the appropriate part of the If statement is executed, which is used to navigate the user to a particular section off the application.

2.4. Graphical User Interface (GUI)

CHAT Screen UI:

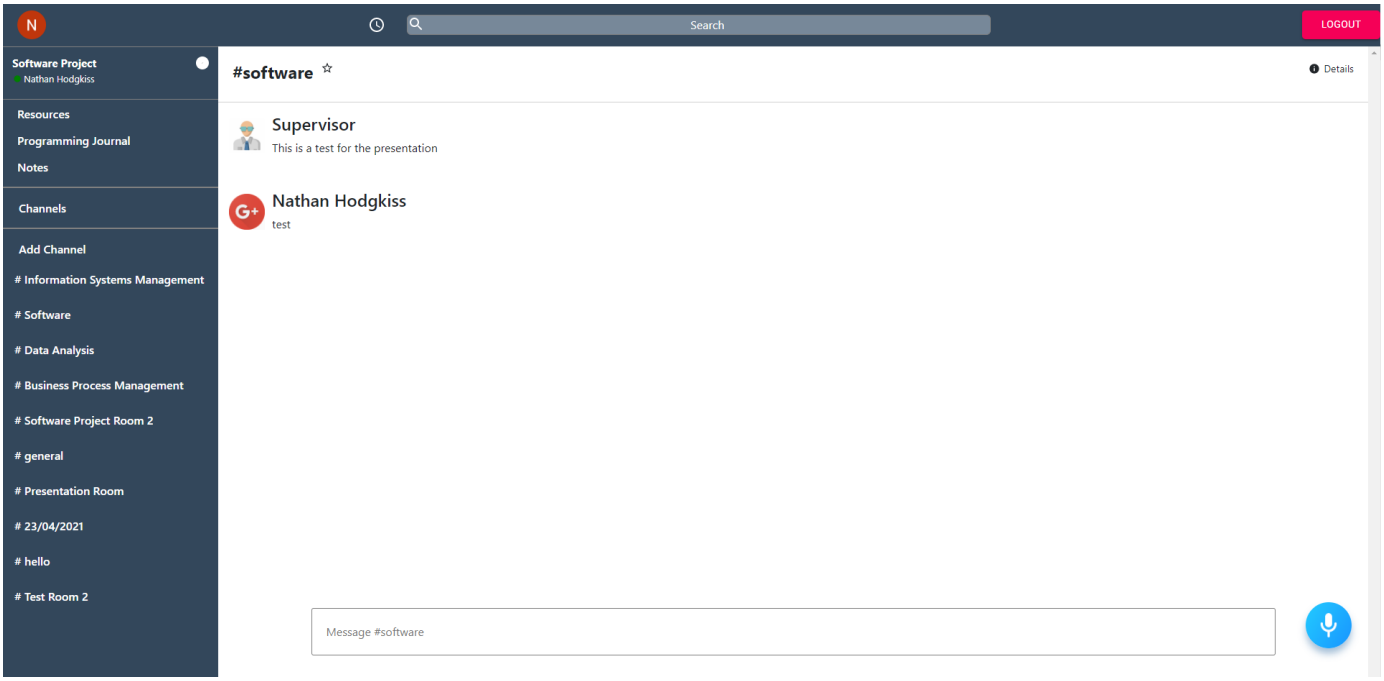


Figure 9

The UI above is an example of the UI for a chat room, this UI will be replicated for each of the chat rooms, with the channel name displayed at the top (#software) changing dynamically based on the room the user is in. In order for the user to access this UI, the user must select the room on the menu on the left side of the screen, all rooms are prefixed with '#'. Users can use the input box at the bottom of the screen in order to type messages in the chat. The messages will then appear in the chat room, along with the user's name and profile picture.

As can be seen on the left of the UI, the users can also click "Add Channel" in order to create a chat room, which will then appear on the left, prefixed with a "#".

RESOURCES UI:

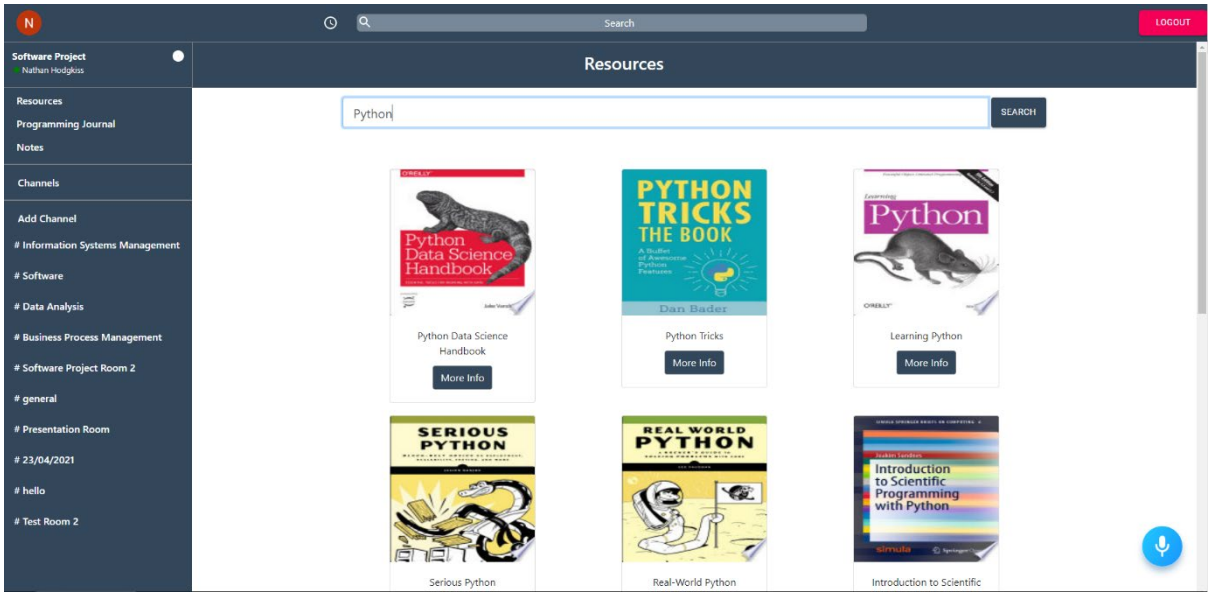
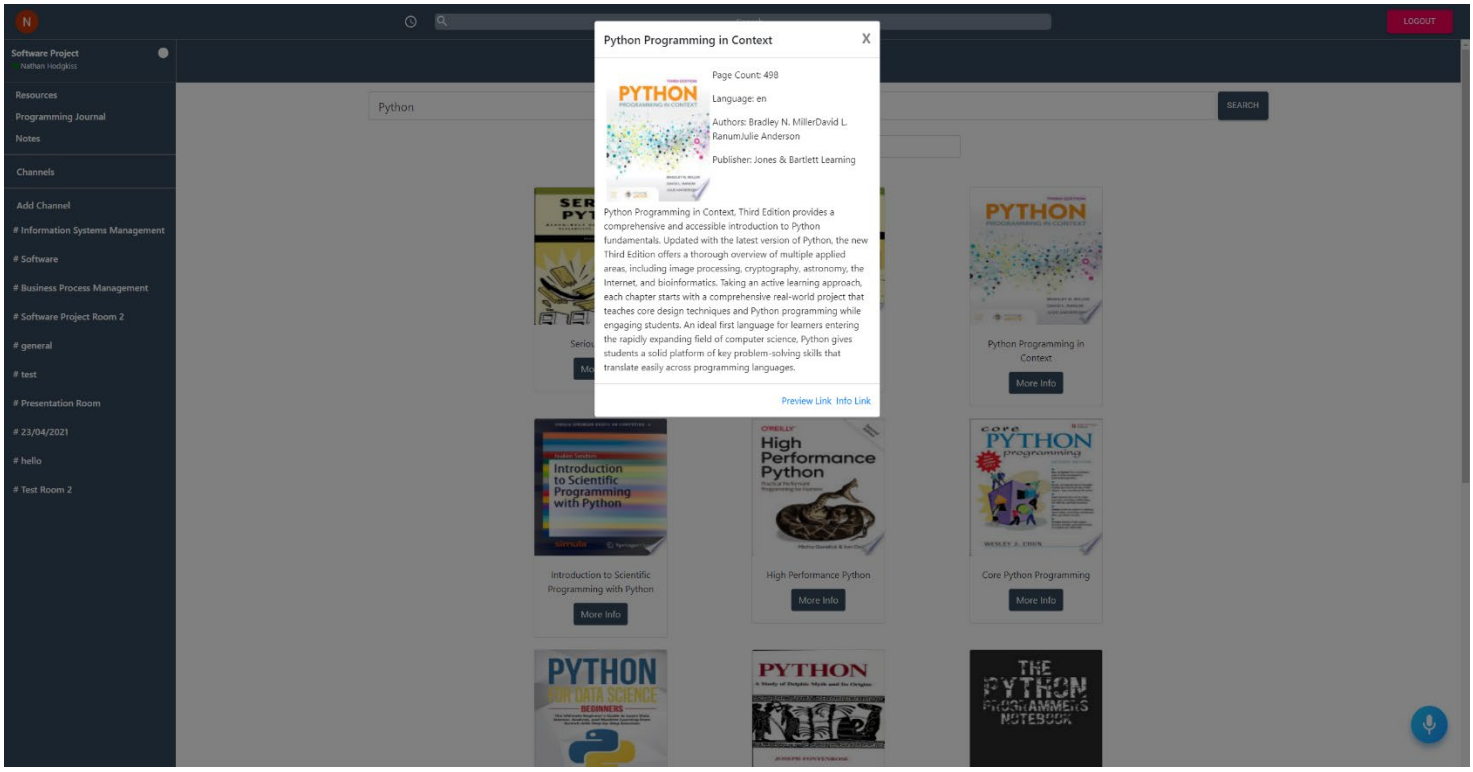


Figure 10

This is the resources section, this is where users can search for resources. The user uses the search bar at the centre top of the screen to type their required text, t hen cab either press enter on their keyboard or search to execute the function. A user can then select one of the books, and will then see a modal appear with information about the book, an example as follows:



The user can then click either link at the bottom of the modal to get access to the book.

PROGRAMMING JOURNAL UI:

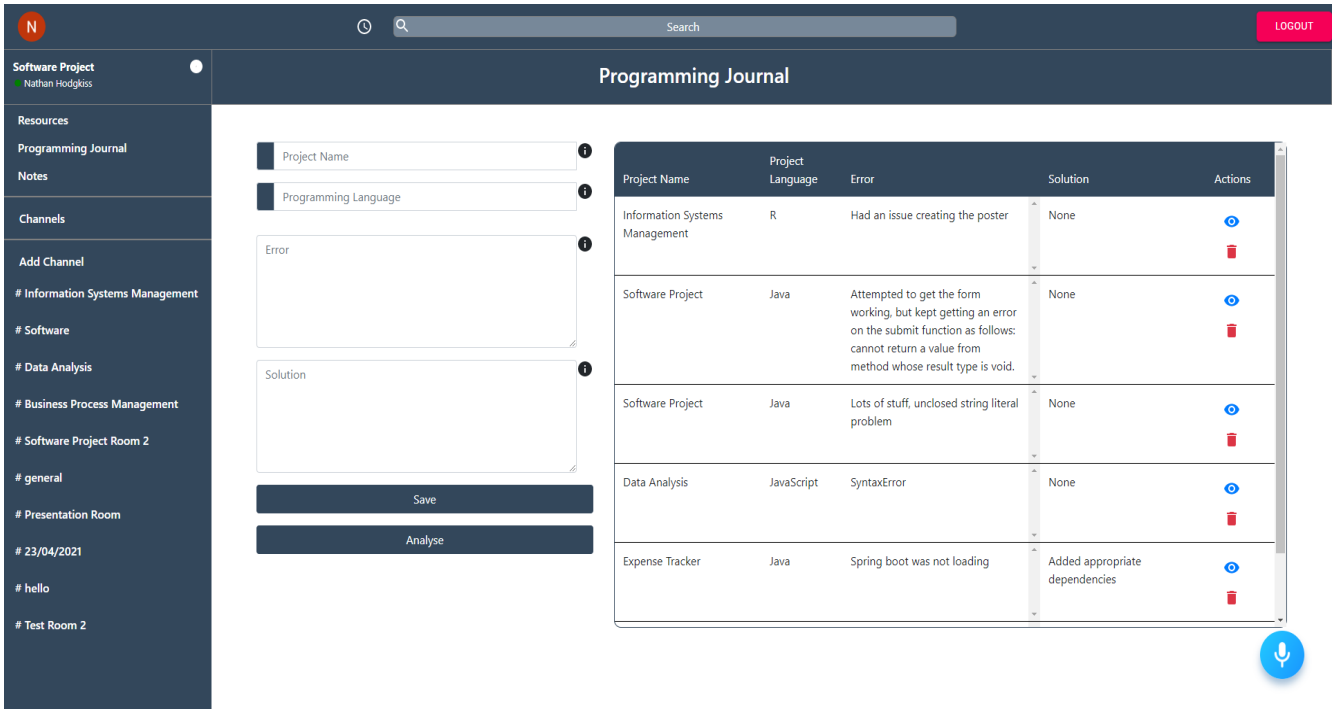


Figure 11

This is the programming journal section. To the left of the screen, you can see the input fields, this is how the user will submit a journal entry. Once the user clicks submit, the entry will be logged on the right. On the right you can see a list of the logged entries, along with the buttons that allows the user to edit or delete that specific entry. The edit buttons are the blue eyes, and the delete buttons are the red bins.

NOTES UI:

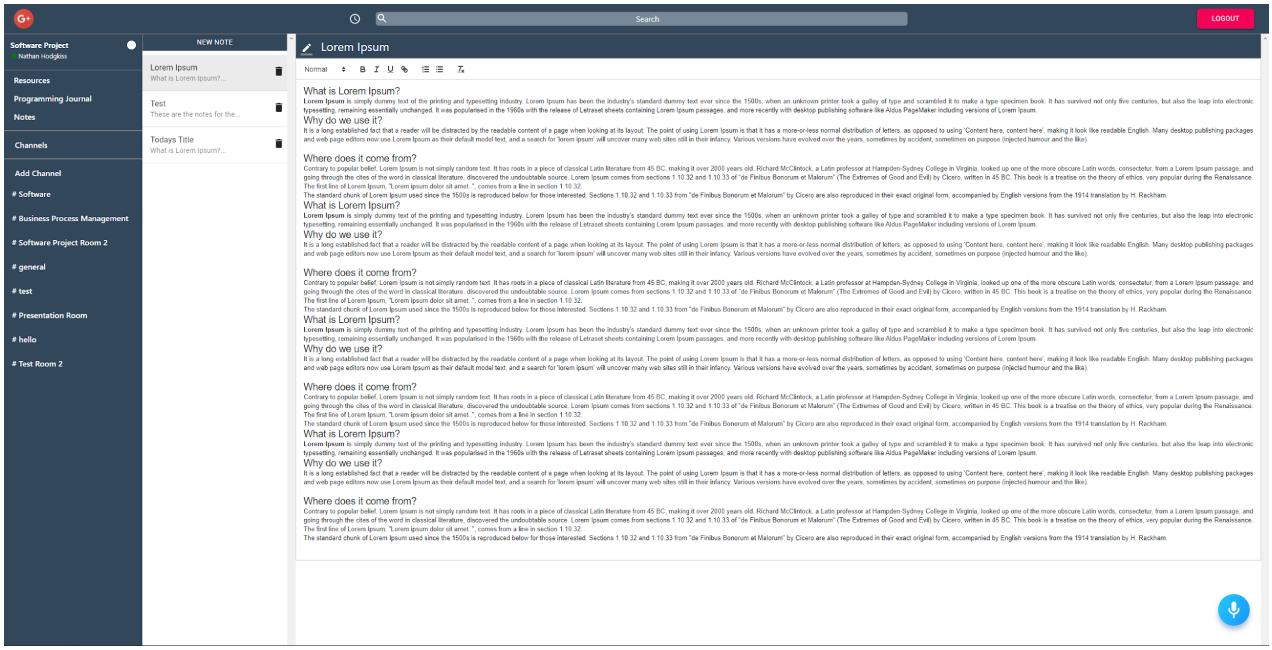


Figure 12

This is the UI of the notes section, on the left you can see the notes that currently exist along with the button to delete that note. On the right of the screen, you can see the content of the currently selected note, along with a toolbar for some basic editor functionality such as underline, bold etc.

2.5. Testing

Journal – Form Test – Unit Test

```
1  import * as React from "react";
2  import * as ReactDOM from "react-dom";
3  import renderer from "react-test-renderer";
4  import ContactForm from "../Components/ContactForm";
5  import { render, screen, cleanup } from "@testing-library/react";
6
7  test("Renders the correct journal form content", () => {
8    // Render the component to the DOM
9    render(<ContactForm />);
10   const form = screen.getByTestId("form-test");
11   const placeholderName = screen.getByTestId("placeholder-name");
12   const placeholderLanguage = screen.getByTestId("placeholder-language");
13   const placeholderError = screen.getByTestId("placeholder-error");
14   const placeholderSolution = screen.getByTestId("placeholder-solution");
15
16   //Testing if form exists
17   expect(form).toBeInTheDocument();
18
19   //Testing to check for appropriate placeholders
20   expect(placeholderName.placeholder).toBe("Project Name");
21   expect(placeholderLanguage.placeholder).toBe("Programming Language");
22   expect(placeholderError.placeholder).toBe("Error");
23   expect(placeholderSolution.placeholder).toBe("Solution");
24 });
25
26 //Snapshot Test
27 it("renders entire journal sections structure correctly", () => {
28   const journalForm = renderer.create(<ContactForm />).toJSON();
29   expect(journalForm).toMatchSnapshot();
30 });
```

Figure 13

The code above is the unit test for the form used for the journal functionality. First, I test that the form is rendered correctly.

As can be seen above, I then test that the placeholder text for the input fields is correct. This can be seen from line 19 to 24. Next as part of the unit test, I conduct a snapshot test, which compares the current DOM to the saved DOM, in order to make sure the component is structured correctly. This combination of tests within the component helps to ensure that the component will function correctly.

Journal – List Test - Unit Test

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { render, screen, cleanup } from "@testing-library/react";
import renderer from "react-test-renderer";
import Contacts from "../Components/Contacts";

test("Renders the correct journal content", () => {
  // Render the component to the DOM
  render(<Contacts />);
  const journal = screen.getByTestId("journal-test");
  const projectNameHeading = screen.getByTestId("heading-name");
  const programmingLanguageHeading = screen.getByTestId("heading-language");
  const errorHeading = screen.getByTestId("heading-error");
  const solutionHeading = screen.getByTestId("heading-solution");
  const actionsHeading = screen.getByTestId("heading-actions");

  expect(journal).toBeInTheDocument();
  expect(projectNameHeading).toHaveTextContent("Project Name");
  expect(programmingLanguageHeading).toHaveTextContent("Programming Language");
  expect(errorHeading).toHaveTextContent("Error");
  expect(solutionHeading).toHaveTextContent("Solution");
  expect(actionsHeading).toHaveTextContent("Actions");
});

//Snapshot Test
it("renders entire journal sections structure correctly", () => {
  const contacts = renderer.create(<Contacts />).toJSON();
  expect(contacts).toMatchSnapshot();
});
```

Figure 14

The code above is the unit test for the list of journal entries, a part of the journal functionality. First, I grab all the appropriate classes and store them in variables. Next, I make sure that all the headings contain the correct text. I then conduct a snapshot test in order to ensure that the current DOM is matching the saved DOM structure for this component. This combination of tests within the component helps to ensure that the component will function correctly.

Resources – Test - Unit Test

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { render, screen, cleanup } from "@testing-library/react";
import renderer from "react-test-renderer";

import Resources from "../Components/Resources";

//Rendering Test
test("Renders the correct resources content", () => {
  render(<Resources />);

  const search = screen.getByTestId("search");
  const searchInput = screen.getByTestId("search-input");
  const searchButton = screen.getByTestId("search-button");
  const searchInputBar = screen.getByTestId("search-input-bar");

  expect(search).toBeInTheDocument();
  expect(searchInput).toBeInTheDocument();
  expect(searchButton).toBeInTheDocument();

  expect(search).toHaveTextContent("Resources");
  expect(searchInputBar.placeholder).toBe("Search");
});

//Snapshot Test
it("DOM is structured correctly", () => {
  const resources = renderer.create(<Resources />).toJSON();
  expect(resources).toMatchSnapshot();
});
```

Figure 15

The code above is the unit test for the resource component. As can be seen above first I capture all the relevant DOM elements and then store them in a variable. Next, I make sure that these elements exist in the component. I then ensure that the heading for the component is set correctly, I then ensure that the placeholder text for the input is correct. Next a snapshot is conducted in order to make sure that the component has the correct DOM structure.

Notes – Test - Unit Test

```
You, a day ago | 1 author (You)
import * as React from "react";
import * as ReactDOM from "react-dom";
import { render, screen, cleanup } from "@testing-library/react";
import renderer from "react-test-renderer";

import EditorNote from "../Components/Editor/EditorNote";
import Note from "../Components/Note";
import SidebarItem from "../Components/SidebarItem/SidebarItem";
import SidebarNotes from "../Components/SidebarNotes/SidebarNotes";

it("renders editor correctly", () => {
  const editor = renderer.create(<Note />).toJSON();
  expect(editor).toMatchSnapshot();
});
```

Figure 16

The code above is the unit test for the note's component. As most of the content generated within this component is dynamic, testing for hard-coded values is not appropriate, hence a basic snapshot test is conducted. Further functionality for this component will be tested with Cypress, in order to ensure that it functions as expected.

Cypress – Testing Chat Functionality:

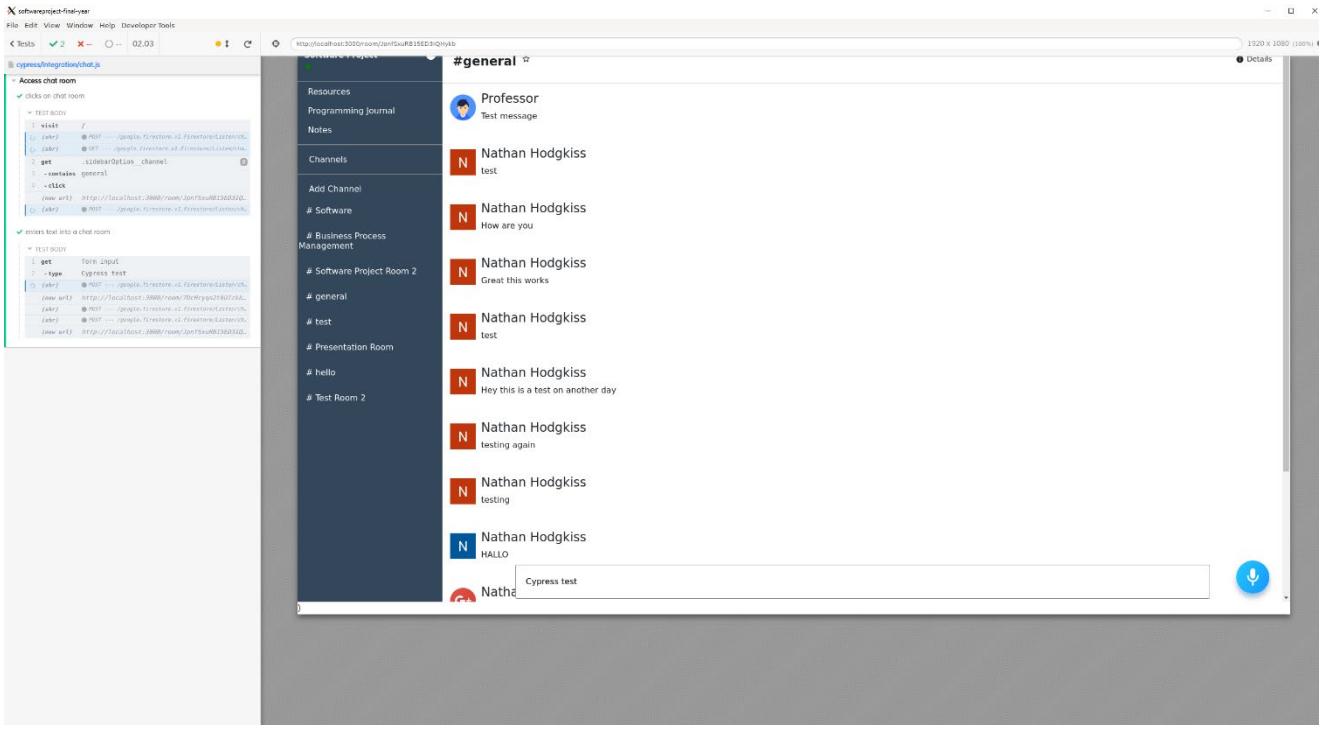


Figure 17

The screenshot above is the Cypress runtime for testing the chat functionality. Cypress runs in a browser and executes the commands in real time. Each command runs procedurally, as can be seen above, Cypress successfully navigated to the general chat, and then typed a message in the chat box.

Cypress – Testing Notes Functionality:

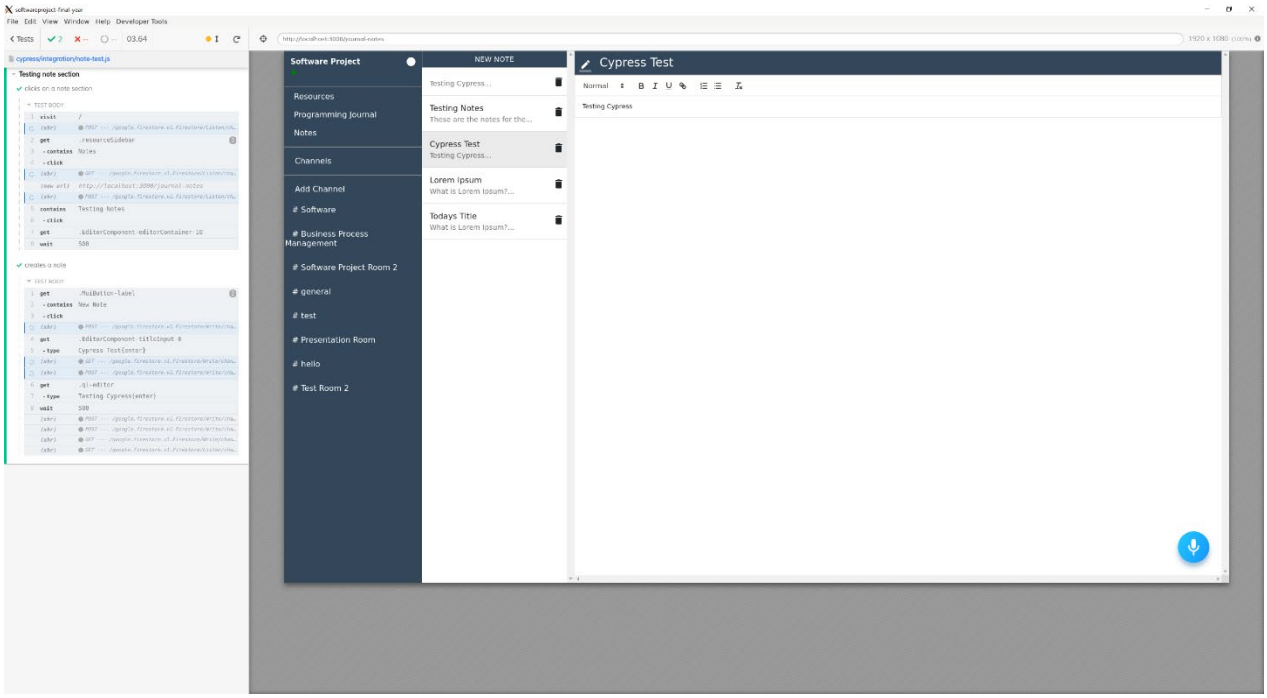


Figure 18

The screenshot above is the Cypress runtime for testing the note functionality. Cypress runs in a browser and executes the commands in real time. Each command runs procedurally, as can be seen above, Cypress successfully navigated to the notes section, and then types out a basic note title and note.

Cypress – Testing Resources:

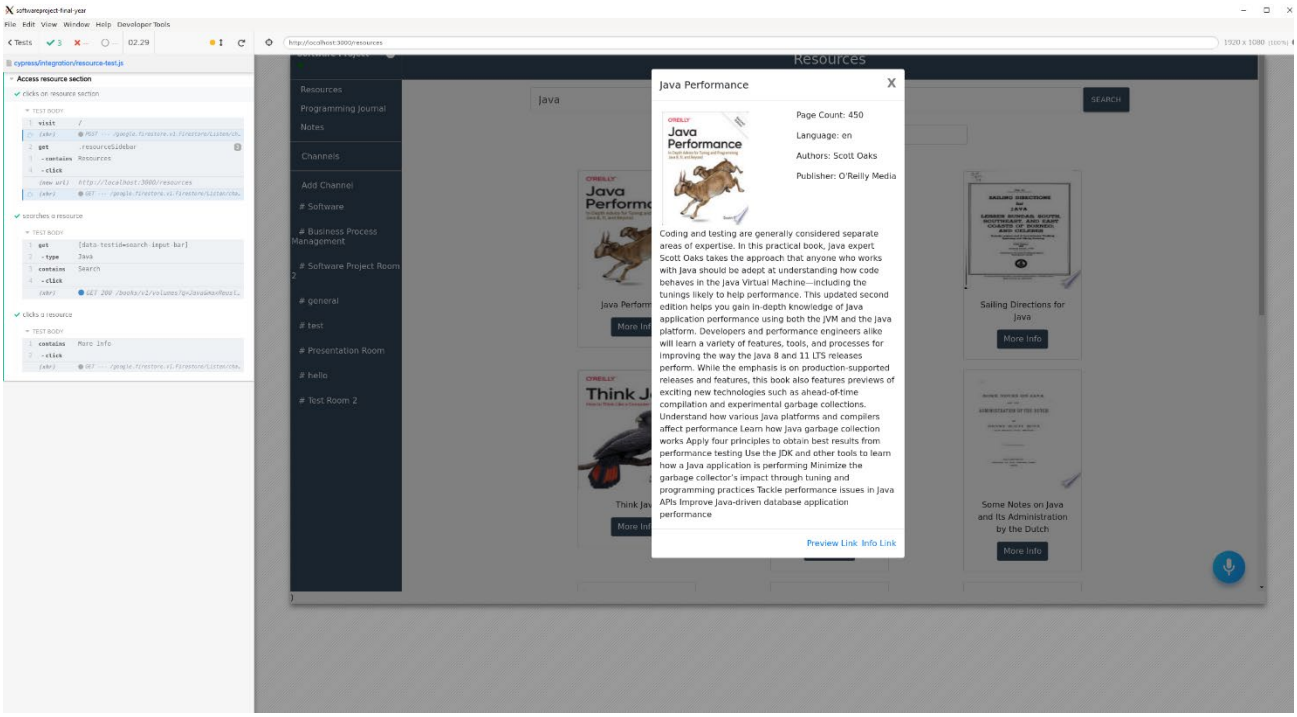
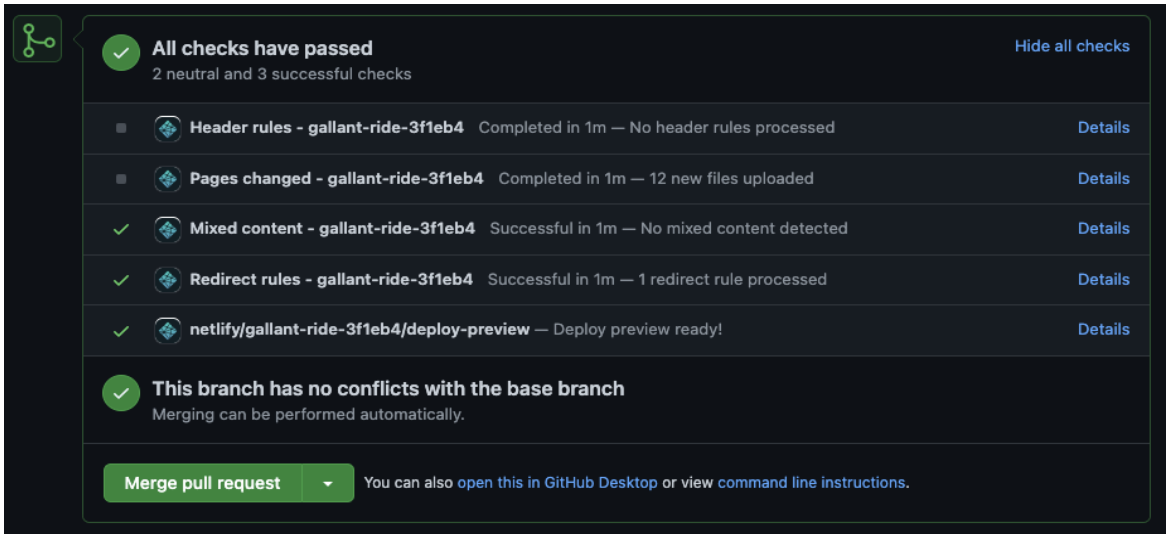


Figure 19

The screenshot above is the Cypress runtime for testing the resource functionality. Cypress runs in a browser and executes the commands in real time. Each command runs procedurally, as can be seen above, Cypress successfully navigated to the resource section, then typed a message in the search bar and then searched. Cypress then clicked on one off the books to make the modal in the screenshot above appear.

GitHub Hooks - Netlify

As part of my quality testing, when I deployed my site I setup GitHub hooks through Netlify, which allowed me to run extra checks whenever I tried to merge any changes into my master branch, as the example below shows:

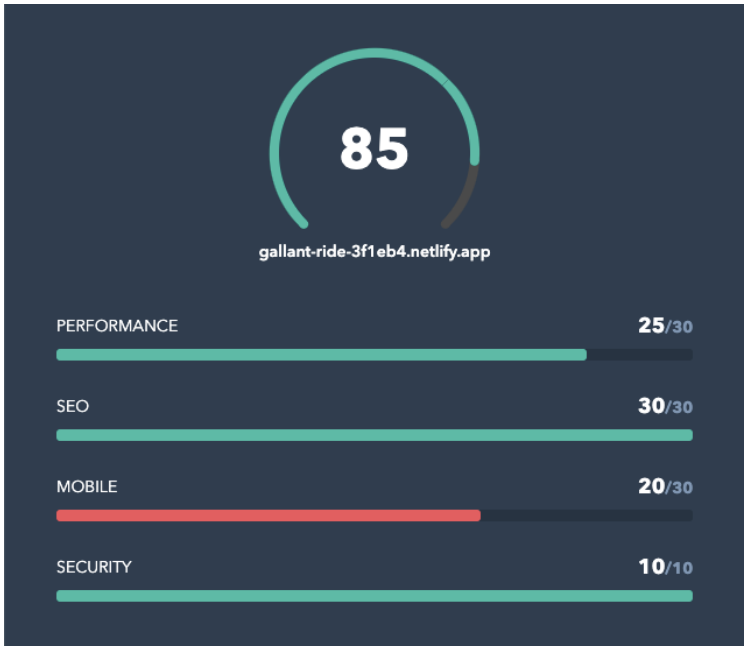


This ensures that the application would continue to run without any issues, which became important later into the development cycle as deadlines became closer and any technical issues could have major ramifications.

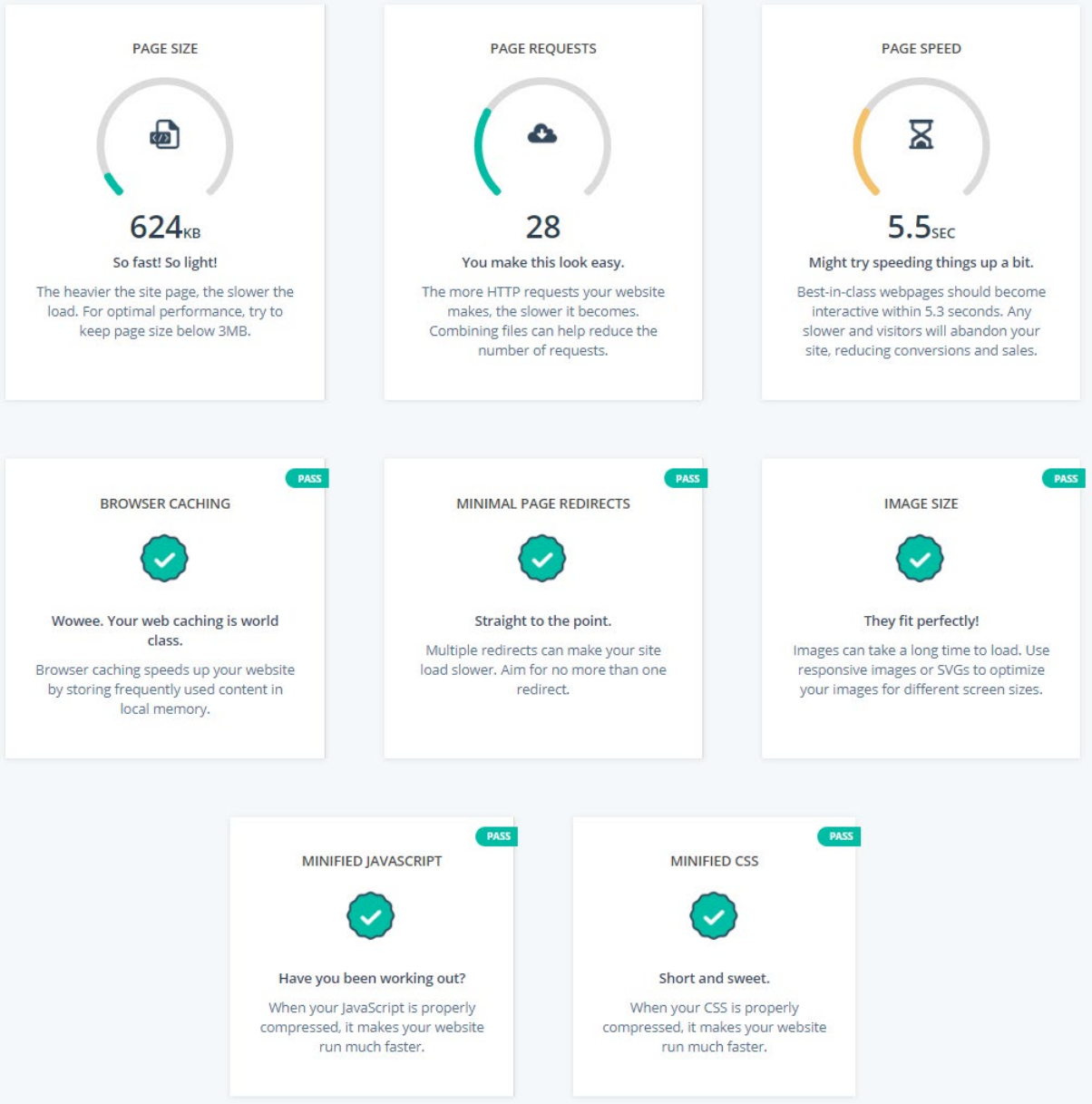
2.6. Evaluation

Site Performance - HubSpot Website Grader

As part of my non-functional testing, I used HubSpot's Website Grader which is powered by Google Lighthouse in order to figure out where my site was losing performance or if there were any security issues (HubSpot, 2021). When these issues were found I then fixed them, these included preloading essential assets, removing unused JavaScript etc. This then resulted in excellence performance, SEO, and security. The review of the website was broken down into four major categories, performance, SEO, mobile, and security.



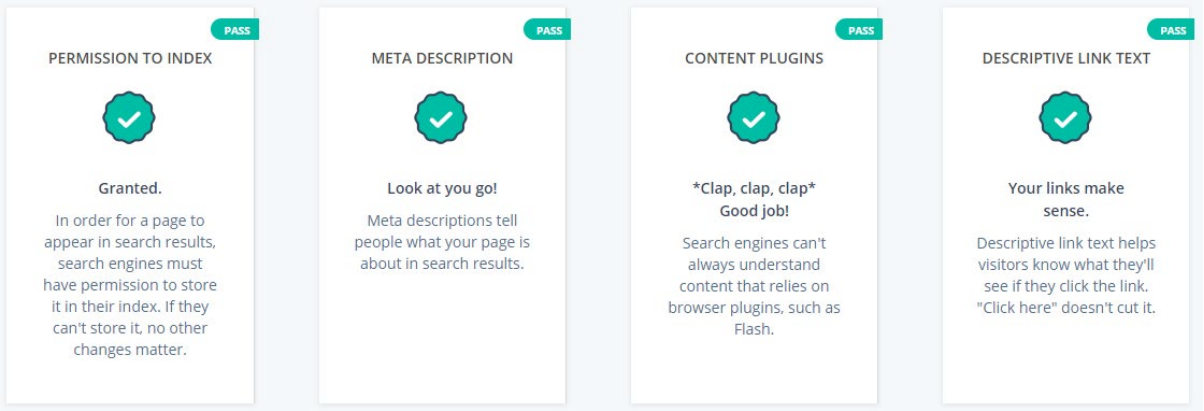
Performance:



Performance was off great importance to this project, in order to provide a great user experience. As can be seen above, the project employs the following methods to ensure maximum performance:

- Low page size
- Minimal page requests.
- Fast page speed.
- Browser caching.
- Minimal page redirects (Single page web application).
- Responsive images.
- Minified CSS.
- Minified JavaScript.

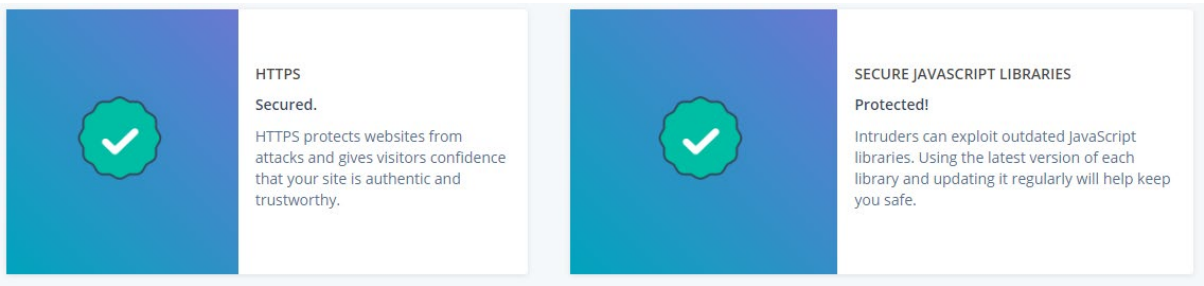
SEO:



Search Engine Optimisation, although not a major concern for this project, is a biproduct off following best practices and excellent performance. The application has excellent SEO performance, as shown above. The application passed the following checks:

- Permission to index.
- Appropriate meta description.
- Content plugins.
- Descriptive text on all links.

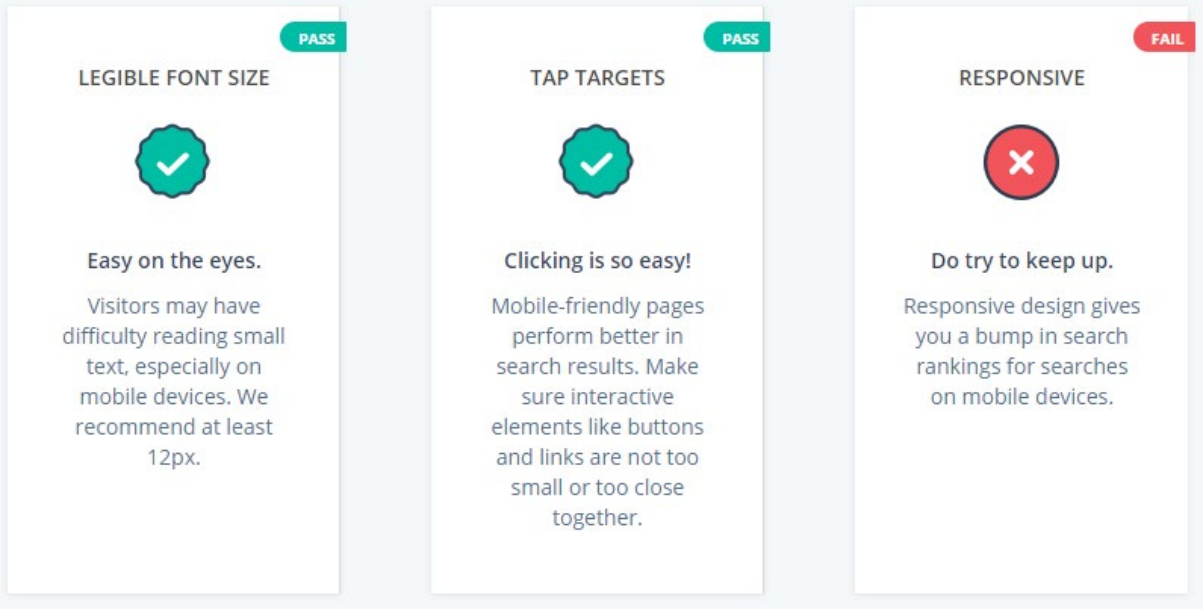
Security:



Security was off paramount importance to this project, in order to ensure that all user details are stored correctly and that the source code is protected. The project passed the following checks:

- My application is deployed and secured with HTTPS.
- Secure JavaScript Libraries, I used the latest version of Node and many other libraries to ensure that there would be no outdated dependencies which could result in security issues.

Mobile:



Mobile optimisation was not a major concern for this project, as it was not in the scope of the project to design all off the functionality to be responsive, due to the nature off the project being a single page web application. The following checks were for the mobile performance of my project:

- My application has legible font size.
- The application has appropriate tap targets.
- My application is not responsive, as it was not in scope for this version.

3.0 Conclusions

For the review, the project that I submitted contained the following functionality:

- Resource section: Users can go to the search bar and search for boo0ks
- Notes Section: Users can use the note taking functionality
- Programming Journal: Users can create journal entries, which if appropriate, the system can then give suggestions on.
- Chat: The user can create and type in chat rooms.
- Voice Assistant: The users can interact with an AI voice assistant, which provides details about the application, allows the user to navigate, and has other useful functionality such as acting as a calculator

Each of the functionality stated above were included in my original project proposal, with the added addition off the notes taking section, which I felt added value to the application. There are both strengths and weaknesses to my application. The first strength off the application is it has completed what it was set out to achieve in the project proposal, therefore the application has a high level of completeness. The second strength to the application is that it has bundled unique functionality that does not exist in other applications that I am aware off, including the voice assistant and the analysis of errors in the user's journal entries. The third strength of the application is that it has excellent performance, as it was designed using best practices which results in a better user experience. The fourth strength of the application is that it has a high level of security, which is paramount as users are required to provide their email address in order to access the application.

The application also has some weaknesses in its current iteration, with the first weakness being that it is not mobile ready. The application does not have a fully responsive design as it is a single page web application and would have been out of scope in order to make it fully responsive for smaller screens. The second weakness off the application is that the resource section is currently limited to just books, which may or may not be off use to all users. The third weakness to my application is that the analyses off users' errors is currently limited to Java, JavaScript, Python and HTTP error codes.

4.0 Further Development or Research

There are a couple of areas where this application could benefit from further development, the first functionality that I would improve is the login, as at the moment the user is required to login through Gmail, which although not a huge problem could limit the number of users the application receives. The second functionality that I would continue to build on, would be the programming journal, by adding support for more languages and error messages, with potentially storing a collection of error codes in an API, and then making calls to that API for results, instead of hardcoding the errors on the client side. The third area that I would like to develop further would be the resources section, by adding support for YouTube videos and other video resources. Finally, I would also put a high priority on making my current application responsive in order for it to be usable on smaller devices such as mobile phones.

5.0 References

- GitHub. 2021. *spencermountain/compromise*. [online] Available at: <https://github.com/spencermountain/compromise> [Accessed 9 May 2021].

- HubSpot, I., 2021. *Website Grader*. [online] Website.grader.com. Available at: <https://website.grader.com/> [Accessed 9 May 2021].

- Firebase. 2021. *Documentation | Firebase*. [online] Available at: <https://firebase.google.com/docs> [Accessed 9 May 2021].

- Material-ui.com. 2021. *Material-UI*. [online] Available at: <https://material-ui.com/getting-started/installation/> [Accessed 9 May 2021].

- Reactjs.org. 2021. *Getting Started – React*. [online] Available at: <https://reactjs.org/docs/getting-started.html#learn-react> [Accessed 9 May 2021].

6.0 Appendices

6.1. Project Plan

ID	Task Mode	Task Name	Duration	Start	Finish	Predecessors	Resource Names
1		**Research/Prelimary-Development**	**35 days**	**Mon 28/09/20**	**Sun 15/11/20**		
2		Project Pitch	15 days	Mon 28/09/20	Sun 18/10/20		
3		Project Proposal + Ethics Form	15 days	Mon 19/10/20	Sun 08/11/20	2	
4		Technical Research	6 days	Mon 09/11/20	Sun 15/11/20	3	
5		**Mid-point Development**	**29 days**	**Mon 16/11/20**	**Tue 22/12/20**		
6		Wireframes	3 days	Mon 16/11/20	Wed 18/11/20		
7		Front End Development	23 days	Thu 19/11/20	Fri 18/12/20	6	
8		Preliminary Back End Development	23 days	Thu 19/11/20	Fri 18/12/20		
9		Midpoint Documentation (Ongoing)	26 days	Mon 16/11/20	Fri 18/12/20		
10		Video Presenatation	3 days	Sat 19/12/20	Tue 22/12/20	7	
11		**Final Development**	**95 days**	**Mon 04/01/21**	**Sun 16/05/21**		
12		Front End Development	44 days	Mon 04/01/21	Thu 04/03/21		
13		Back End Development	37 days	Fri 05/03/21	Sun 25/04/21		
14		Final Documentation (Ongoing)	91 days	Mon 04/01/21	Sat 08/05/21		
15		Video Presentation	7 days	Sat 08/05/21	Sun 16/05/21		

Project: Simple project plan.mp
Date: Mon 02/11/20

Task	Inactive Summary	External Tasks
Split	Manual Task	External Milestone
Milestone	Duration-only	Deadline
Summary	Manual Summary Rollup	Progress
Project Summary	Manual Summary	Manual Progress
Inactive Task	Start-only	
Inactive Milestone	Finish-only	

Page 1

National College of Ireland

Project Proposal

Title

25/10/2020

Business Information Systems

2020/2021

Nathan Hodgkiss

x17381176

x17381176@Student.ncirl.ie

Contents

1.0 Objectives 2

2.0 Background 3

3.0 Technical Approach 5

4.0 Project Plan 6

5.0 Technical Details 7

6.0 Evaluation 7

7.0 Objectives

The objective for this software project is to create a high-quality software artefact for beginners learning to code, that allows users to send text messages in 'chat rooms', that they have control over, while providing a 'resource hub' for information relating to learning to code. The idea is to allow users to access information based off filter or selection. For example, if someone is interested in learning Python, they can select an image of the Python symbol, or a dropdown etc, that would then provide the user with the appropriate material. This will help to define my project from other pieces of software that are available on the market. My strategy was to focus on a very specific use case for the application and then build features around that use case, therefore differentiating it from others as much as possible.

Another aspect of this project will be the voice assistant feature, that will allow users to control parts of the applications with their voice. I believe this could contribute to a better user experience for most users, but more importantly I think it is an important accessibility feature. This project will also include a login/authentication system.

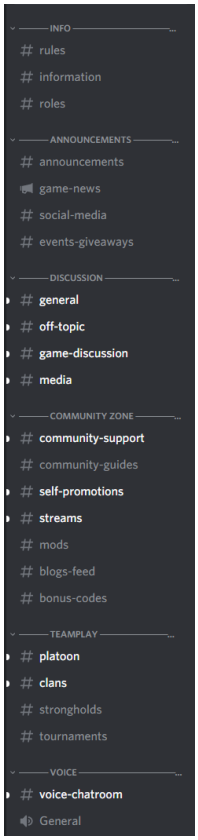
This project has a clearly defined target audience; users who are beginning to learn to code. This project is not targeted at advanced programmers or as a tool for advanced programmers, though the chat aspect could still easily be used by advanced programmers. As such, when I am developing the features for this application, or when I am brainstorming requirements for this application, this will be my target audience in mind.

Other objectives for this software project are to successfully learn and implement new technologies that I have not previously worked with, including React, Redux, AlanAI etc. The primary reason for wanting to learn and successfully implement these technologies as I am targeting a career in software development post-graduation. During my internship I got hands on with enterprise level applications that used React, allowing me to glimpse the potential of the framework, in particular the programmatical nature in which state can be manipulated interested me greatly.

8.0 Background

There are two other major chat applications on the market as it stands, these being Discord and Slack. Discord is an application that allows users to create a 'Server', where they can create both voice communication rooms and chat rooms, where users can communicate through text. The application also allows users to upload custom emotes, use bots made by other users and allows for a host of options to be configured by users. The general use case for Discord is often for gamers who want to have a central place for a large community or just a group of friends to communicate and interact. Discord is not intended for one use case, rather it allows the users to decide how they want to use the software, it is available for both PC and mobile. Discord is an application that I use every day and works flawlessly. I intend to take inspiration from its minimal but clean design. Some of the features that Discord has implemented will just not be realistic for my application in version one, such as an extensive user permission configuration per server.

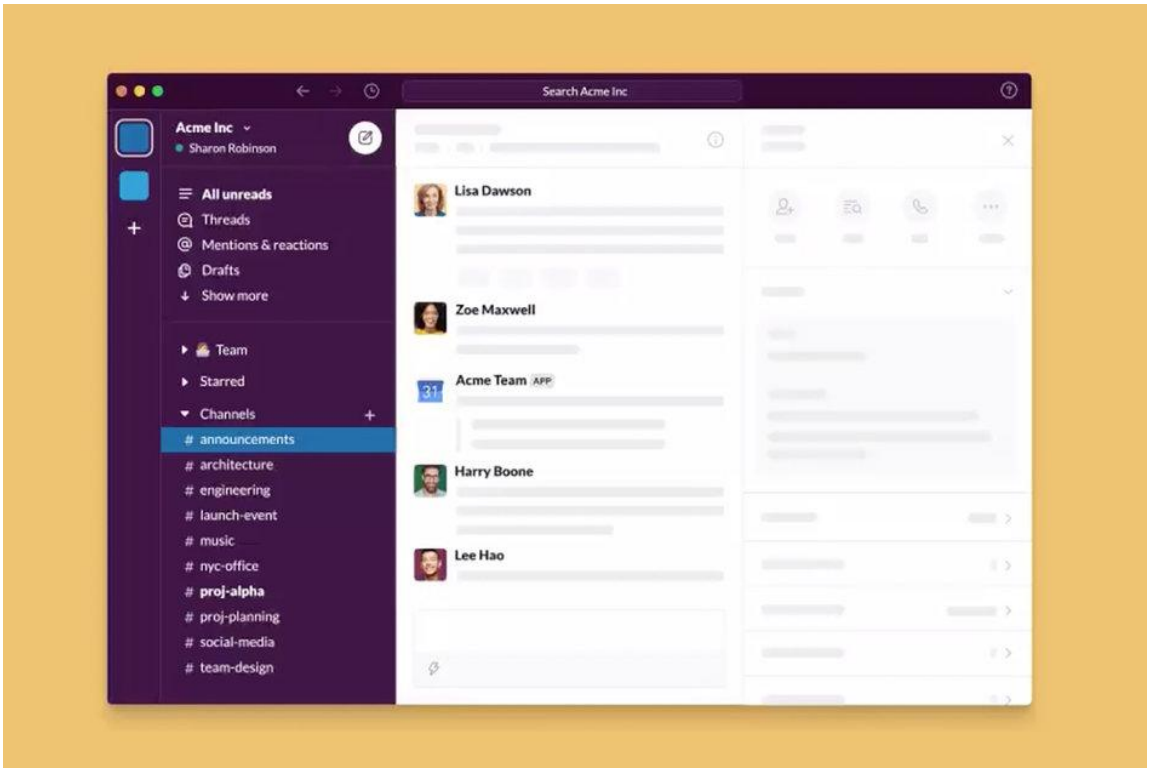
Discord UI Screenshot:



Overall, Discord provides users a host of options for users to configure, but my application will be intended to be used for a more specific use case – learning to code – will allow me to build in more targeted features, for example, the 'resource hub', the 'voice assistant' that will differentiate my application from the likes of Discord. I am also considering attempting to integrate in some small helpful personal/project management tools, for example a to do list etc.

Slack is a piece of software for business communication, that allows an organisations' employees to talk and discuss at scale. This was a piece of software that I had not previously had experience with until recently, where I used it in a large multi-national organisation. I was impressed with the software's ability to scale with a large user base and the ease at which users could interact with each other. Slack differs from Discord in the fact that Discord gives equal importance to chat channels and voice channels, whereas Slack's implementation is more focused on the chat aspect, while still offering the ability to create voice calls, in the more traditional sense, similar to Skype. One of the major advantages of Slack is its ability to integrate with other professional tools to help with a user's workflow. An example of this that I have used and seen was used for logging as part of a software development team. A software called 'Sentry', which is an error tracking tool, can be integrated into slack, to provide real-time notifications into a configured channel with regards to any errors, all of which can be configured by the user. This is integration aspect is a major competitive advantage for Slack in the enterprise market.

Slack UI:



Similar to Discord, Slack offers a clean and minimalistic UI design, that makes navigating the application very intuitive. Slack has a slightly more targeted audience than Discord, as it is very clearly aimed at professionals who want a piece of software that allows for communication and helps to refine certain user's workflow. My application will take inspiration from Slack but will differentiate itself through it's more targeted audience, Resource Hub and its voice assistant feature, which should be a major accessibility feature. Another aspect that I am considering implementing into my application is an RSS feed from various coding blogs.

9.0 Technical Approach

This project will require extensive research during development, as I am working with unfamiliar technology. There are some questions to be answered, such as how I instance the chat rooms, what exactly is my resource hub going to look like, and what database am I intending to implement. After this proposal is submitted, I intend to spend two weeks researching certain technical aspects prior to even touching a piece of code. The main technical question that needs to be answered is how I instance chat rooms based on user accounts etc. The technical approach that I am intending on following is as follows:

- Pre-development research
- Technical Report
- Wireframing
- Front End Development
- Back End Development
- Unit testing/End to End (Iterative/Ongoing)

As mentioned, my pre-development research will consist of finding the answers to certain technical questions, but also getting a better idea of the user requirements for the front end. After I have gathered the requirements and have a better hold of certain technical aspects, I intend to mock-up wireframes of the main interface stages that I have off my application. This will then act as the basis of design when I start into front end development. I am considering using Storybook for developing parts of my components, but I am still undecided on that point.

Beginning development, I will setup a repository on GitHub, where all documentation and code relating to my project will be stored. I gained valuable insight into how to properly use GitHub from my internship and intend on getting the full use out of it for my project. My IDE is Visual Studio, which I have configured to my liking, and has excellent integration with GitHub. For visual components, I am liking going to use multiple libraries, such as material UI, but I am considering Storybook, but that will require further research. For testing the front end of my application, I will likely use Jest, as it works excellently with React.

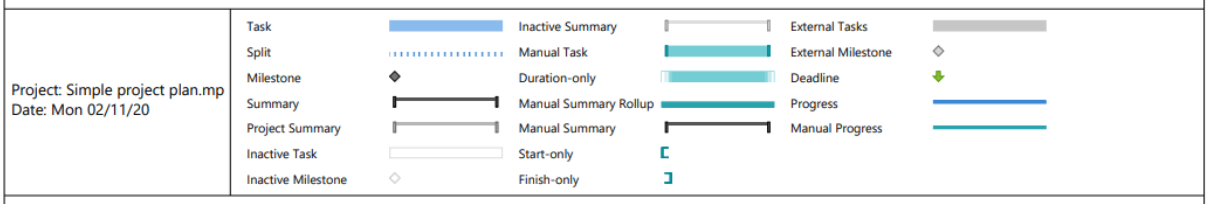
The back-end aspect off my application I am not as sure on, but as it stands, I am intending on using Firebase for most of it. Firebase will allow me to deploy, host and version my application easily, with their intended firebase cli tool. Firebase also includes two databases, FireStore or Realtime. I will likely be using FireStore for this project, though I am going to research other options such as MongoDB. For hosting I have also used Netlify before, with good success.

Firebase also offers a host off extra utilities such as monitoring, logging and analytics, which could also prove very useful for my project.

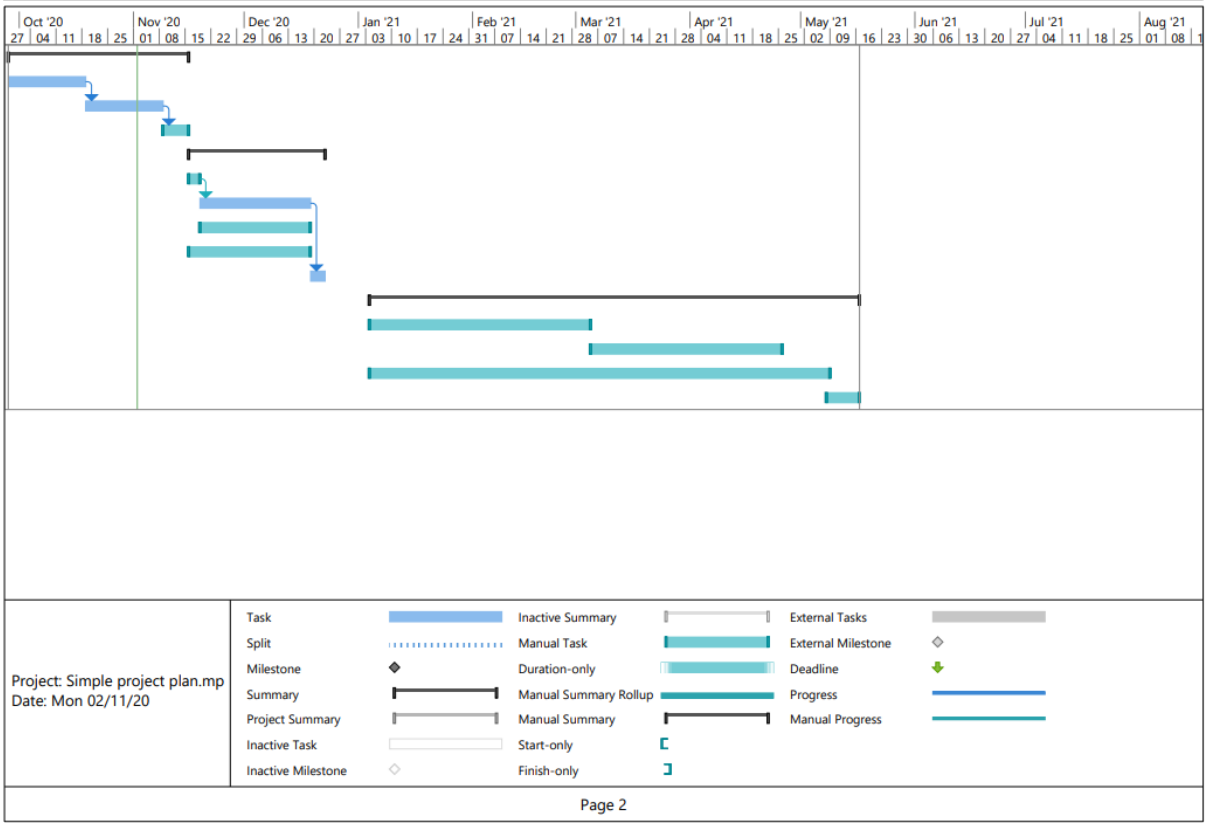
10.0 Project Plan

Generic plan detailing rough outline of timelines - will update as I get a more detailed idea of project.

ID	Task Mode	Task Name	Duration	Start	Finish	Predecessors	Resource Names
1		**Research/Prelimary-Development**	**35 days**	**Mon 28/09/20**	**Sun 15/11/20**		
2		Project Pitch	15 days	Mon 28/09/20	Sun 18/10/20		
3		Project Proposal + Ethics Form	15 days	Mon 19/10/20	Sun 08/11/20	2	
4		Technical Research	6 days	Mon 09/11/20	Sun 15/11/20	3	
5		**Mid-point Development**	**29 days**	**Mon 16/11/20**	**Tue 22/12/20**		
6		Wireframes	3 days	Mon 16/11/20	Wed 18/11/20		
7		Front End Development	23 days	Thu 19/11/20	Fri 18/12/20	6	
8		Preliminary Back End Development	23 days	Thu 19/11/20	Fri 18/12/20		
9		Midpoint Documentation (Ongoing)	26 days	Mon 16/11/20	Fri 18/12/20		
10		Video Presenatation	3 days	Sat 19/12/20	Tue 22/12/20	7	
11		**Final Development**	**95 days**	**Mon 04/01/21**	**Sun 16/05/21**		
12		Front End Development	44 days	Mon 04/01/21	Thu 04/03/21		
13		Back End Development	37 days	Fri 05/03/21	Sun 25/04/21		
14		Final Documentation (Ongoing)	91 days	Mon 04/01/21	Sat 08/05/21		
15		Video Presentation	7 days	Sat 08/05/21	Sun 16/05/21		



Page 1



Page 2

11.0 Technical Details

Implementation Language: JavaScript

Framework: React

Libraries: Redux, Material UI,

Database: Firestore / MongoDB

Hosting: Firebase/Netlify

Other: AlanAI, HTML, SCSS, JSX, Firebase Monitoring/Logging

It is worth nothing that I am still not entirely certain how I want to handle the back end of this application. I am also considering and researching using the combination of NodeJS and Socket IO, as I am unsure of the limitation of the purely Firebase back end. This is something that I intend on getting an answer to during my research prior to starting the development of the application. Another technology that I am considering using for this application is Storybook, but I am unsure whether it is worth it for an application of this size, which is comparatively small for its typical use case, but it's a very valuable technology to learn.

12.0 Evaluation

Unit Testing: Jest

Integration/End to End Testing: Cypress

For my unit testing, I will be using Jest, as it works very well with React components. Each of my React components will have tests assigned with them. By testing my individual components, it will help to ensure the overall stability of the application. I also intend to use Cypress for end-to-end testing as I develop the application. Cypress is an end-to-end testing framework that allows the developer to write tests, and is executed in the browser, which eliminates some of the weaknesses of previous testing tools, that require the developer to write implicit wait times etc. Cypress can also be used for integration testing. By the using both Jest and Cypress, I anticipate that my application will have sufficient testing to ensure stability. I will be writing these tests as I develop the various parts of my application.

12.1 Declaration of Ethics Consideration

National College of Ireland

DECLARATION OF ETHICS CONSIDERATION

School of Computing

Student Name: Nathan Hodgkiss

Student ID: X17381176

Programme Business Information Systems **Year:** 4

Module: Software Project

Project Title: Learning to Code Chat Application

Please circle (or highlight) as appropriate

This project involves human participants	Yes / No

Introduction

Secondary data refers to data that is collected by someone other than the current researcher. Common sources of secondary data for social science include censuses, information collected by government departments, organizational records and data originally collected for other research purposes. Primary data, by contrast, is collected by the investigator conducting the research.

A project that does not involve human participants requires ONLY completion of Declaration of Ethics Consideration Form and submission of the form on module's Moodle page

A project that involves human participants requires ethical clearance and an Ethics Application Form must be submitted through the module's Moodle page. Please refer to and ensure compliance with the ethical principles stated in NCI Ethics Form available on the Moodle page.

The following decision table will assist you in deciding if you have to complete the Declaration of Ethics Consideration Form or/and the Ethics Application Form.

Public Data	Y	Y	Y	Y	N	N	N	N
Private Data	Y	Y	N	N	Y	Y	N	N
Human Participants	Y	N	Y	N	Y	N	Y	N
Declaration of Ethics Consideration Form	x	X	x	X	X	X	x	
Ethics Application Form	X		X		X		X	

Please circle (or highlight) as appropriate

The project makes use of secondary dataset(s) created by the researcher	Yes / No
The project makes use of public secondary dataset(s)	Yes / No
The project makes use of non-public secondary dataset(s)	Yes / No
Approval letter from non-public secondary dataset(s) owner received	Yes / No

Sources of Data:

It is students' responsibility to ensure that they have the correct permissions/authorizations to use any data in a study. Projects that make use of data that does not have authorization to be used, will not be graded for that portion of the study that makes use of such data.

Public Data

A project that makes use of public secondary dataset(s) ***does not need ethics permission****, but* ***needs a letter/email from the copyright holder*** *regarding potential use.*

Some websites and data sources allow their data sets to be used under certain conditions. In these cases, a letter/email from the copyright holder is NOT necessary, but the researcher should cite the source of this permission and indicate under what conditions the data are allowed to be used. See Appendix I for examples of permissions granted by Fingal Open Data, and Eurostat website.

Where websites or data sources indicate that they do not grant permission for data to be used, you will still need a letter/email from the copyright holder. For example, see Appendix II for an example from the Journal of Statistics Education.

Private Data

A project that makes use of non-public (private) secondary dataset(s) must receive data usage permission from School of Computing.

An approval letter/email from the owner *(e.g. institution, company, etc.)* ***of the non-public secondary dataset must be attached to the Declaration of Ethics Consideration****. The letter/email must confirm that the dataset is anonymised and permission for data processing, analysis and public dissemination is granted.*

Evidence for use of secondary dataset(s)

Include dataset(s) owner letter/email or cite the source for usage permission

CHECKLIST

Non-public/private secondary dataset(s) -Owner letter/email is attached to this form ***OR*** Citation and link to the web site where permission is granted – provided in this form	Yes / No Yes / No

ETHICS CLEARANCE GUIDELINES WHEN HUMAN PARTICIPANTS ARE INVOLVED

The Ethics Application Form must be submitted on Moodle for approval prior to conducting the work.

Considerations in data collection

- Participants will not be identified, directly or through identifiers linked to the subjects in any reports produced by the study
- Responses will not place the participants at risk of professional liability or be damaging to the participants' financial standing, employability or reputation
- No confidential data will be used for personal advantage or that of a third party

Informed consent

- Consent to participate in the study has been given freely by the participants
- participants have the capacity to understand the project goals.

- Participants have been given information sheets that are understandable
- Likely benefits of the project itself have been explained to potential participants
- Risks and benefits of the project have been explained to potential participants
- Participants have been assured they will not suffer physical stress or discomfort or psychological or mental stress
- The participant has been assured s/he may withdraw at any time from the study without loss of benefit or penalty
- Special care has been taken where participants are unable to consent for themselves (e.g children under the age of 18, elders with age 85+, people with intellectual or learning disability, individuals or groups receiving help through the voluntary sector, those in a subordinate position to the researcher, groups who do not understand the consent and research process)
- Participants have been informed of potential conflict of interest issues
- The onus is on the researcher to inform participants if deception methods have to be used in a line of research

I have read, understood, and will adhere to the ethical principles described above in the conduct of the project work.

Signature: ……

Date: 02/11/2020

Appendix I

1) Fingal Open Data: http://data.fingal.ie/About

Licence

Citizens are free to access and use this data as they wish, free of charge, in accordance with the Creative Commons Attribution 4.0 International License (CC-BY).

Note: From November 2010 to July 2015, data on Fingal Open Data was published in accordance with the PSI general licence.

Use of any published data is subject to Data Protection legislation.

Licence Statement

Under the CC-BY Licence, users must acknowledge the source of the Information in their product or application by including or linking to this attribution statement: "Contains Fingal County Council Data licensed under a Creative Commons Attribution 4.0 International (CC BY 4.0) licence".

Multiple Attributions

If using data from several Information Providers and listing multiple attributions is not practical in a product or application, users may include a URI or hyperlink to a resource that contains the required attribution statements.

2) Eurostat: https://ec.europa.eu/eurostat/about/policies/copyright

COPYRIGHT NOTICE AND FREE RE-USE OF DATA

Eurostat has a policy of encouraging free re-use of its data, both for non-commercial and commercial purposes. All statistical data, metadata, content of web pages or other dissemination tools, official publications and other documents published on its website, with the exceptions listed below, can be reused without any payment or written licence provided that:

- the source is indicated as Eurostat

- when re-use involves modifications to the data or text, this must be stated clearly to the end user of the information

Appendix II

Journal of Statistics Education: http://jse.amstat.org/jse_users.htm

JSE Copyright and Usage Policy

Unlike other American Statistical Association journals, the Journal of Statistics Education (JSE) does not require authors to transfer copyright for the published material to JSE. Authors maintain copyright of published material. Because copyright is not transferred from the author, permission to use materials published by JSE remains with the author. Therefore, to use published material from a JSE article the requesting person must get approval from the author.

12.2 Reflective Journals

Reflective Journal – October

Student Name: Nathan Hodgkiss

Group: Business Information Systems

Student Number: x17381176

What was done:

The first few weeks of the software project module the focus was coming up for an idea for the software project. Last semester I had completed a 6-month internship as a web development intern, so quite quickly that was where I started to think of some ideas, as that is the direction I want to continue my career once I graduate. I had a couple of initial ideas, which I sent to Frances to discuss, and eventually ended up on the idea for my project pitch.

Discussion:

I personally found coming up with the idea extremely difficult, which is a recurring problem I have faced throughout any module that tasks you with coming up with an innovative idea, but due to the circumstances of online learning I found it particularly challenging. During the time I spent researching my project, I also spent some time thinking of the technologies that I wanted to use, these included React, Redux, AlanAI and a database. I wanted to learn and develop with these technologies as React is one of the most popular frameworks for web development currently and is something that I got a small example of during my internship that interested me.

I have two personal goals for this project, the primary one is to achieve a good grade, and my secondary goal is to use it as a showcase project when I go looking for further employment in the software development field.

What is next:

My focus now turns to preparing my project proposal report over the next few weeks, assuming that my idea is accepted by my assigned supervisor. In my final year, I have put an emphasis on planning a lot more than I would have previously, which I hope to carry over into my software.

Reflective Journal – November

Student Name: Nathan Hodgkiss

Group: Business Information Systems

Student Number: x17381176

What was done:

During the month of November, the focus shifted from finishing up on the draft of the project proposal document, to now focusing on the technical report. In the meantime, I met with my supervisor, to discuss the project proposal. During this meeting, it was decided that one aspect of the project, the 'Resource Hub' aspect, needed further fleshing out. It was also during this meeting that I started to think of the application in a potentially different light, perceiving it as more of a mini project/personal management tool as well as a chat application, for these heavily digital times. From further discussion with my assigned supervisor, the idea of an online programming journal, potentially with an AI implementation that analyses the code, along with a resource & chat section, could make up the bones of the application. I have completed the first section of the technical report, I am targeting completing the rest of the document before the end of the month

During the next stage of the technical report, I will create some examples use cases, as well as some mock-ups of the UI that I am thinking off, as well as overall requirements gathering.

Discussion:

This phase off the project where I am required to continue to flesh out the concept of the application, has been my weakest personal point, as I struggle with coming up with creative ideas, that are unique, that solve problem that has not already been solved, as this has usually been the task with regards to projects that I have undertaken. After further discussion with my supervisor, I am pleased with the idea that has been created above and will move onto fleshing out the use cases and figuring out potential technologies to use for each use case.

What is next:

After my project technical report is complete/mostly complete, I will begin development on the prototype of the midpoint video presentation. As it stands, I will be targeting a relatively developed front end and potentially a mocked-up back-end, to highlight functionality.

Reflective Journal – December

Student Name: Nathan Hodgkiss

Group: Business Information Systems

Student Number: x17381176

What was done:

During late November/ early December I completed my technical report draft for midpoint. My main focus during this period was getting a functioning prototype developed for the midpoint. I created the skeleton UI that would fit around all my different sections/component. I created the resources section, which made an API call using the user input, and dynamically displayed books based on the search, including a fully fleshed out UI. I created the chat UI, along with some mock functionality that did link to my Firebase database. I created my programming journal section, which was effectively a form that turned the users input into JSON. I did this as I am contemplating how best to take the user input in order to perform an analysis on it using AI. This section is the one that has the most work left and unanswered questions.

What is next:

When we are back in the new year, I intend to continue to develop my application, by building on what I have already created.

Reflective Journal – January (2021)

Student Name: Nathan Hodgkiss

Group: Business Information Systems

Student Number: x17381176

What was done:

In January, the objective was to get back up to date with regards the software project. The first task that I completed was creating a draft for my profile overview for the project website. I then began development on the software project itself. I began by working on the development for the login/registration functionality, as that was something that I skipped for the prototype. I implemented a sign-in using Firebase, using the Google Sign in functionality. I will now be adding a more traditional login as another option for the application. This month was about getting back up to speed on the project and remind myself of what my key goals/functionality that I need to achieve are.

What is next:

I plan on finishing my authentication system for the application in the next week or two. I then plan on tackling the functionality of logging journal entries which will require some more research to implement. I will also need to figure out the AI implementation that I intend on using for the project.

Reflective Journal – February (2021)

Student Name: Nathan Hodgkiss

Group: Business Information Systems

Student Number: x17381176

What was done:

In February, I continued to work on my software project, having got back up to speed. I started by fixing some of the UI issues I was having on the note taking section of the application. I then made good progress on the journal entry aspect of the application. I also did some research into natural language processing libraries for JavaScript, to see if there was the possibility of implementing it in some aspect of the application. My research showed that two libraries in particular, Natural and Compromise could be of use.

What is next:

I plan on finishing my journal entry system before moving onto trying to implement some aspect of natural language processing to my application, and also continuing to review the UI to clean up the overall application.

Reflective Journal – March (2021)

Student Name: Nathan Hodgkiss

Group: Business Information Systems

Student Number: x17381176

What was done:

In March I finished up the basic functionality for my project, including the base programming journal aspect. I ran into a few game breaking bugs upon implementation that I worked through. This was required in order to move onto another major piece of functionality, IE natural language processing, as the text that will be analysed is provided by the user, so I had to get this piece working in order to begin that. I also cleaned up some other glaring bugs through testing in other parts of my application, including the resources section, where I had to include some try/catch as it was crashing the application. Also, during March I met with careers and finished my project overview and personal bio. At the end of March, I began to implement the natural language processing aspect of the journal, using a JavaScript library called Compromise. At this stage I am just testing various ways to implement to try and figure out which is the cleanest and most efficient method. Finally, I added some content to the testing section of the technical report and added a use case diagram/flow as well.

What is next:

Next, I will focus on attempting to implement the natural language processing while revising my technical report. I will also be making UI changes/cleaning up a few bugs.

Reflective Journal – April (2021)

Student Name: Nathan Hodgkiss

Group: Business Information Systems

Student Number: x17381176

What was done:

During April I finished up my natural language processing implementation for the languages of Java, JavaScript, Python and HTTP error codes. I also continued on my documentation, fleshing out the GUI sections and began the implementation section off my report. I also test deployed my application to check if there would be any errors when live which was useful as there were a few bugs that I plan on fixing before final deployment.

What is next:

I plan on getting a basic implementation of AlanAI in my project, which is the last piece of functionality that I have yet to implement. I will also fix the bugs that occurred live in my test project deployment. After these are completed, I will be working on my project poster, presentation, and video as part of my final submission.