OPEN BANKING MANAGEMENT APPLICATION

Computer Science with Cybersecurity

Joseph Batchelor
Student ID: 19804258

Supervisor
Jennie Harding

Second reader
Robin Heath

Abstract

Open banking platforms offer secure quick access to finical data from any location. They are a growing service that satisfies the needs of customers by providing quick access to real time banking data. Open banking services allow businesses to use data to provide secure and beneficial services while giving consumers more control over their data. With the use of this report, it explores the different development operations, critical considerations and implications made that all contributed to open the final design of my open banking applications.

URL location: https://jb1828.brighton.domains/FinalYearProject/index.php

GitHub source code link: JosephBatchelor/FinalYearProject: Here is the source code for my Open banking application (github.com)

Table of Contents

Introduction

What have I created

The artifact created is an Open Banking management application that retrieves client banking data from external FSCS approved banks. This application utilises banking data to perform analytical operations to produce visual charts and to display multiple account transactions, balances, standing orders and current savings. It provides the ability to view and manage multiple banking accounts from a single device.

What problem does this solve and how

The problem this application solves is to lessen the control that monopolies have within the financial sector. This application and others alike enable consumers to remove the exclusive possession that banks have on customer data. Open banking applications allow other businesses to use data to provide secure and beneficial services while giving consumers more control over their data.

Aim and objectives of this project

My aims and objectives for this project are to design, create and deliver a professionally styled application which can offer users the ability to view different financial information, perform different financial operations and to have more control over their data from a single location.

Below are some objective that I need to perform in-order to achieve my aims.

- Follow the Gantt chart and complete all critical required operations to create a fully functional project. This ensures I have enough time to correctly implement critical operations.
- To generate important and necessary functional and non-functional requirements that will help me meet my aims. This provides focus points for me to work towards.
- Collect data and research different UI design to provide a professional visual design and to integrate effective analytical operations.
- To code and or introduce security methods within my project as well as to configure software and other components so that they meet security standards.

Deliverables

ID	Deliverables	Summary	How will these be measured
D1	A fully functioning Web and mobile application	My expectation is to create an application that users can access through either a website or through a responsive mobile version of the website.	To use of various tests to verify that the application created is functional. Some tests such as responsiveness to see if the application is mobile friendly, accessibility testing and even security testing. If these tests are successful, this will support the deliverable and suggest that is was achieved.
D2	Project report	A compressive report showcasing the learning, researching and development of this project.	By making sure that the contents of the dissertation are met, and I have fully explained each criteria needed for a suitable mark as well as made a submission before the due date.
D3	An example use account	This is an example account that will be used to demonstrate my project, it has been designed in a way to be accessible and to provide a scenario of a long-term user that used different functionalities.	By creating an example account to be used for testing and understanding on my software artefact. The measurable way of determining whether this will be met is if I did or did not make an example account.
D4	Effective security and privacy implementations	These are different security and privacy integration techniques that have been implemented or used to configure the working environment and or used within my application to help mitigate external and internal threats.	Deliverable by performing numerous penetration and security testing to verify the effective of the various security implementations I have introduced. If I am able to affectively prevent and or mitigate 80% of common web site vulnerabilities or exploits this can suggest that the deliverables will be successful.

Table 1 – displays the what I will deliver.

Methodology

This section will discuss in detail the methodology used during development of my software artifact. This will cover different aspects such as explanation and justification to the development tools and environments used, project management, testing, risk assessment and legal and ethical considerations.

Development tools and environments used

HTML & CSS

HTML will be used to create the structure of my web application and CSS is used to control the design and presentation of the site. The decision was made to use both tools as they provide easy front-end development of my application (MDN, 2022).

Alternatively, java applets was considered instead of traditional HTML however, this approach would not support server-side development when using the desired API and so the choice of HTML seemed sensible. Additionality, my confidence, and knowledge in traditional web development tools such HTML, CSS and JavaScript outweigh the use of Java applets.

JavaScript

JavaScript is used to control the behaviour of objects to provide a dynamic response. The decision to use JavaScript was considered due to its larger support of packages and libraries as well as it being more effective and efficient when working within the cPanel environment. The use of JavaScript will be for integrating API's as it provides third party libraries to extend functionality such express framework (MDN, 2022).

However, Typescript was considered instead of JavaScript as it offers better technical features, such as better memory management and an arrange of runtime and compile time operations. This is because Typescript must be compiled in order for it to be functional within HTML documents.

jQuery

jQuery is a JavaScript library that will provide extended features such as event handling and ajax. This offers versatility by making JavaScript easier to use. It is a highly sought and used library that will provide a better and efficient approach to web development. Attentively angular was considered however, with research it showed that jQuery is a highly used library (W3techs, 2022)

AJAX

AJAX allows for asynchronous communications with different servers. This will work alongside JavaScript to extract and send data to PHP which will then be processed and used to perform different database related operations.

PHP

PHP is used to create a dynamic application to process and handle data requests. PHP is a language I am familiar with and will be used for sending and retrieving data. The idea to use JavaScript for data handling was considered, this approach is currently on my node.js server which handles banking data. The idea to also use it for my front-end was also considered however, PHP offers better database security integration methods as it is ran server side.

SQL

SQL is a structured query language I am most confident with using. It will be used alongside PHP and JavaScript for making requests to the database. During early development, SQL was heavily used on PhpMyAdmin for the preparation phase of creating different tables and fields for data to be stored in.

Brighton domains (cPanel)

Brighton domains is a service that utilises cPanel's hosting software online which provides an array of applications and tools for web development. This allowed me to set up my project using a public domain making it publicly accessible. This was a significant improvement as it provided every resource needed to complete my project. Another reason is my familiarity with this software from previous exercises and projects meaning that I was comfortable with using this application and the resources it provided like phpMyAdmin and Node.js.

WordPress, which is a similar application to cPanel was considered however, at the expense of paying a monthly subscription, this wasn't an ideal approach as I have never used the application before. To be paying for something that necessarily doesn't provide all the resources needed for development was a firm decline.

Visual studio code(VSCode)

VScode is an IDE that will be used for writing code as it is deemed the best for both web development and JavaScript development. (ANON, 2022). The IDE offers built in support for JavaScript and Node.js and extensions for PHP such PHPUnit and Jest for unit testing. This heavily helps with development, as Brighton domains does not use an IDE but instead uses basic text Editor. This does not provide any features that VScode would, such as syntax highlighting, templates and easy debugging.

Node.js

Node.js is a backend environment that is designed to execute JavaScript code outside of the web browser. This was used to house my Rest API to retrieve clients banking data and to store them within the database. The reason I choose to use Node.js is because of its use of JavaScript meant that I was familiar with most libraries. However, with Node.js this environment offers limitless packages to be utilised as well as frameworks designed to help and improve development of web applications and integration of APIs for example Express. CPanel has its own Node.js application which is an integrated feature that allows quick and easy development of Node servers. This further supports the decision to integrate this within my project.

Express Framework

Express is a framework designed to improve the way web application are built (MDN, 2022). This will be used for the backend development to hold my REST API. The reason I choose to implement express over the traditional backend approach of using the HTTP library is because it provides a simpler approach and requires less code for the same actions to be completed.

PhpMyAdmin

PhpMyAdmin is a MySQL visual management application designed to provide a visual interface for the creation and manipulation of a database. After choosing which relational database management system to use (MySQL), I needed to decide which management tool to use coincide with it. PhpMyAdmin was my first choice for database management as it is an environment that I am familiar with as well as it is freely accessible within the cPanel dashboard. I haven't had much experience with other management tools and so I was confident with the decision to use PhpMyAdmin.

True layer Data API

True layer is an open banking platform that provides the necessary tools to securely access financial information. It is a highly used platform used by popular retailers such as Revolut or Trading 212. True layer offers a development environment that allows you to integrate their various API's. This console will be heavily used to adjust, bridge, and secure the connection between users and their ability to access data. The True layer Data API is the main critical part of my software artifact. Its job is to retrieve and process user banking data securely and quickly. It requires backend integration using Node.js and works alongside true layers console.

Project management

Deciding a suitable project methodology approach was important. I had two options to choose from which was an agile or waterfall approach. Before I decided, I did research to determine which one would be more relevant for my project (Forbes, 2022). Agile is the approach of managing a project by breaking it down into several tasks and is the idea of continuous integration and development. Waterfall approach is focused on linear progression from a start point to the finished project being the ned point.

Waterfall		**Agile**	
Advantages	**Disadvantages**	**Advantages**	**Disadvantages**
+ Outlines a concrete plan for the project from start to finish. + Requirement can be established early on, and development can stay in line with them. + Workflow of the project is more structured and requires a deliverable for progression.	- Because development requires a linear progression some areas can postpone further progression. - Errors and problem may not occur until later with later development tasks. This may require checking through previous tasks. - A preliminary outline of the project plan is required before beginning, this eliminates flexibility and or potential changes.	+ Offers flexibility to change project direction and to try new ideas and approaches. + More productivity and efficiency because tasks are broken down. + Tasks can be completed at different rates using sprint schedules. Instead of focusing everything on one deadline. + Requirements can be improved during development.	- Trying new ideas or approaches can potentially extend the timeline and or ultimately affect the overall project negatively. - Dealing with multiple complex tasks separately can cause problems and backtracking as well as potential of problems with previous tasks. - Determining the timeline for a project can be unpredictable and sometime delay it.

Table 2 – pros and cons table of the different methodologies

For the initial stages of the project, I intend to use the waterfall approach. This allows me to create a strict timeline that focuses on the project as a whole, allowing me to develop requirements from the start. Developing requirements at the start will provide focus points for me to achieve as well as to make sure my project includes relevant features that are required. Waterfall will also provide time management and a better understanding of how each area of the project should be undertaken. Front-end development can follow a linear timeline as this an easy approach to do. However, other critical areas of development may require repetitive work and the opportunity to be flexible to introduce extended features, unfortunately, waterfall does not offer this approach. This is why I feel that also introducing an agile methodology would be beneficial for critical areas of the project such as integrating my API. This is so I can continuously adapt and improve that aspect through development. It will not negatively affect previous areas of development such as my front end as they would be separate. Additionally, because this is an individual project and agile typically is applied to group work, I will need to make changes, so it adapts to my project.

Planning and management tools

I have researched the best waterfall and agile tools to use throughout my project. For waterfall I will use a gantt chart, which is a simple plan that schedules the timeline of each task within the project and how long they should take. It is a useful tool to predict the resources needed and how long the project and each task should take. I developed this using excel and can be seen in ***figure 3***.

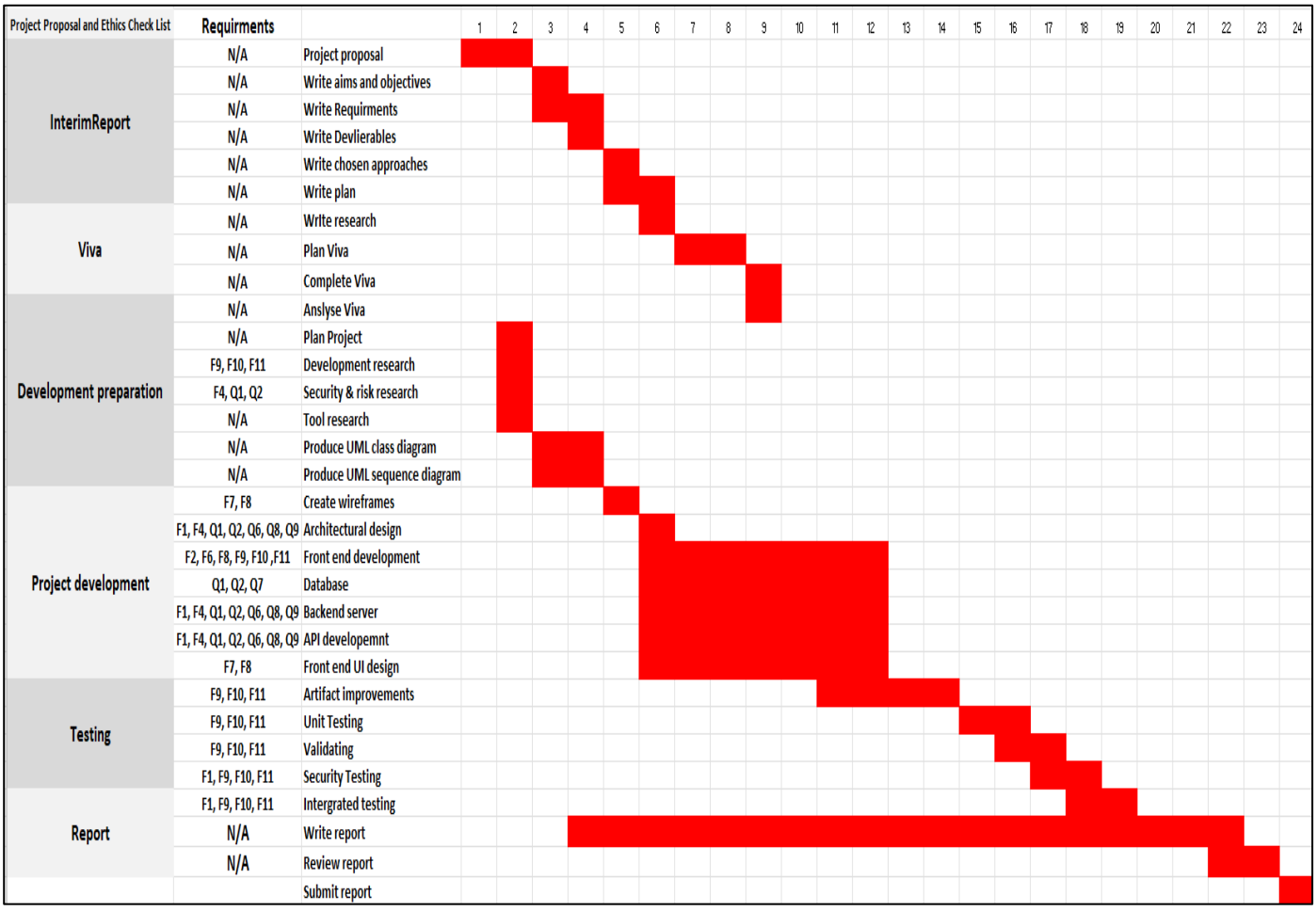


Figure 3 – Gantt chart used for long term project management.

For agile I will be using Trello this will specifically focus on the coding development as changes and adaptation can be made to offer extended features of improvement allowing me to be flexible. This organises different areas into separate tasks and provides an easy way to manage technical aspects to be completed on time and effectively, ***figure 4*** provides visualization of my Trello board.

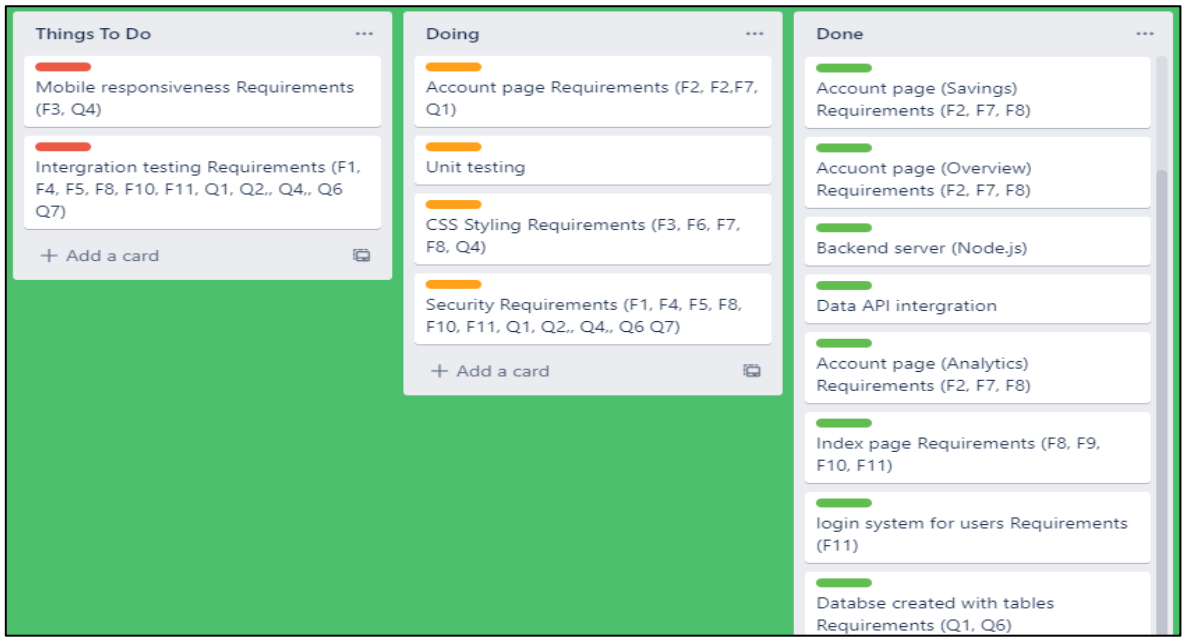


Figure 4 – Trello board

Testing framework and approach

Below is a list of testing methods that I have researched and decided to use throughout development. These methods have been selected to help support and provide evidence that certain elements are in line with my requirements.

WbPageTest.org

This is a performance testing tool that will be used to test and optimise my application (Meenan, 2022). The decision to use this tool is because its free and reliable, as I don't know any other site that can produce legitimate results. Most other tools require payment so the decision to use this tool was firm. Additionally, this site also offers other tools for testing such as security testing which may also be useful. This tool will help contribute to **Q3** of my non-functional requirements.

Responsive testing

To test responsiveness, I will be using AmIresponsive? (Am I Responsive?, 2022). This site provides different emulators to view my website on multiple devices. This will allow me to see and test how my site will look on different devices. This is a site I am familiar with as I have used it previously within the CI406 introduction to web development. This site was chosen as other responsive sites did not support my URL link.

HTML & CSS validators

These are HTML and CSS scanning tools used to verify that my application has valid HTML and CSS. The reason I choose to use this specific tool is because it was created by an international standard for the world wide web (World Wide Web Consortium). This tool will allow me to uncover HTML errors and problems to make sure that critical functional requirements like **F11** and **F1O** are achieved. As well as offering the ability to check my CSS so that styling values and tags are correct, this will help with F6, F7 and F8 of my requirements (W3C, 2022). There are other free validating tools such as HTML validator online (AppDevTools, 2022) which is developed by appdevtools. However, I decided to not choose this version as the W3C version offers URL scanning, source code scanning and offers both HTML and CSS scanning.

Node.js Jest unit testing

Jest is a unit testing library used to test and verify that critical backend operations are performing as intended. This is a new Unit testing approach I have recently learnt how to do. I decided to use Jest as it seemed to be a commonly used liberally with easy setup. Another library considered is Cypress which is another unit testing tool that I am unfamiliar with, I did not choose this library as it does not provide the same number of methods that Jest does as well as Jest has its own plugin package within the VScode IDE which allows for extended features to be used. Unit testing the backend environment allows me to make sure that my Rest API is correctly configured. This will help with **F1, F2** and **F5** functional requirements as they are API dependent objectives (Jest, 2022).

PHPUnit testing

PHPUnit is a unit testing library that will be used to test critical frontend operations such as registration (**F9),** updating user data (**F10**) and Logging in users (**F11**) which are all different functional requirements. An alternative library that I did consider was Simple Test, however, setting up the environment is more complicated than PHPUnit. It also requires the use of specific assert methods, whereas PhpUnit allows both native testing methods as well as custom methods provided by the framework (PHPUnit, 2022).

Risk analysis

Below ***table 3,*** lists potential risks that I may face within my project along with relevant mitigating methods. The CI607 information security management module taught me how to create a detailed risk assessment for a company. The same logic is used and applied for my project, to better calculate the probability of risks and the relevancy of risks to my project.

#	Risk	Prob	Impact	Mitigation
1	All my project code and or report is lost or deleted.	Possible	Will have re-do anything lost or even start project or report again from scratch.	Perform regular backups and use OneDrive as my back up source so that I'm able to access it from other connected devices.
2	Become ill and unable to enter uni.	Unlikely	Productivity will be affected ad schedule maybe affected.	Work from home and attend online lectures until better.
3	Another lockdown has occurred, and we all have to isolate.	Low	Unable to attend uni to access resources and to have face to fae meetings.	Work under lockdown conditions making sure I keep in communication with supervisor through email.
4	Brighton domains software which is how I host my website and database fails and or has a problem.	Low	Unable to access project affecting productivity and affecting schedule.	Make of copy of the project folder regularly and store it within an external device control panel software as a backup.
5	Failing to meet a deadline for another assessment due to lack of time management.	Possible	Can affect mental health as well as project may be set back.	If possible, apply for an extension as well as focus on what needs more attention and make sure work is completed on time.
6	Laptop unusable due to hardware failure.	Unlikely	Unable to access local files and may affect productivity.	If my laptop ever becomes usable then I can use my desktop to access my files using OneDrive.
7	Part time job making it difficult to focus on project.	High	Potentially affecting time management and or schedule can cause delays or setbacks.	Reduce hours and arrange for certain days off from work.
8	Project does not meet requirements, aims or expectations.	Possible	Can affect my final grade and final outcome of my project.	Regularly do work making sure that my overall goals, aims and requirements are met and are that tasks are being completed as pictured within Gantt table.
9	Not being able to learn or find it difficult to understand new areas needed for development of project i.e., certain coding or software aspects.	High	Makes development harder for key areas as well as makes certain task takes longer to create and understand. Can affect the duration the project takes.	This can be achieved using different resources and materials as well as using members of staff to help you completed or understand certain tasks or topics.

Table 3 – project risk assessment

Legal and ethical considerations

This section will explain relevant legal and ethical laws and regulations that should be considered when creating this application. The CI500 The law and cybersecurity module helped understand and find relevant cyber security legislative laws and guidelines for my project.

GDPR

The general data protection regulation is designed to maintain the privacy and security of data being processed. This is relevant to my project as I will be using an API to retrieve sensitive data, this means that I must handle and process this data lawfully. This legal law influenced me to create a requirement **Q2** which is to make sure data is processed correctly and lawfully (GOV.UK, 2022).

DPA

Another law learnt withinCI500 is the data protection act. This governs the way a company collect and process data, to protect people and their data from being inappropriately used without consent (GOV.UK, 2022). This law influence me to make sure that when retrieving banking data clients understand what they're retrieving and what is happening with their data. Also adding the ability to update personal data **F10** will provide freedom to control their data stored.

Web Content Accessibility Guidelines (WCAG 2.1)

These are a set of guidelines designed to provide insight on how to make web content more accessible to people with disabilities. This influenced me to introduce a wide range accessibility features and designs to make it was easy for those. This also led to me making an accessibility requirement which is to add accessibility features to improve user experience **F3**.

Product Description

Product description

The artifact created is an open banking web application that utilizes both client-side and server-side code as well as an integrated RESTful API. It does this to securely retrieve clients personal banking data and process these banking data into visualized charts and information. The management system provides detailed analytics and information into the clients' accounts.

Requirements

My functional and non-functional requirements have been chosen and ranked by prioritisation (1 being most important) ***table 6.***

Functional	Non-functional
F1. API retrieval of banking data. F2. Tailored analytics for customers. F3. Accessibility features added to improve user experience. F4. Security measures introduced against threats. F5. Authentication of a user when logging into banks. F6. Accessible through both web and mobile. F7. Added visualizations for clarity and better understanding. F8. Professional design. F9. Registration for users. F10. Updating and or adding users' data to accounts. F11. Logging in for users.	Q1. Availability of data is quick and always secure. Q2. Data is processed correctly and lawfully. Q3. Client requests must be processed by the system within a 5second response time. Q4. Project should be responsive between different devices. Q5. Frontend should have Encrypted channels between the database. Q6. The integrity of data should be maintained regularly. Q7. Passwords are hashed when stored onto the database Q8. Application and web server should run continuously to make the web portal accessible at least 95% of the time. Q9. The web application should be able to effectively manage 1000 active users before noticeable delays or performance affected.

Table 6 – project requirements

Reasons to why I choose these requirements

Identifying relevant requirements is important, requirements are focus points to ensure the creation of a fully functional application is met. Below are specific areas that I have used to determine my requirements.

- **Security: Q1**, **Q2**, **Q5**, **Q6**, **Q7**, **F4** and **F5** are different security related requirements. These will be used to introduce secure methods and connections to my web applications as my application will be dealing with sensitive banking data. Maintaining the integrity of data should be the main focus of development and will be achieved with various requirements.
- **Scalability: F1**, **F6** and **Q4** are scalable requirements which will help expand and scale certain properties of my application. An important scalable aspect of a web applications is multi device accessibility, with **F6** this will introduce responsiveness development allowing different size device to display my application.
- **Performance: Q3**, **Q8** and **Q9** are performance related requirements designed to help improve different components such as speed, memory, or power consumption .
- **Availability:** Allowing clients to maintain constant access to the application is an important design consideration that needs to be introduced. **Q1** and **Q8** requirements will allow me to introduce consistent connections with different resources offer continuous ability to different resources and operations.
- **Design:** Making sure the design of a system is clear and professional is important, as making an impression entices clients to use our application. This will be achieved as requirements **F2,F7** and **F8** are design related requitements which helps focus on the importance of styling.

Architectural Design

Through analyzing and researching popular architectural designs I was able to break down my application into different specific areas that contributed to my requirements and the final design of my application.

I learned how to create an architectural document from CI615 Object Oriented design and architecture module. This module provided a better understanding on how to correctly layout a document, how to correctly design different diagrams and to provide insight on the different types of architectural styles.

Architectural styles used

N-Tier client server style

This architectural style provides a multi-tier architecture, that separates an application into different components.

- This is beneficial as resources are not shared allowing for maximum performance from each tier.
- More security is achieved by separating components.
- Modifications can be made without affecting any of the other tiers.
- Can accommodate different size applications.

The way I have used this architectural style is by introducing a basic 2-tier web application design, this means that my webserver will communicate with my database server to retrieve and display data. My application can be considered small and therefore a 2-tier design seems like a reasonable and manageable size for one person to maintain. This architectural style allows me to meet and introduce desired qualities such as security (**Q1**, **Q2**, **Q5**, **Q6, Q7, F4** and **F5**), Modifiability, scalability (**F1**, **F6** and **Q4**) and performance (**Q3**, **Q8**, **Q9** and **Q3**).

REST architectural style

The REST architectural style is an API design used to retrieve user banking data via routing data through URL requests. True layer provides a restful data API service that provides quick and secure ways to retrieve users' personal information. Repetitional state transfer software is a very common approach used to retrieve resources from external entities. I have integrated this architectural style by using Node.js I used this style as it was a mandatory implementation in order to integrate the True layer data API.
However, this Style helps achieve the security (**Q1**, **Q2**, **Q5**, **Q6, Q7, F4** and **F5**) desired quality as the encryption method used by the API is AES.

Client server style

The client server style is an architectural approach where one or more clients interact with a server by making requests for resources and services. When applied to my application the virtual webserver provided by cPanel will interact with client to serve HTML pages and provide operations for different resources. This is a common approach to multiple clients allowing for concurrency. This was automatically integrated due to the cPanel software providing virtual webservers to serve HTML pages to clients (waterloo, 2022).
This meets the scalability (**F1**, **F6** and **Q4**) and performance (**Q3**, **Q8, Q9** and **Q3**) desired qualities when implemented.

Architectural Design Diagrams

This section of the document delivers different design diagrams learnt within the CI6015 module. Designed to provide better clarity of the system architecture, this should consist of the following:

- Package diagram
- Deployment diagram
- Sequence diagram
- Use case diagram

Figure 5 provides a basic UML diagram of the applied architectural styles.

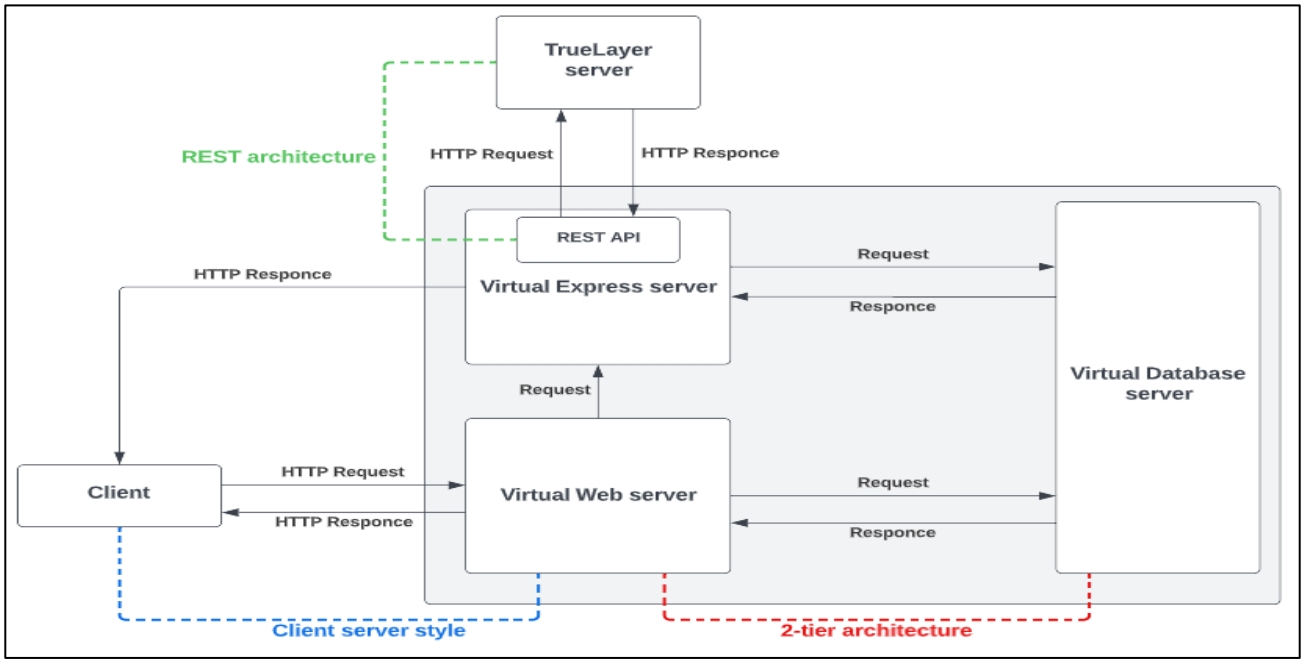


Figure 5 – basic UML layout component diagram of the project

Package diagram

The package diagram seen in ***figure 6*** provides an insight of various elements used to create my application.

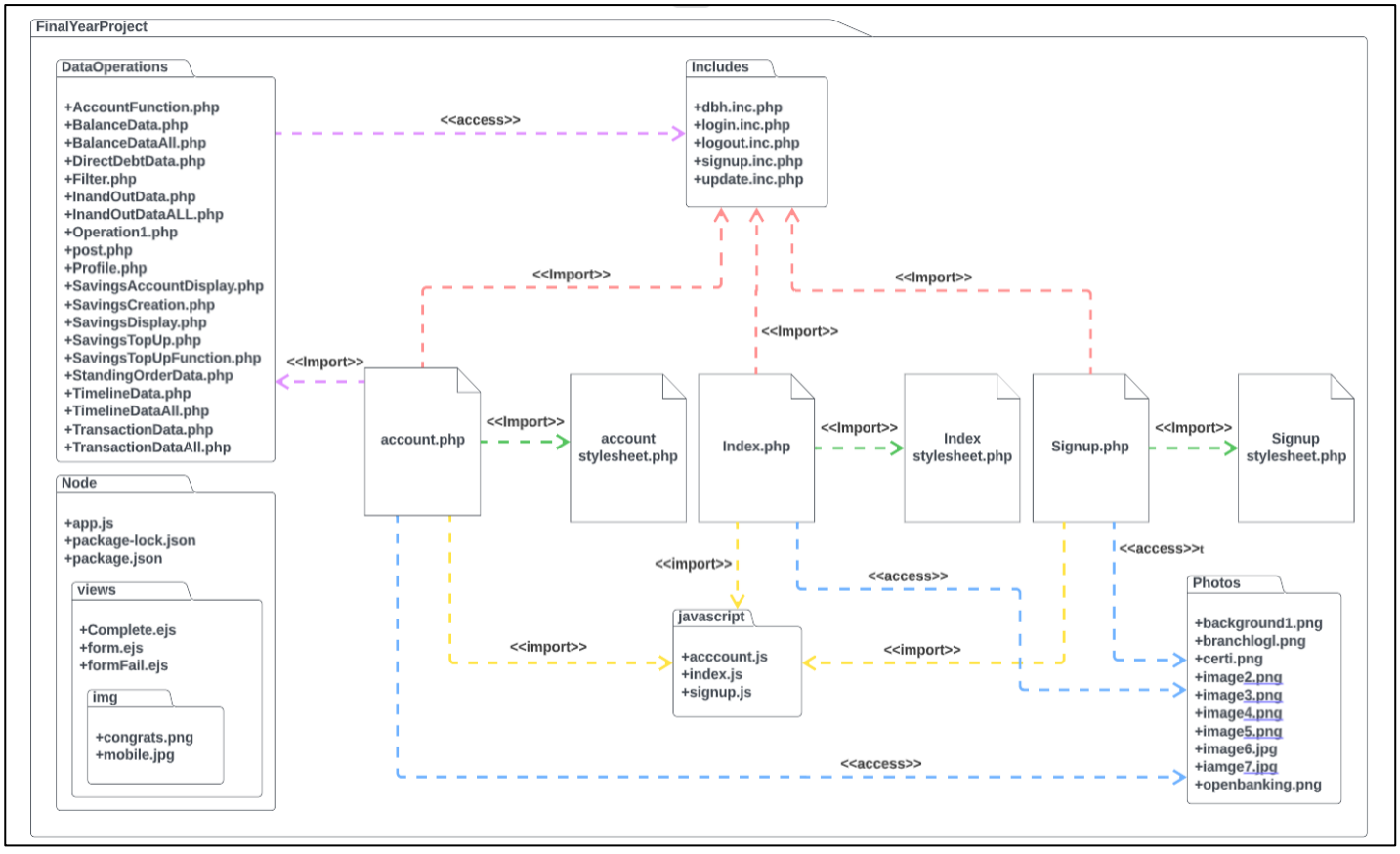


Figure 6 – package diagram

Deployment diagram

Figure 7 displays a deployment diagram which shows the various nodes and their components and artifacts. This is to provide an understanding of the physical architecture and communication of nodes.

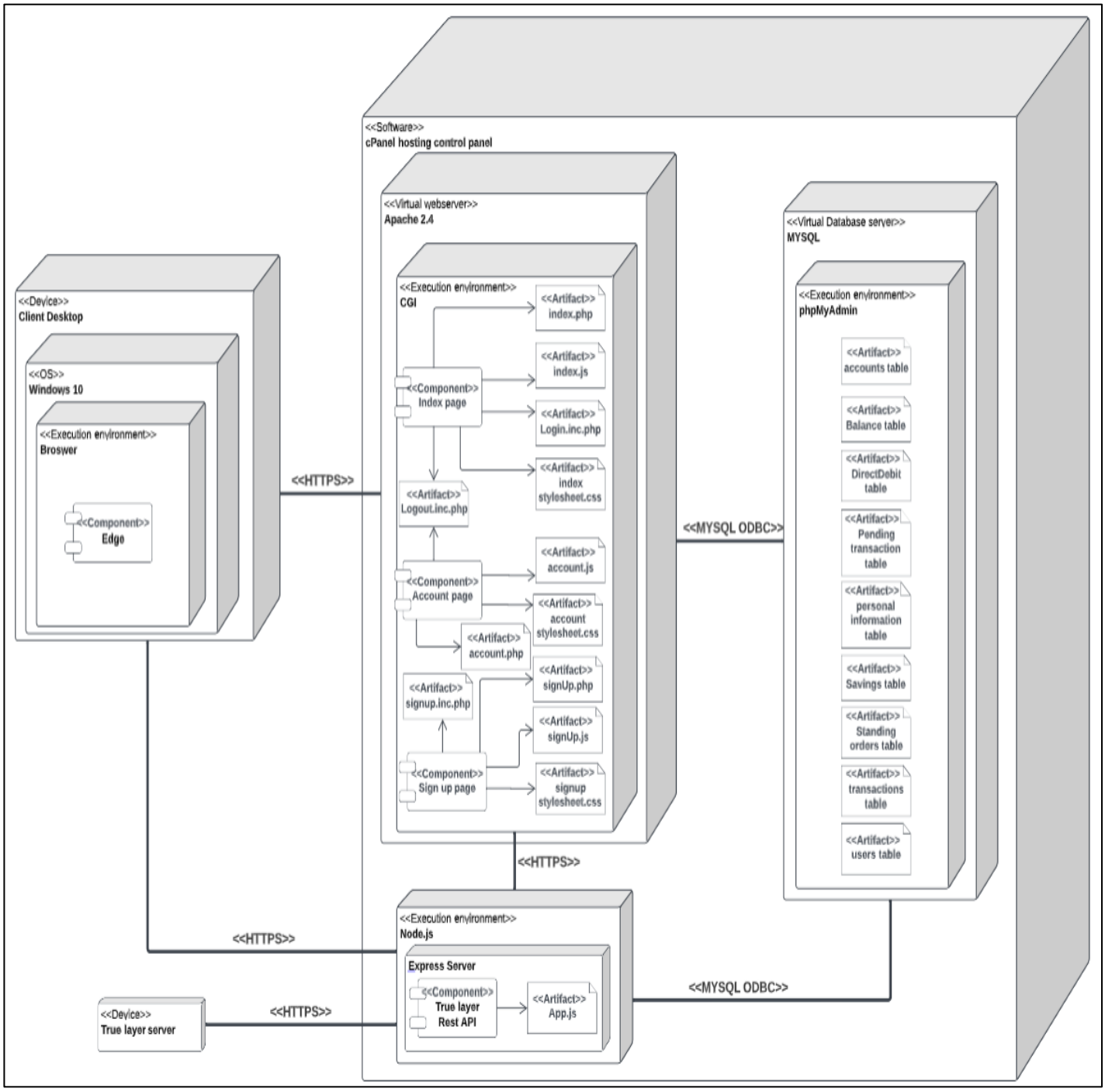


Figure 7 – Deployment diagram

Sequence diagram

User operations sequence diagram

The following diagrams shows the different objects that the client can interact with. This provides insight to the communication between different objects in order for the client to perform different account related operations.

Login operation

Figure 8 shows the login in operation.

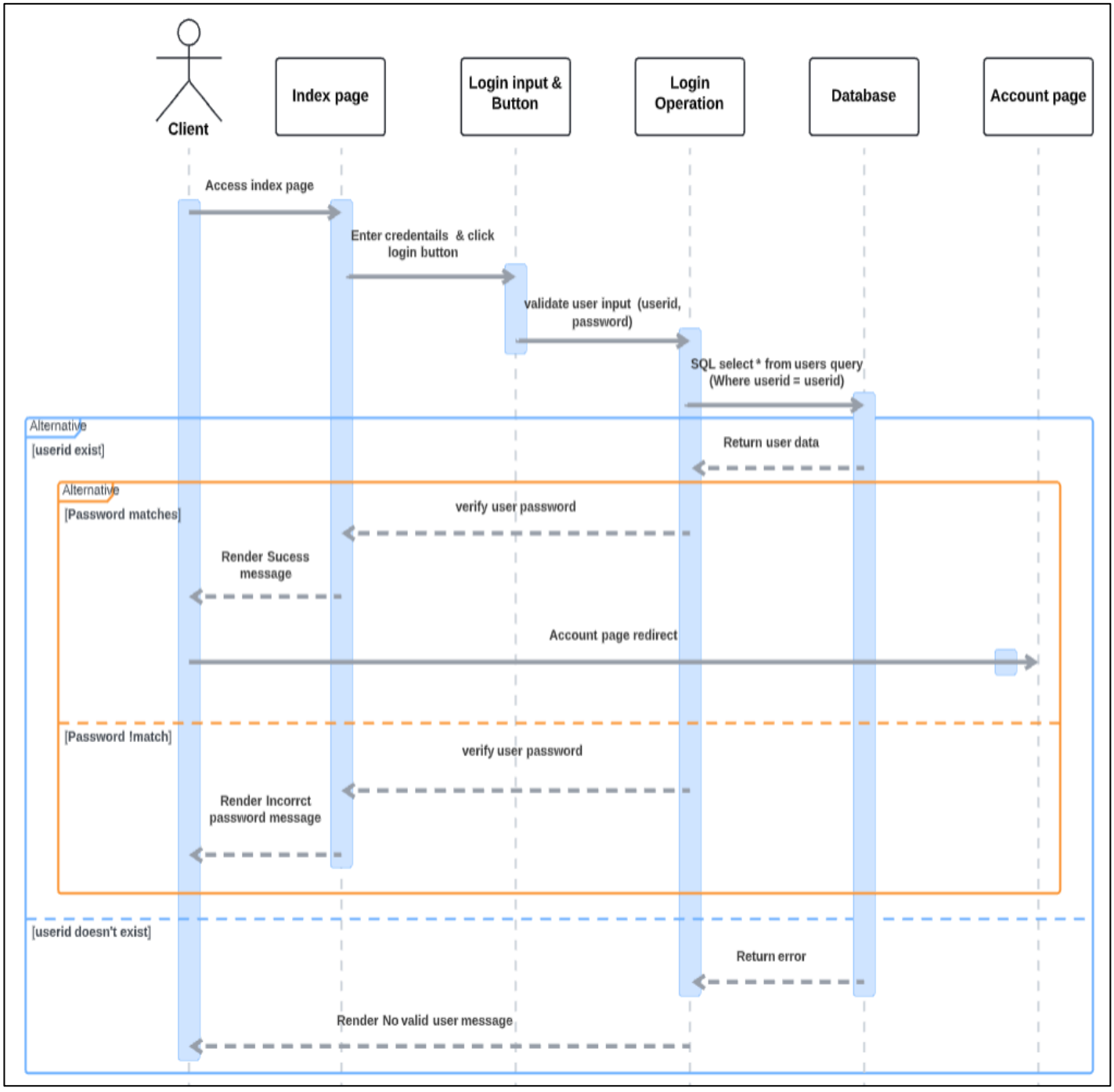


Figure 8 – sequence login operation

Registration operation

As seen in ***figure 9***, the next operation that the client can do is to create an account.

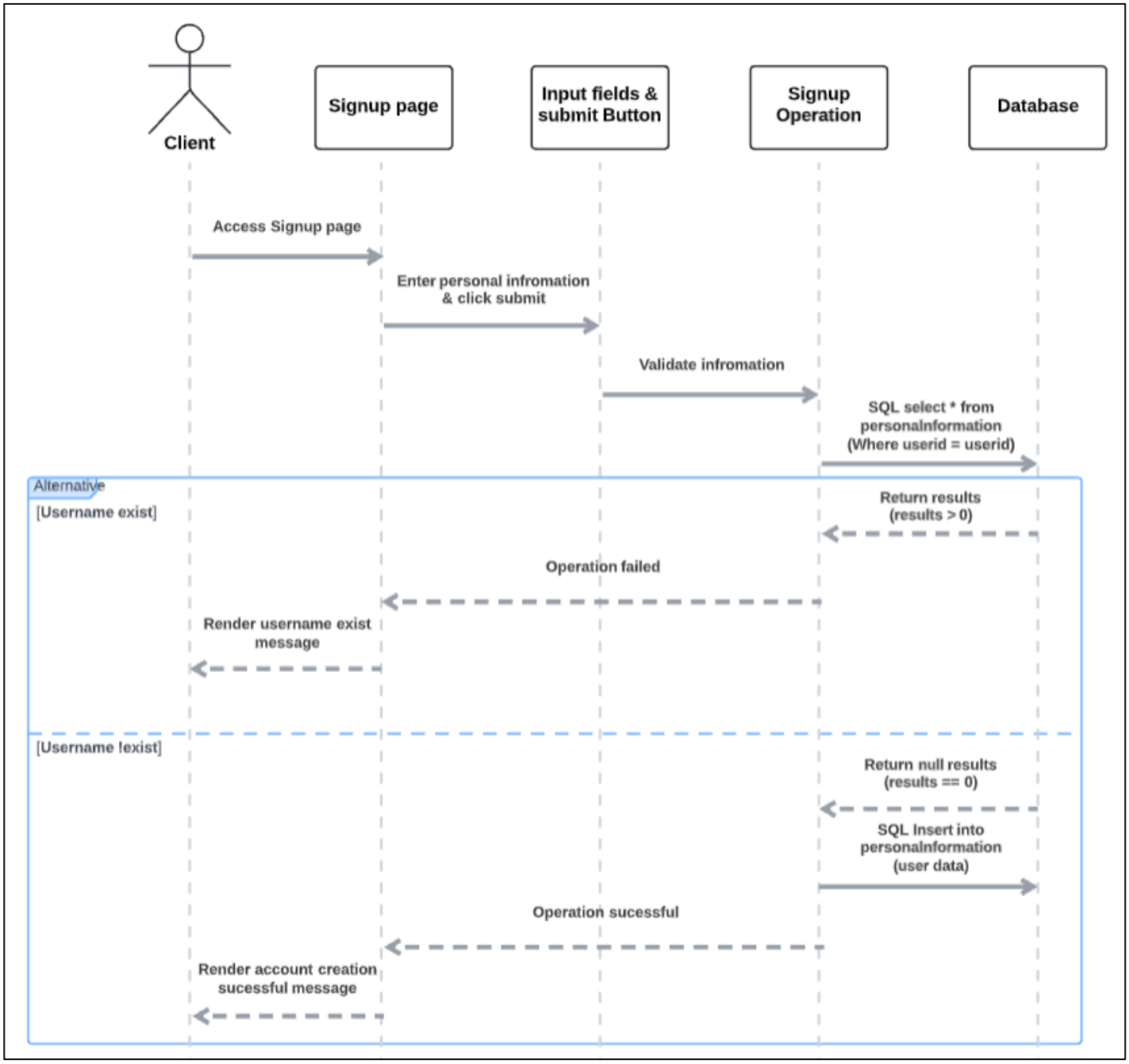


Figure 9 – sequence registration operations

Update account data operation

Figure 10 shows the final operation which is the ability for clients to view and update their personal data.

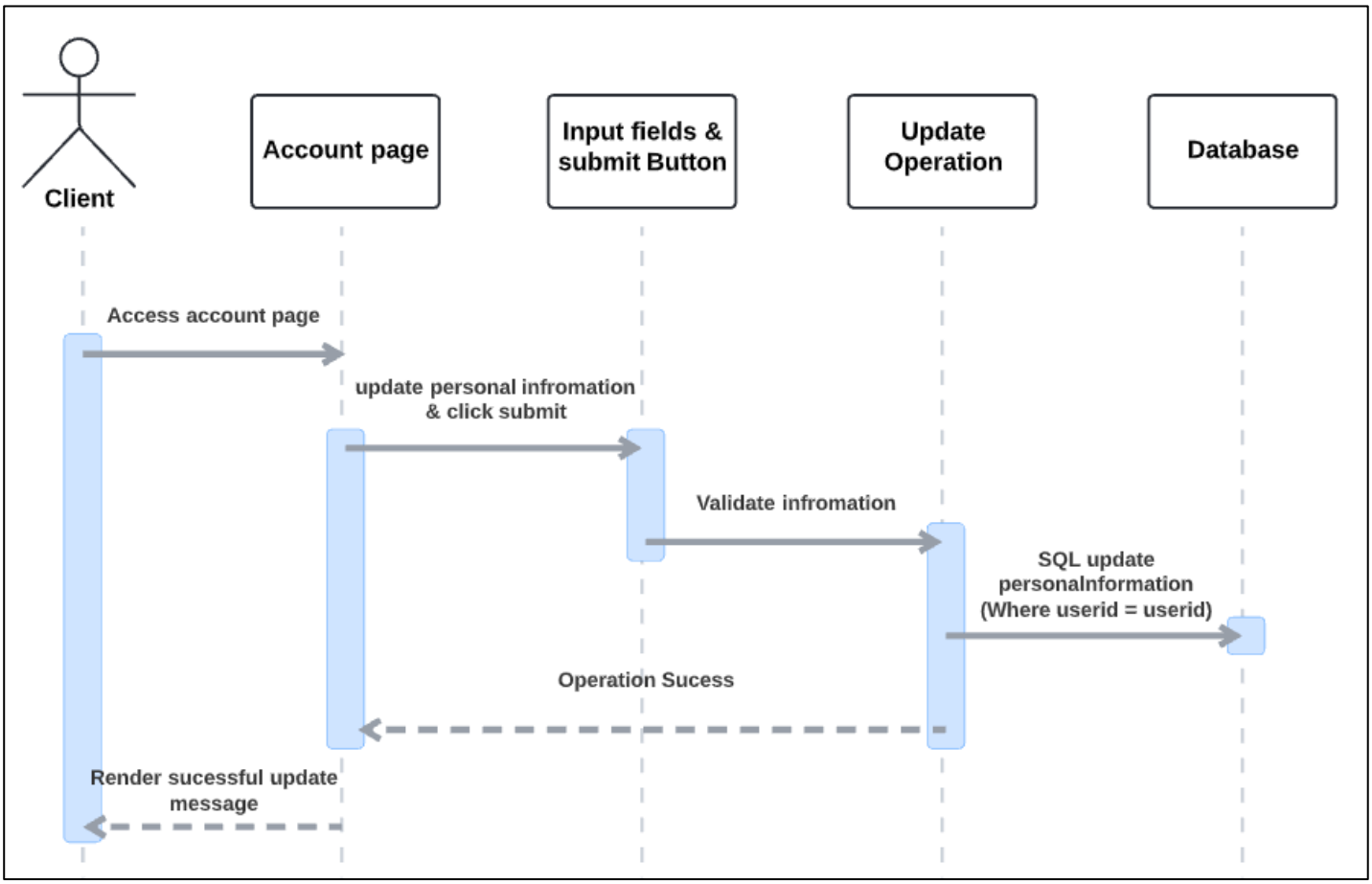


Figure 10 – sequence update data operation

Hardware communication sequence diagram

Figure 11 provides a basic sequence diagram showing the communication between hardware. The following operations being performed are serving content to the user, the client login into their account and adding new banking data.

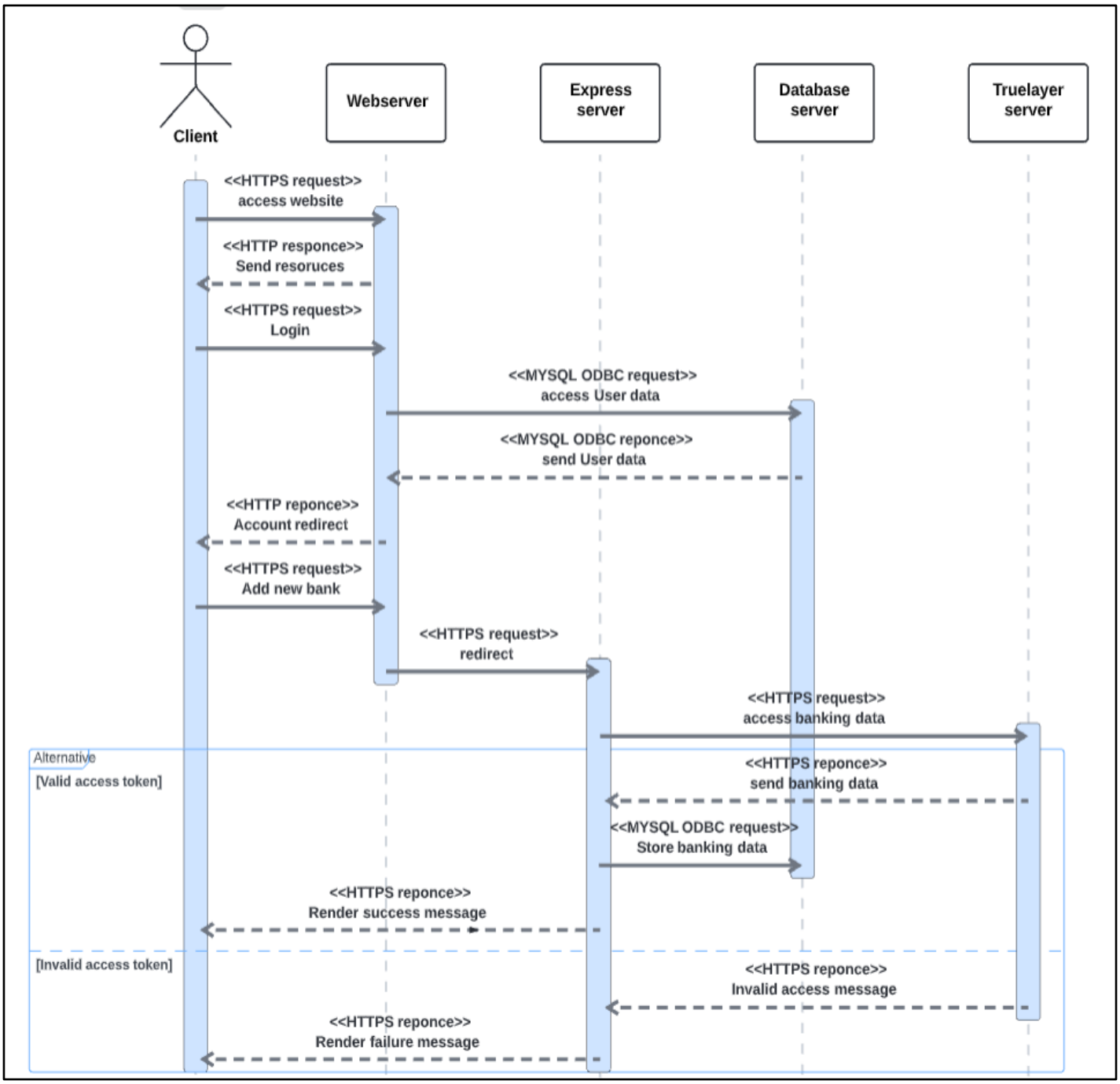


Figure 11 – Hardware communication sequence diagram.

Use case diagram

Figure 12 provides a use case of what the client can perform within the application as well as the different elements and external entities involved.

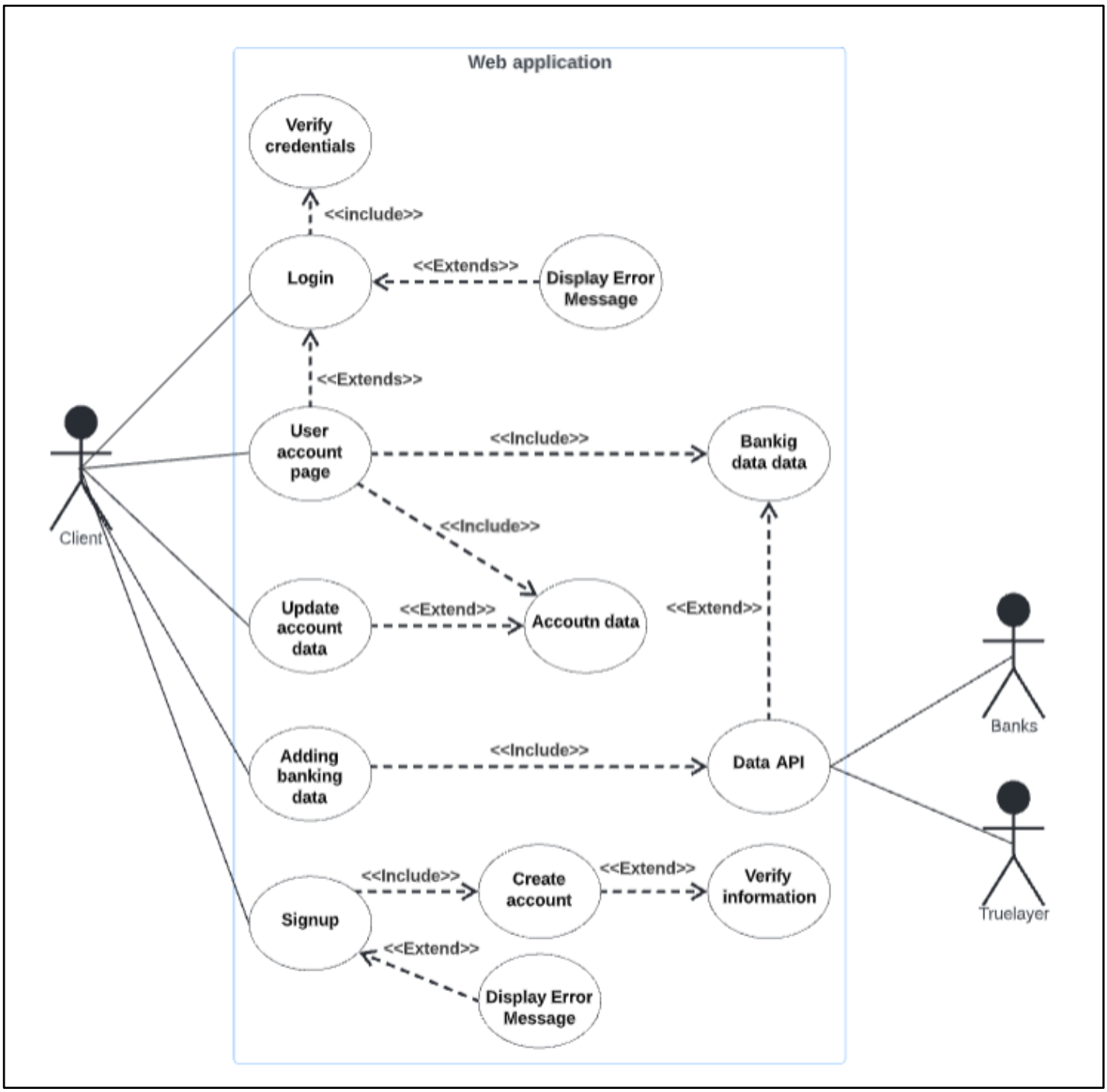


Figure 12 – use case diagram

Design and development

This section will discuss every aspect created, considered, and improved upon during development. This will be broken down into the following sections: UI design, Front end technical development, back end technical development, Database development, security and privacy development and issues that occurred during development.

UI design

This section will provide illustrations to demonstrate the initial designs created using wireframes compared to their final designs as well as an explanation as to why I chose the final design and how they meet my requirements. This section used the UI competitor design research to gain insight into the different design choices made. The UI design should consider **F3**, **F7** and **F8** of my functional requirements.

Index page

This is the landing page for all users and should provide a clear design and understanding of what the website intends to do. From this page users should be able to access all resources and operations. Information regarding the site should be clearly presented and informative.

Wireframe

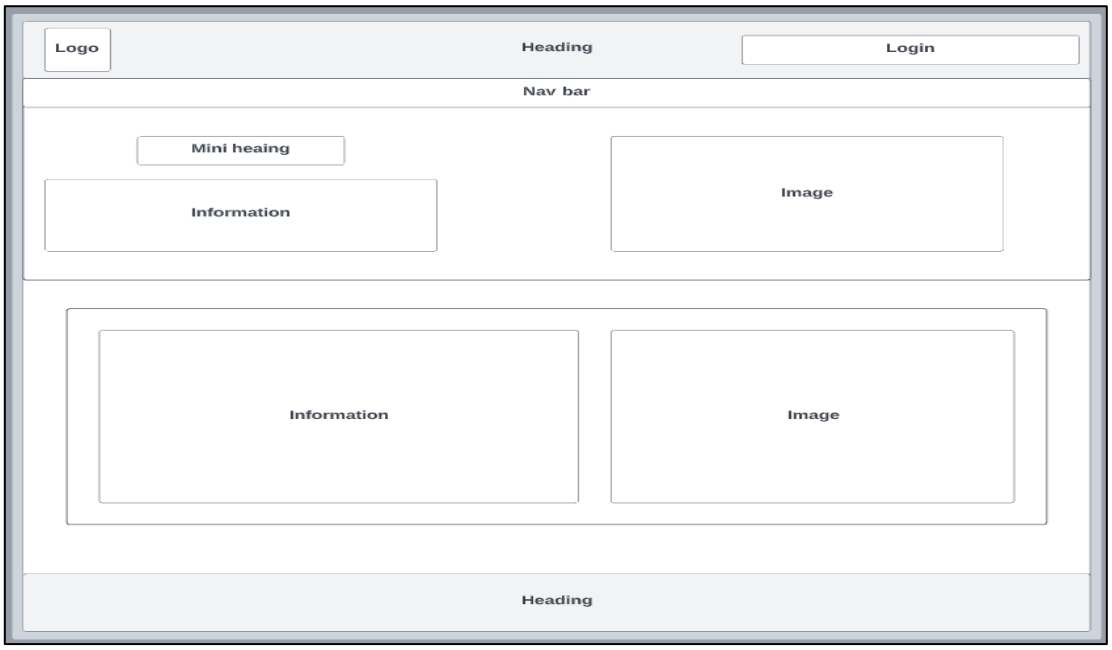


Figure 13 – index page wireframe.

Index page Final design

One change was adding a signup button as well as merging the navigation bar into the banner to provide a simple sleek look. The login-in design was created to provide straight forward access to user's accounts, commonly sites will redirect users to separate pages to perform different operations however, this provides another layer of difficulty for users who have disabilities, keeping in line with functional **F1** requirement (SAGE ,2022). Additionally, the size of buttons were made bigger to help with accessibility issues. Research suggested that font and font-size should be set to Tahoma with a minimum size of 16px as that is considered as an optimum style to clearly read from (Siteimprove, 2022) (ACCESSIBLEWEB, 2022). The colour scheme uses dark colours with bright white text, different colours have different affects on people. From research the colour blue is able to evoke feelings of reliability and people feel more secure when they see this (PUMPKIN, 2022). Due to the fact we are handling delicate client information I wanted to ensure that they had nothing to fear regarding the security of their data.

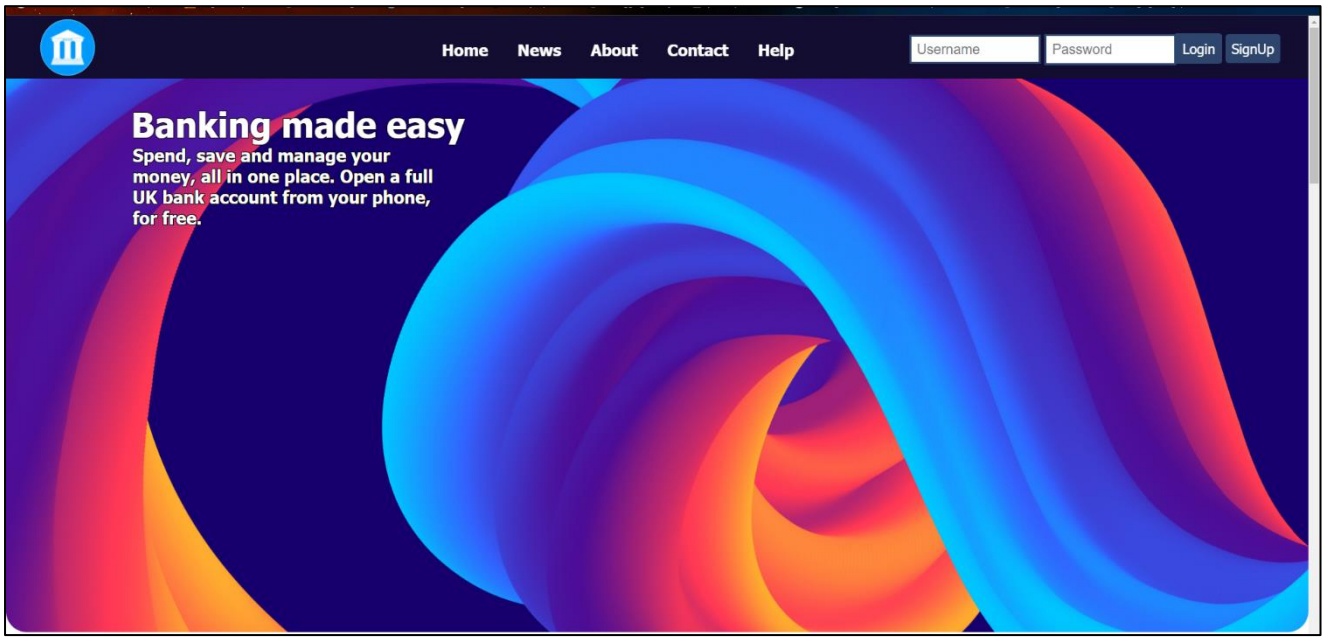


Figure 14 – index page final design

Sign up page

This page is responsible for creating user accounts from input data. It should provide clear instructions of the necessary data required to create an account. The idea is to allow users to create an account smoothly without any difficulties.

Wireframe

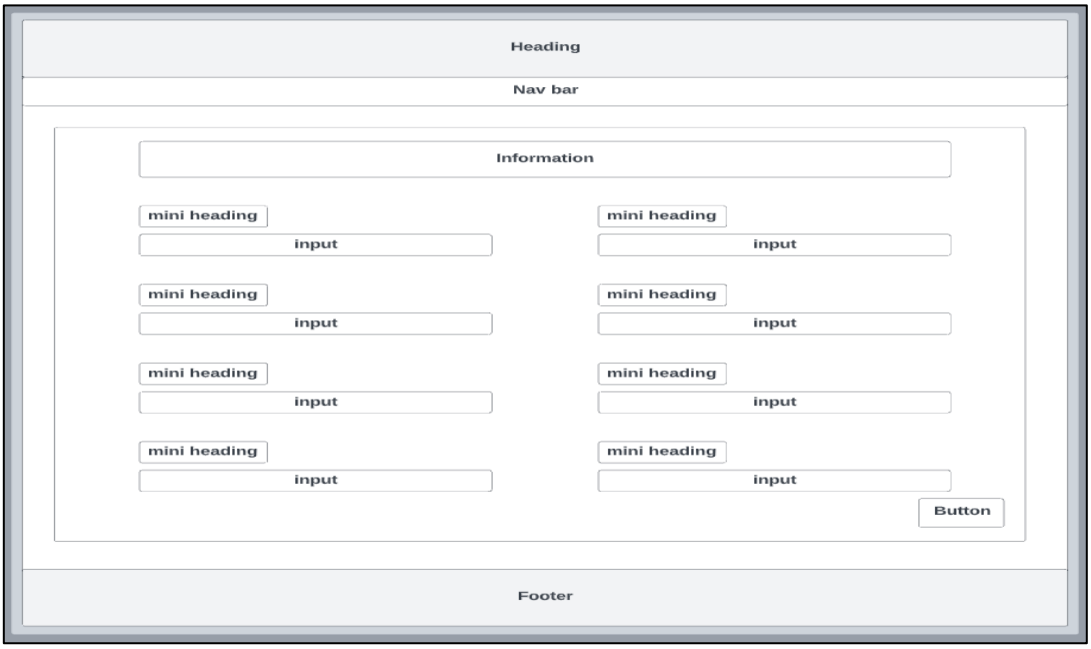


Figure 15 – sign up page wireframe

Sign up page Final design

Some changes were made to create a professional design and feel for users. One major change was having all your data displayed on one page, Instead I added the approach to change different categories of information. This was to prevent clutter on the screen as well as to make input fields bigger for accessibility issues. However, this does add the problem of having to switch back and forth to different categories if there is a problem. It was important to have a professional design as users need to be comfortable when entering their private data. Also, the colour blue is again used throughout to provide a feeling of reliability and security to the user. Research showed that adding images is also another approach to providing a profession design (**F8**) and reliability as most people look at images before reading anything (Logicdesign 2022).

Some UI designs that can help with security is to ensure that font is not too big or bold as sensitive data the user provides could be susceptible to shoulder surfing. One method I chose to prevent this is by making the background light and to make input values entered to also be light. At close distance they can be easily read however, it can be difficult for anyone else at a distance.

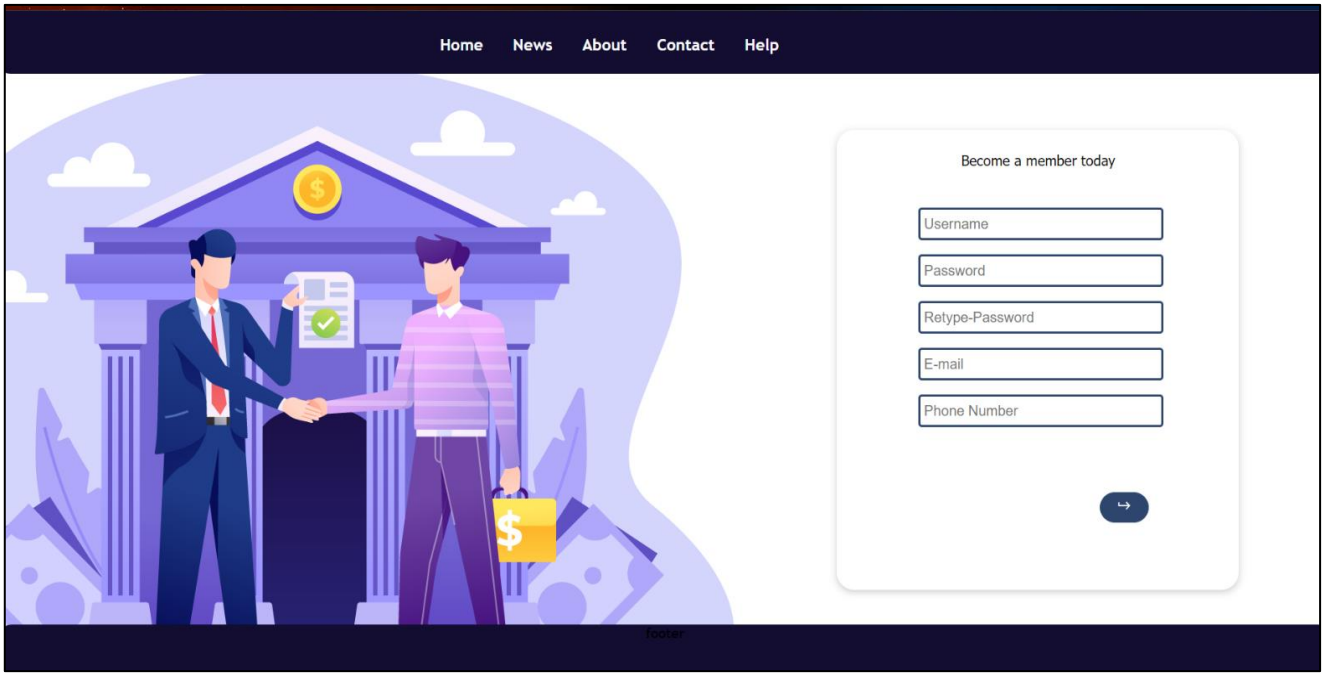


Figure 16 – final signup page design

Account page

This is the dashboard that provides different operations such as to update account information, creating savings widgets and accessing different bank accounts. It is able to provide visual charts and information from client banking data.

Wireframe

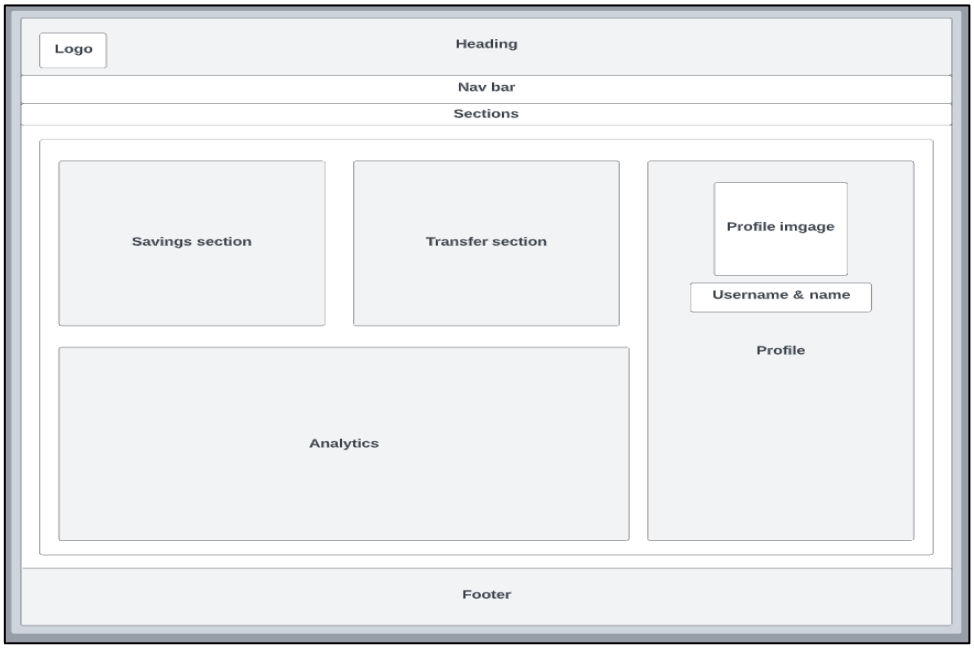


Figure 17 – account page wireframe

Overview

The overview section will provide all the general information about each account individually or together. Information such as balance, transaction history, current standing orders, and direct debits the account has. Also features a line chart which displays the transactions history visually to provide a better understanding.

Figure 18 – Final overview design

Savings

This section of the accounts page allows users to create savings widgets for saving up on different expenses. Users can transfer money from their accounts to gradually top up different widgets.

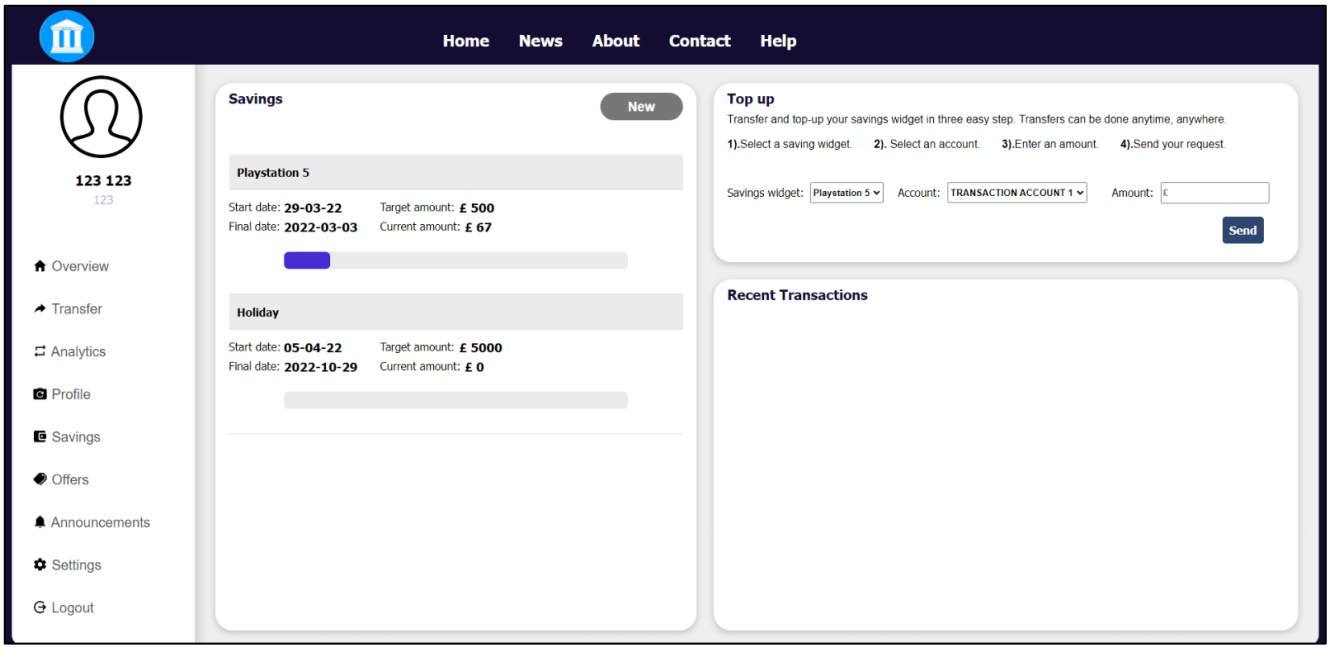


Figure 19 – Final savings design

Profile

The profile page is designed to gives users the ability to modify their account data. This allows them to have control over the amount of data they want to share.

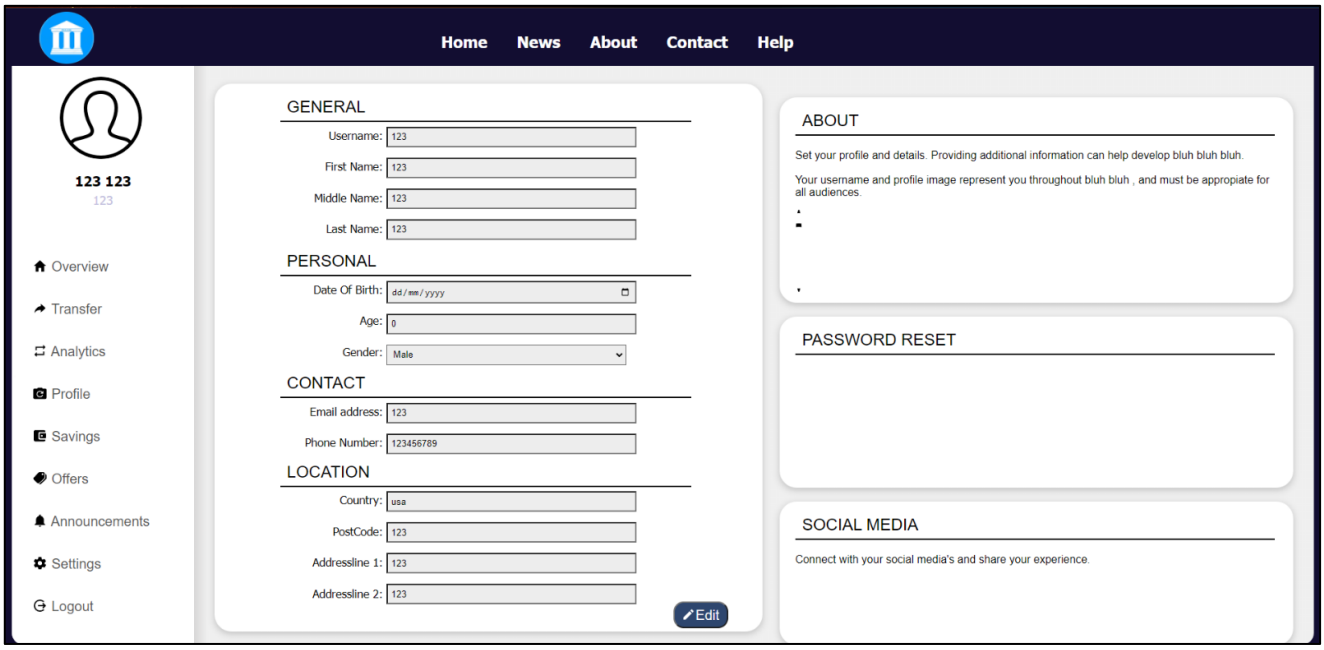


Figure 20 – Final profile design

Account page Final design

The reason I made a significant design change is that each operations can now be clearly sized to offer clarity when reading. This also offers a sleek minimalistic approach that separates each operation into their own areas, this was done to achieve my **F8** requirement. It clearly highlights what each operation does and makes reading information easier with the white background. I used (Monzo, 2022) to provide an idea of the type of design they offer. Buttons and text were made bigger and easier to access for accessibility issues. Using

Development

Signup Operation

Figure 21 displays a HTML form found on the sign-up page designed to POST input data from Signup.php to signup.inc.php file. Data is then retrieved and formatted to securely transmit to the database.

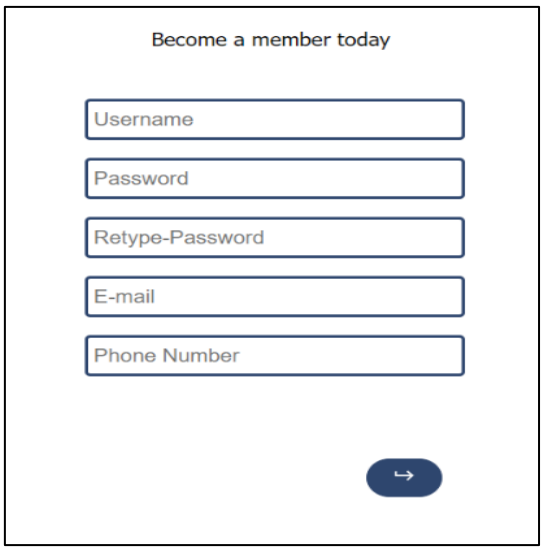


Figure 21 – Signup page HTML form

An important consideration I researched is the approach to preventing SQL injections attacks. This is important as my **F4** requirement is to introduce security prevention techniques. Data related attacks are a very common problem for vulnerable applications and are typically caused by inefficient input security and design (OWASP, 2022). Form validation is a security technique used to prevent SQL injections by whitelisting specific characters. ***Figures 22*** *and* ***23*** provide different code snippets of how this is achieved.

Figure 22 shows the process of validating data, which checks if data contains valid characters and not detrimental SQL code. This technique helps support my **Q6** requirement by maintaining the integrity of data.

```
}
else if(!preg_match("/^[a-zA-Z0-9]*$/", $username)){//Checks for a valid username
  header("Location: ../signup.php?error=invaliduid&email=".$email);
    exit();
 }
```

Figure 22 - Validating data received.

Figure 23 shows another approach of preventing an SQL injection which is the use of prepared statements. Prepared statements are parameterized SQL queries that separate SQL commands from data allowing for safe execution. Using the official PHP documentation (PHP.net, 2022), I was able to create a prepared statement using the SQL library.

```
//Creating prepeared statement to safely insert data. Preventing SQL code injection.
$sql = "INSERT INTO personalinformation (uidUsers, emailUsers, pwdUsers,firstname,middlename,lastname,phonenumber
VALUES (?,?,?,?,?,?,?,?,?,?,?,?,?,?)";
$stmt = mysqli_stmt_init($conn);
```

Figure 23 – Creating a prepared statement to safely send data to my database.

Update operation

The update functions allows users to control the data we store about them. The update functionality retrieves data from input fields and uses a prepared SQL update statement to send the data to the database. ***Figure 24*** shows a segment of the PHP code used to achieve this. This operation allows me to introduce the process of quickly and securely retrieving user data which is **Q1** of my requirements. The design of this operation was influenced by the Data *protection act*, the freedom to control data principle was considered, allowing users to change what we store about them.

```
//Inserts a data type (String)
mysqli_stmt_bind_param($stmt, "s", $username);
mysqli_stmt_execute($stmt);//Runs the username to see for match within database
mysqli_stmt_store_result($stmt);
$result = mysqli_stmt_num_rows($stmt); //Checks the amount of matches 0 meaning no match.
if ($result > 0) {
  header("Location: ../account.php?error=usertaken&email=".$email);
    exit();
}
else{
  $sql = "UPDATE personalinformation SET firstname = ?, firstname = ?, middlename = ?,lastname = ?,Gender = ?,ph
  $stmt = mysqli_stmt_init($conn);
  if (!mysqli_stmt_prepare($stmt, $sql)) {
    header("Location: ../account.php?error=sqlerror");
      exit();
  }
  else {
            mysqli_stmt_bind_param($stmt, "sssssisssi", $firstname,$firstname,$middlename,$lastname,$gender,$phc
            mysqli_stmt_execute($stmt);
            header("Location: ../account.php?update=success");
```

Figure 24 – Update operation PHP code

This operation allows me to achieve **F10** of my requirements which is introducing the operation for users to update their account data.

Login operation

This operation allows users to access their accounts using their credentials. ***Figure 25*** shows the process of verifying credentials entered by a user.

```
  $sql = "SELECT * FROM users WHERE uidUsers=?";
  $stmt = mysqli_stmt_init($conn);//Used to initialize an sql statement.
  if (!mysqli_stmt_prepare($stmt, $sql)) {
   header("Location: ../index.php?error=sqlerror");
     exit();
}
else {
  mysqli_stmt_bind_param($stmt, "ss", $uidUsers, $uidUsers);
  mysqli_stmt_execute($stmt);
  $result = mysqli_stmt_get_result($stmt); //Checks the amount of matches 0 meaning no match.
  if ($row = mysqli_fetch_assoc($result)) {
    $pwdcheck = password_verify($password, $row['pwdUsers']);
    if ($pwdcheck ==false) {
      header("Location: ../index.php?error=incorrctpassword");
        exit();
    }
    else if ($pwdcheck == true) {
       session_start();
       $_SESSION['userId'] = $row['idUsers'];
       $_SESSION['userUid'] = $row['uidUsers'];
       header("Location: ../index.php?login=success");
```

Figure 25 – Signing in operation found in sign.inc.php file.

Charts.js visualization

The use of charts for visualization of data was an important part of my application as I wanted to provide a clear way for users to interpret their data. Creating charts was done using chart.js (Chart.js, 2022) which is a JavaScript library that adds the ability to visually display data. ***Figure 26*** shows the approach to creating a chart and ***Figure 26*** shows the results of some of the charts created. Chart.js helped support **F2, F7** and **F8** of my requirements.

```
<script>
  const ctx2 = document.getElementById("myChart2").getContext("2d");

  const data = {
      labels: [],
      datasets: [{
      label: "Transactions",
      data: [],
      backgroundColor: "rgba(255, 99, 132, 0.2)",
      borderColor: "rgb(255, 99, 132)",
      borderWidth: 1
  }]
  };

  const myChart2 = new Chart(ctx2, {
   type: "line",
   data:data
});
</script>
```

Figure 26 - CHART.JS JavaScript code

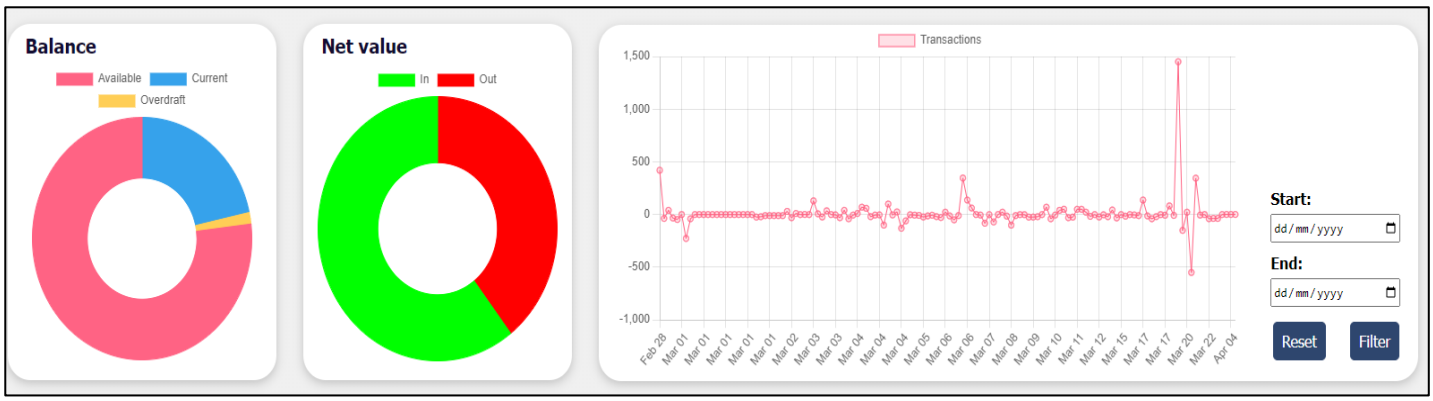


Figure 27 – Visualise charts found within my overview section

Using ajax & jQuery to retrieve data

Both ajax and jQuery were used to add additional behaviours and functionality. Ajax was used to introduce the unique approach of making POST requests to internal PHP documents without the use of a form, as I wanted to display and update data without having to refresh the webpage.

Figure 28 shows the use of jQuery to prevent the application from refreshing when loading in different information. The main line that prevents it is the **e.preventDefault()**, this cancels the event of refreshing.

Figure 29 shows the process of using ajax to POST data to an internal PHP file. Submitting data with HTML forms will refresh the page however, disabling the refresh event prevents data from being sent. In order to overcome this problem, we can use Ajax to load data to the desired PHP file. The process of setting this up is easy and only requires four parameters.

```
$(document).ready(function(){
    $("#accounts").load("AccountFunction.php");
     $("#accounts").submit(function(e) {
  e.preventDefault();
  });
```

Figure 28 – jQuery JavaScript code use to disable refresh event

```
$.ajax({
type: "POST",
url: "SavingsTopUpFunction.php",
data: { tdvalue: tuinput, tdvalue2: tuamount, tdvalue3: accountval },
success: function(response) {
        $("#topupmsgdisplay").html(response);
```

Figure 29 – ajax used to POST data

True layer Data API integration

The true layer data API is a critical service that will allow users to securely connect to their registered banks and safely retrieve their banking data. API development is a new aspect that I've have never learnt until now so using the official true layer documentation was a very helpful approach to better understanding how API's work (TrueLayer, 2022). Integration of the data API requires both code and configuration of their included true layer console which allows you to manage their various API's. ***Figures 30, 31*** and ***32*** provides insight of the preliminary tasks needed to configure the data API. Whereas ***figures 33, 34, 35, 36*** and ***37*** shows how to integrate the API into my node.js environment.

Figure 30 shows the client_id and client_sercret which are both used to identify and authenticate the client whenever a request to their API is made. These two values will be used later within the code integration to connect to the API.

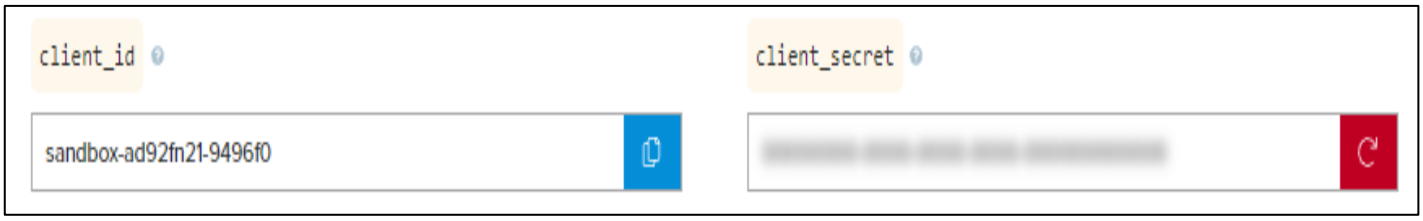


Figure 30 – true layer console client credentials

Figure 31 shows the creation of a redirect link, this is used to provide a location to redirect users after they have finished the verification and authentication process. This redirect route will perform all the operations such creating a token, retrieving, and formatting data and storing it to the database. This route was created using the express framework, it performs a GET request to the API using the token generated which grants us access to the user's data.

Figure 31 – Redirect URL

Figure 32 shows the authentication link used for clients to connect to the API interface . The authentication link can be configured to display specific details such as the number of banks shown as well as the amount of information we can collect.

Figure 32 – Authentication link

Figure 33 shows the API integration using the previous data created on the console. We can create a new client object using the id and secret generated earlier, this informs the API who is accessing their interface. We also have the redirect link which sends users to a different route after they complete the banking process.

```
const redirect_uri = "https://jb1828.brighton.domains/FinalYearProject/truelayer-redirect";
// Create TrueLayer client instance
const client = new AuthAPIClient({
        client_id: "ad92fn21-9496f0",
        client_secret: "bb172316-cfe5-4731-9e0c-af6d58b3136c"
});
```

Figure 33 – JavaScript API varibles

Figure 34 shows the start of retrieving data, once the user finishes requesting data, we retrieve the code generated by the user after successfully being authenticated. We retrieve this code by making a request to the API using one of the prebuilt functions. The function takes a client object to verify who is accessing the token followed by the provided redirect we are using and the code. The token provides access to all the necessary banking data.

```
const code = req.query.code;
const tokens = await client.exchangeCodeForToken(redirect_uri, code);
```

Figure 24 – generating an access token

We can use the object DataAPIClient to make requests for certain types of data. ***Figure 35*** shows the server requesting the balance data. This requires the access token generated previously and additional information such as the account number we are trying to get the balance for.

```
const balance = await DataAPIClient.getBalance(tokens.access_token, accounts.results[x].account_id);
```

Figure 35 – True layer balance fetching method

Figure 36 shows how we format JSON data. When retrieving data, it comes as pure JSON in order to correctly store it we need to convert it into a JSON object. This process requires the use of the JSON library, with this we can convert it into a string to make it readable and then into an object to make it manipulable.

```
var User_balance = JSON.stringify(balance,null, 2);
const Balance_obj = JSON.parse(User_balance);
```

Figure 36 – Json string and object

Figure 37 shows the iteration of JSON balance data being stored to the database. When JSON data becomes an object, it acts as an array and therefore you can perform array operations like indexing. In the example below we search the database looking for the desired user and then iterate through the JSON array storing data into their necessary rows. With the integration of the REST API this this all

```
var sql = "SELECT * FROM Balance WHERE account_id= '"+Account_id+"' AND DataToken = '"+T+"';";
 con.query(sql,function(err, row){
   if(row && row.length){
   }else{
       const Currency = Balance_obj.results[0].currency;
       const available = Balance_obj.results[0].available;
       const Current = Balance_obj.results[0].current;
       const overdraft = Balance_obj.results[0].overdraft;
       var sql = "INSERT into Balance (DataToken,Account_id,currency, available, current, overdraft)
    con.query(sql);
```

Figure 37 – Storing balance banking data JavaScript operations

Security and privacy designs and implementation

Below in this section I discuss the security and privacy methods I introduced to my application. My CI608 secure networks module helped provide an understanding into some of the listed security methods below. This module was useful to provide insight into how to protect my application from detrimental threats, how to securely configure data for transmission and the use of encryption.

Prepared statements

A previous mentioned safety technique is the use of prepared statements to format SQL queries to not execute detrimental SQL. This was introduced to prevent SQL injection attacks and maintaining database security. **Figure 38** shows an example of a prepared statement

```
//Creating prepeared statement to safely insert data. Preventing SQL code injection.
$sql = "INSERT INTO personalinformation (uidUsers, emailUsers, pwdUsers,firstname,middlename,lastname,phonenumber
VALUES (?,?,?,?,?,?,?,?,?,?,?,?,?,?)";
$stmt = mysqli_stmt_init($conn);
```

Figure 38 – prepared statement

Hash function

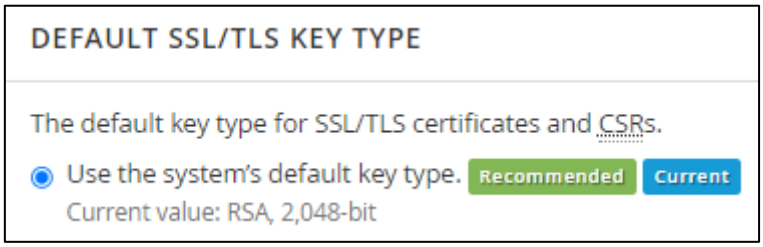
From previous lectures within the CI608 module I gained an understanding into the different types of hashing algorithms, which helped influence the decision to finding a suitable algorithm. Bcrypt is a 128-bit hashing algorithm that is based on an improved version of the Blowfish algorithm (Bcrypt, 2022). The reason I chose this algorithm is that it allows me to incorporate safe transmission of data on the front and back end of my application. There are also currently no known security issues with it and it is extremely effective against brute force and rainbow attacks. However, it only utilises 128-bit salt for random string generation which is weaker compared to a 256-bit algorithm. Unfortunately, I was not able to find an algorithm that had 256-bits and was also supported on both JavaScript and PHP. This was important as they had to decrypt the hashed passwords on both ends. ***Figure 39*** shows the process of hashing passwords on PHP to safely transmit sensitive data. This algorithm will prevent threats from obtaining plaintext data, which is in line with **F4, Q6** and **Q7** of my requirements. With the signup functionality working effectively this helps me achieve my **F9** functional requirement.

```
//Hashin passwords to prevent plaintext interception when transmitting data.
$hash = password_hash($password, PASSWORD_DEFAULT);//Hashes the password
mysqli_stmt_bind_param($stmt, "ssssssisssisss", $username,$email,$hash,$firstname,$middlename,$lastname,$phonenumber,$country,$da
mysqli_stmt_execute($stmt);
$sql = "INSERT INTO users (uidUsers, emailUsers, pwdUsers) VALUES (?,?,?)";//prepeared statemnet
mysqli_stmt_bind_param($stmt, "sss", $username,$email,$hash);
$stmt = mysqli_stmt_init($conn);
if (!mysqli_stmt_prepare($stmt, $sql)) {
```

Figure 39 – Hashing sensitive passwords for transmission.

SSL Encryption

SSL Encryption is another technique learnt within the CI608 module that allowed me to secure pages so that information such as credentials, account numbers, etc are sent encrypted. This process can be achieved by using Let's encrypt, an application on cPanel which allows you to configure and set an RSA encryption type. ***Figure 40*** shows the default key set which is RSA, an asymmetric encryption method which uses a pair of generated keys (a public key and private key). The way this encryption method works is by encrypting data before transmission by using the public key and then decrypting the data with the private key once it has arrived at its destination. ***Figure 41*** is a UML diagram illustrating the process of RSA encryption. Additionally, True Layers API uses AES encryption which is a popular industry standard 256-bit encryption algorithm, which means that it helps achieve secure transmission of banking data meeting **Q1** and **Q5** of my non-functional requirements



.

Figure 40 – SSL keys

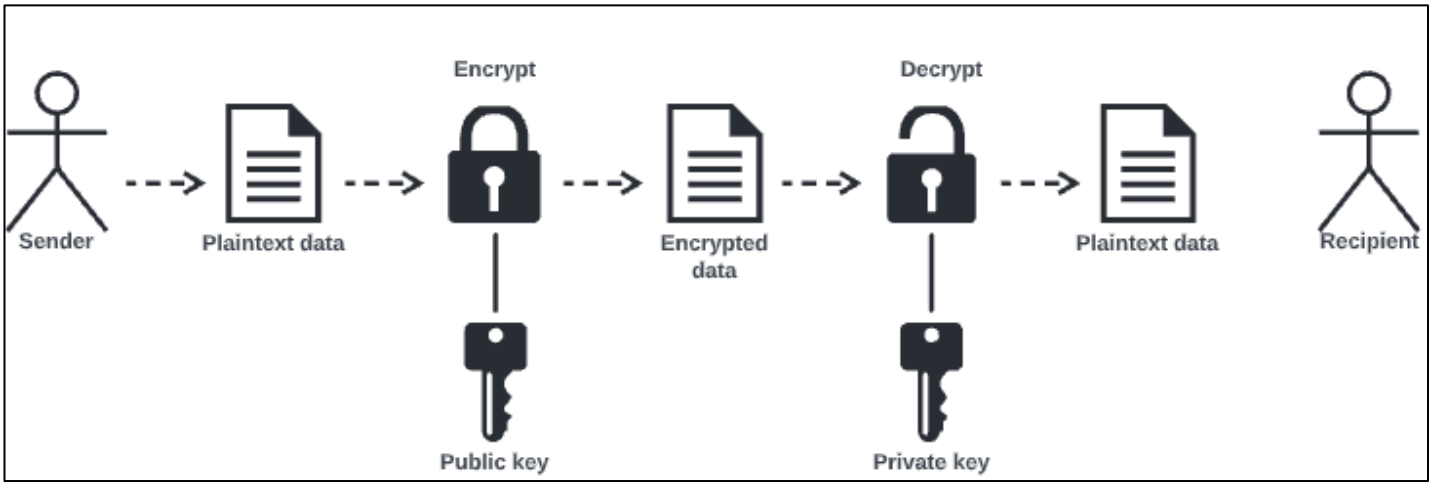


Figure 41 – Encryption process diagram

Testing

This section shows the various testing methods used to check critical operations are correctly working. This is important as this will determine whether my requirements are met.

W3C HTML & CSS validator

Below ***Figures 42, 43*** and ***44***, show the HTML validation test being performed on all my pages. This is used to check for any HTML errors by scanning the source code. This is a useful testing tool to uncover hidden problems, which helps me meet my functional **F7, F8, F9, F10** and **F11** requirements. This ensures that critical operations are not affected by inefficient HTML development.

Below ***Figures 45, 46*** and ***47*** show all three of my pages passing the CSS validation test. This prevents conflict with other elements and helps achieve **F7** and **F8** of my functional requirements ensures that all the styling for critical elements are accessible and clear.

Figure 42 – Index page HTML validated

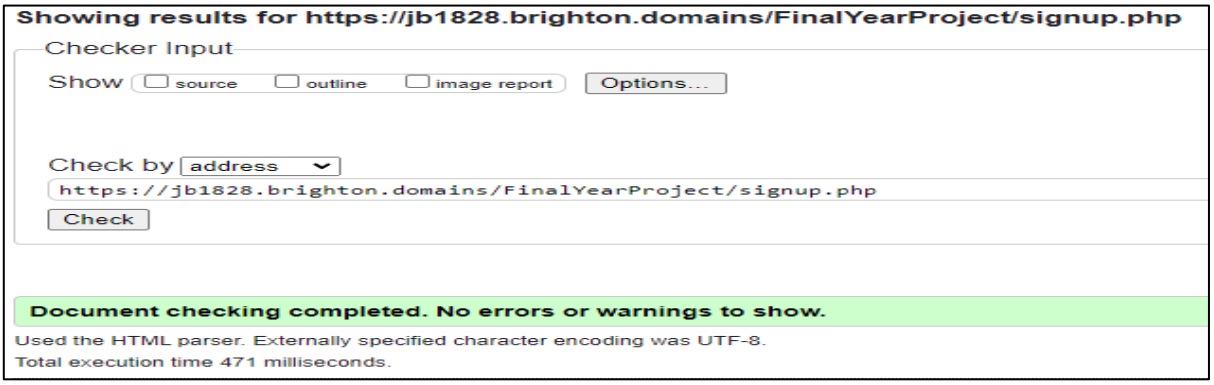


Figure 43 – Signup page HTML validated

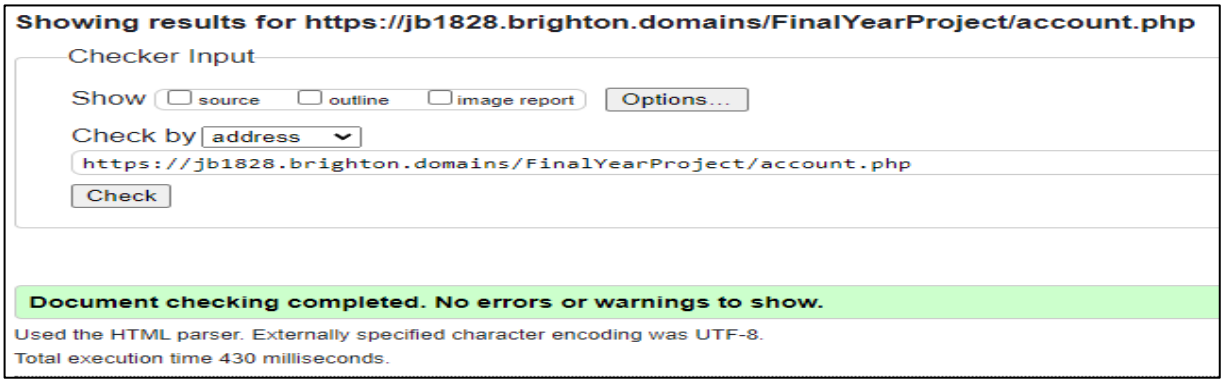


Figure 44 - – Account page HTML validated

Figure 45 – Index page CSS validated

Figure 46 – Signup page CSS validated

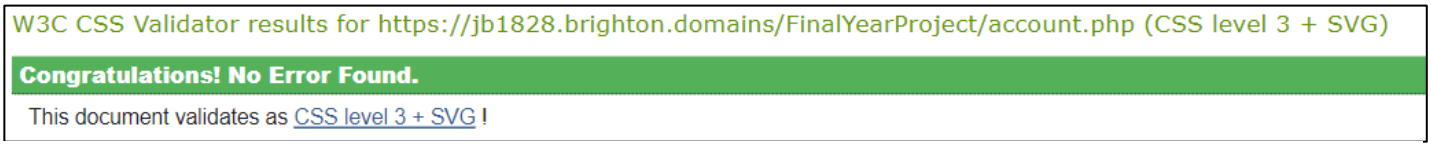


Figure 47 – Account page CSS validated

Performance testing

Using WebPageTest.org I was able to test the performance of my application to see the response time. This was important as one of my requirements is to be able to process client requests within 5 seconds **Q3**. With ***figure 48*** I was able to run a metrics test to obtain useful information such as index speed which is the render time of loading my pages. It takes 2 seconds to load and render the content to a user's screen. With these results this helps support the achievement of my **Q3** requirement.

Figure 48 – performance metrics

Security scan

I was able to perform a security scan with a site called Snyk which is a free open-source vulnerability site scanner. This was used to uncover any detrimental code or unsafe elements . However, with the security research performed I was able to use it to implement enough techniques to obtain a high security score. ***Figure 49*** can support the claim of successfully achieving **F4** and **Q6** which are both security related requirements.

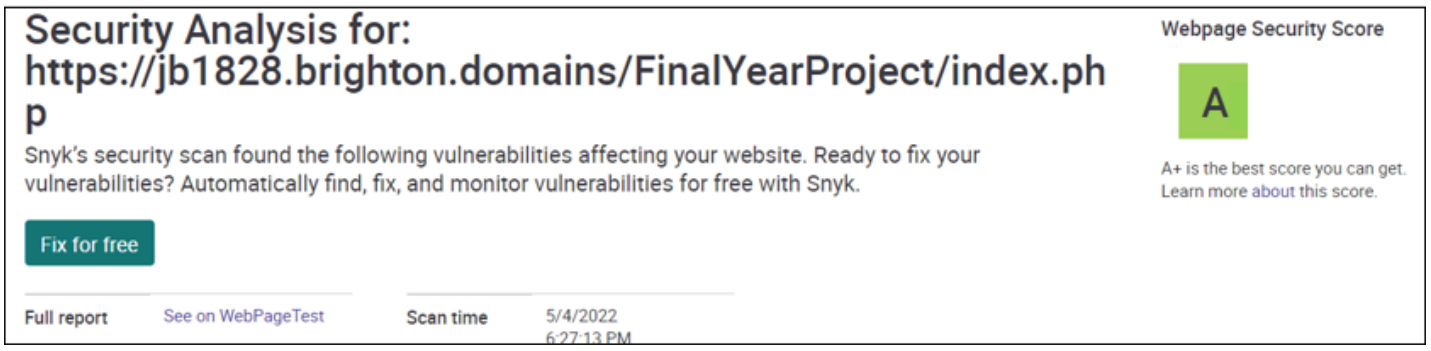


Figure 49 – security scan result

Responsiveness test

Am I Responsive ?, is a site that will allow me to emulate different pages of my application through different devices. Below ***figures 50*** *and* ***51*** display my index and signup page on different device. Unfortunately, I was not able to display my account page as I am unable to access it without having to sign in. However, a solution to this is to use my phone and tablet to confirm its responsiveness. This test concluded my application is accessible through different devices, support both **F6** and **Q4** of my requirements.

Figure 50 – Index page responsive test

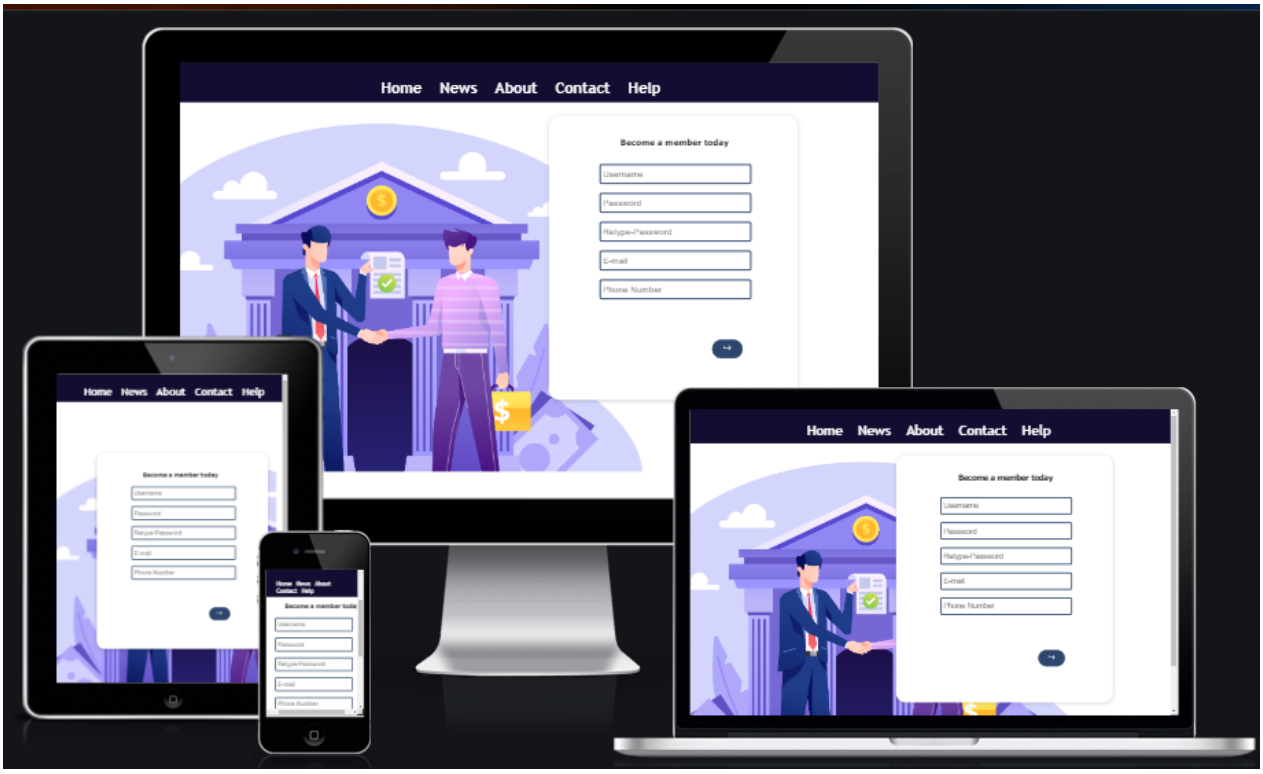


Figure 51 – Signup page responsive test

PHP testing

Below are various PHPUnit tests that were used to test the different front-end operations. Below ***table 6*** provides a test required matrix which details the tests being performed and how they will help achieve my requirements.

ID	Test	Description	How does this contribute to my requirements	Result
1	Database connection	Figure 52, will test if I am able to connect to my database via my PHP SQL connector.	As a failed test will mean some operations such as **F9,F10** and **F11** requirements will not work correctly and so this will affect my requirements from being met **F2, Q1** and **Q3**.	**Pass**
2	Database query	Figure 53, will test that queries are correctly structured to retrieve data.	This ensures that **F9**, **F10**, **F11, Q1** and **Q2** requirements are met allowing data related operations to be performed correctly.	**Pass**
3	Signup verification	Figure 54, this will test that my sign up verification operation correctly works.	This test will ensure that my **F9** requirement is met, as the this will provide evidence that my registration operation work correctly.	**Pass**
4	Sign in operation	***Figure 55***, checks that it doesn't store matching user data, by trying to store existing data.	For if this problem were to occur this would mean that **F2, F5, F9** and **F11** of my functional requirements would not be implemented which are user related operations.	**Pass**

Table 6 – PHPUnit tests

```php
public function testDatabseConnection(){
    $servername = "localhost:3306";
    $dbUsername = "jb1828_root";
    $dbPassword = "A1B2C3a1b2c3";
    $dbName = "jb1828_loginsystem";
    $conn = mysqli_connect($servername ,$dbUsername, $dbPassword, $dbName);
    $boo = true;
    if (!$conn) {
        $boo = false;
    }
    $this->assertEquals(true, $boo);
}
```

Figure 52 - Database connection test

```php
public function testDatabseQuery(){
    $boo = true;    $sql = "SELECT * FROM personalinformation";
    $result = mysqli_query($conn, $sql);
    $resultCheck = mysqli_num_rows($result);
    $this->assertGreaterThan(0, $resultCheck);
}
```

Figure 53 - Database query test

```php
public function testSignUpVerification(){
    $username = "JohnDoue";
    $email = "JohnDoe@Gmial.com";
    $password = "pasword123";
    $paswordRepeat = "pasword123";
    $bool = true;

    if (empty($username) || empty($email) ||  empty($password)|| empty($paswordRepeat)) {
        $bool = false;
    }else if (!filter_var($email.FILTER_VALIDATE_EMAIL) && !preg_match("/^[a-zA-Z0-9]*$/", $username)) {
        $bool = false;
    }else if(!filter_var($email.FILTER_VALIDATE_EMAIL)){
        $bool = false;
    }else if(!preg_match("/^[a-zA-Z0-9]*$/", $username)){
        $bool = false;
    }else if ($password !== $paswordRepeat) {
        $bool = false;
    }
    $this->assertEquals(true, $bool);
}
```

*Figure 54 - **Signup verification test***

```php
public function testSignInOperation(){
    $bool = true;

    $mailuid = "john";
    $password = "Doe";

    if (empty($mailuid) || empty($password) ) {
        $bool = false;
    }else {
      $sql = "SELECT * FROM users WHERE uidUsers=? OR emailUsers=?";
      $stmt = mysqli_stmt_init($conn);//Used to initialize an sql statement.
      if (!mysqli_stmt_prepare($stmt, $sql)) {
        $bool = false;
     }else{
        mysqli_stmt_bind_param($stmt, "ss", $mailuid, $mailuid);
        mysqli_stmt_execute($stmt);
        $result = mysqli_stmt_get_result($stmt); //Checks the amount of matches 0 meaning no match.
        if ($row = mysqli_fetch_assoc($result)) {
          $pwdcheck = password_verify($password, $row['pwdUsers']);
          if ($pwdcheck ==false) {
             $bool = false;
          }
          $this->assertEquals(true, $bool);
        }
```

*Figure 55 - **Sign in operation test***

```
PS C:\Users\joeba\Desktop\FinalYearProject> ./vendor/bin/phpunit
PHPUnit 9.5.20 #StandWithUkraine

....                                                                4 / 4 (100%)

Time: 00:00.137, Memory: 6.00 MB
```

Figure 56 – PHPUnit test results

Node.js Jest unit testing

Below are various Jest unit tests that were used to verify that critical functions were performing correctly within the express backend server. Below ***table 7*** provides a test required matrix which details the tests being performed and how they will help achieve my requirements. The design of each test used my requirements as focus points to make sure that critical areas are working correctly and being delivered.

ID	Test	Description	How does this contribute to my requirements	Result
1	Route	***Figure 57***, tests the routes used within my express server are accessible by checking for a 200-status code.	This verifies if critical operations are being performed. If URL routes are not accessible this suggests that they are not performing correctly. This test ensures that requirements **F1, F5** and **F10**, are which are critical operations are performing correctly. Also checks requirements **Q1** and **Q3** are achieved as data is available and the client request is processed efficiently.	**Pass**
2	Database config	***Figure 58***, performs a test that checks the database configuration information is correct.	If a connection is not established, then certain operations are not working. This affects requirements **F2,F9,F10,F11** which are dependent on database access in order to functions.	**Pass**
3	Database connection	***Figure 59,*** verifies that the connection to the database was successful.	The ability to determine if a connection to the database is crucial. If a connection is not established then requirements **F2,F9,F10,F11, Q1 and Q3** will not be achieved. This test will verify if a connection is made. This will allow operations to operate correctly, ensuring the success of my requirements.	**Pass**
4	Database query	***Figure 60***, Tests if data can be retrieved from the database.	This test will check that data can be retrieved successfully without problems. This will help contribute to the my **F9**, **F10**, **F11, Q1** and **Q2** requirements allowing data related operations to be performed.	**Pass**
5	Hash function encryption	***Figure 61***, Tests to see if strings are encrypted into unreadable text.	This test verifies that my hash function is correctly encrypting data. As one of my non-functional requirements is to stored hashed passwords **Q7.**	**Pass**
6	Hash function decryption	***Figure 62***, Tests to see if strings are decrypted correctly.	This test is to make sure that my hash function will correctly decrypt hashed values ensuring that **Q7** will be achieved.	**Pass**

Table 7– Jest unit tests

```
test("1). Route test", async() =>{
    const Signintemp = await request(app).get("/FinalYearProject/Signintemp").expect(200);
    const TruelayerRedirect = await request(app).get("/FinalYearProject/truelayer-redirect").expect(200);
    const Display = await request(app).get("/FinalYearProject/display").expect(200);
```

Figure 57 – Route testing

```
test("2).Database config value test", () =>{
    expect(value.Con.config.host).toEqual("localhost");
    expect(value.Con.config.database).toEqual("jb1828_loginsystem");
    expect(value.Con.config.user).toEqual("jb1828_root");
    expect(value.Con.config.password).toEqual("A1B2C3a1b2c3");
```

Figure 58 - Database configuration test

```
test("3). Database connection test", async() =>{
value.con.connect(function (err) {
        if (err) throw err;
```

Figure 59 - Database connection test

```
test("5). Hash Function encryption test", () =>{
 const hash = bcrypt.hashSync(Plaintext, 10);
 expect(hash).not.toBe(Plaintext);
```

Figure 60 - Database query test

```
test("6). Hash Function decryption test", () =>{
const hash = bcrypt.hashSync(Plaintext, 10);
bcrypt.compare(Plaintext , hash, function(err, result) {
        if (result == true) throw err;
```

Figure 61 - Hash function encryption test

```
PASS ./app.test.js
  √ Route test (458 ms)
  √ Database config value test (11 ms)
  √ Database coneetion test (5 ms)
  √ Database coneetion test (1 ms)
  √ Hash Function encryption test (84 ms)
  √ Hash Function decryption test (69 ms)
```

Figure 62 – Jest unit test results

Development Implications

Below discusses some implications I faced during development of my application. These are problems I found that had negatively impacted the development timeline, preventing me from working on the project for 1-3 days.

Accessing and sharing data between routes and True Layer functions

A problem occurred when integrating the True Layer data API, which prevented sharing data from routes into True Layer functions. The problem was caused by the True Layer library, as some functions did not support the use of calling external variables, this was a security implementation that True Layer added. It treated external data as potential detrimental code that could interfere with the process of retrieving sensitive data.

Only local data can be used which can be a problem when a value is needed from another route, a solution was found while researching coding forums, it was found that using a temporary cookie to transfer data was an effective way to overcome this problem. By using a cookie to store and transfer data I was able to solve the problem.

Database Configuration problem

When configuring my database in early development I faced a problem that affected my SQL operations which meant that data was not being stored. The problem was how I configured my table and was solved when a meeting with my supervisor. The tables were referenced to a primary key located on the personal information table this meant that tables that had a foreign key could not store data unless it already existed within the parent table.

This meant I had to reorder the operation of storing user data in the front-end. As long as the personal information table received data before the users table there would not be any problems.

True layer Data API became unavailable due to maintenance

While testing I experienced a problem of not being able to progress through the data API interface. The reason was due to True layer performing maintenance as seen *in* **figure 63** however, this maintenance update was only meant to last for an hour but instead there was another internal problem with the API. ***Figure 64*** shows me messaging support and uncovering the same problem on their end. This problem persisted for four days until I was notified that the problem had been resolved. This was an unexpected risk that wasn't considered within my risk assessment as the potential problem of external services being unavailable can have caused a long

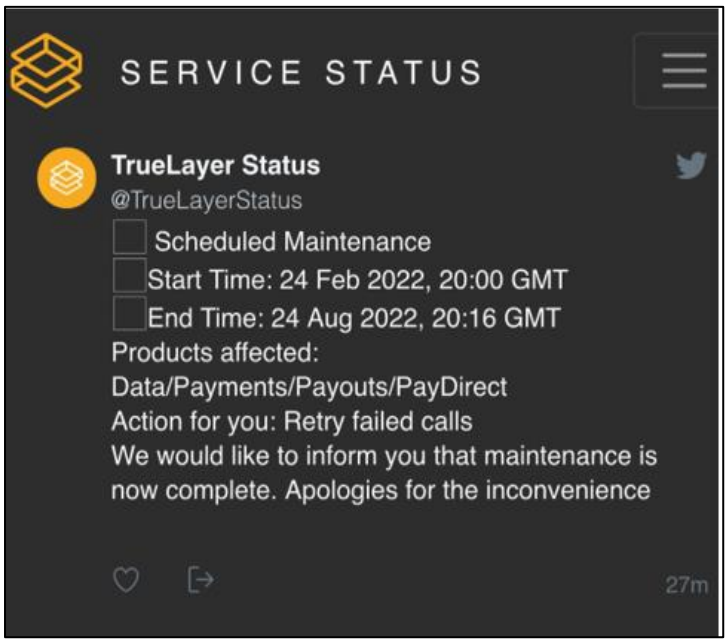


Figure 63 – twitter service status tweet

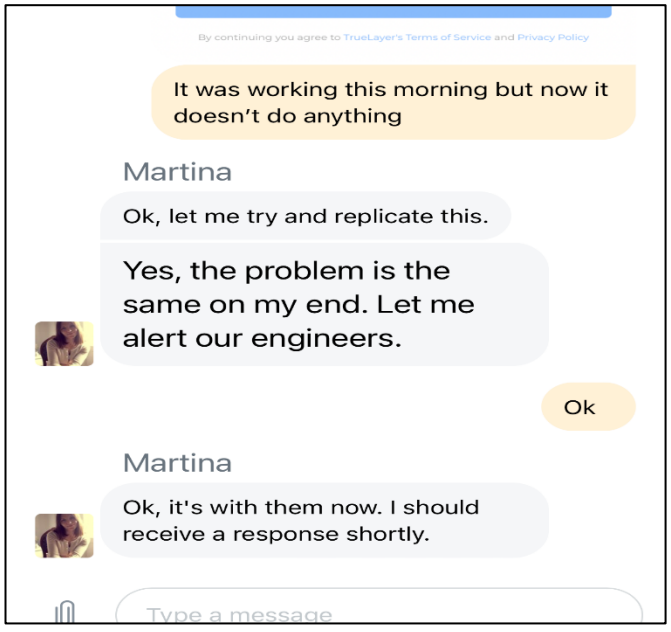


Figure 64 – True Layer customer support

Critical Review

This section explores successful areas I found within the project as well as potential areas of improvement, areas that I leant and different approaches for future projects.

Areas of success

The chosen methodology was a successful approach to developing my application as I was able to fully deliver a functional application that met my requirements, within a strict set timeline using a waterfall methodology. Using a hybrid approach meant that I was able to fully utilize the useful features of an agile methodology allowing me to be flexible with adding extended features.

Another area of success that I achieved was the implementation of my functional and non-functional requirements throughout development. This meant that I was able to provide critical functional operations such as registration for users (**F9**), updating and or adding users' data to accounts (**F10**) and a log-in for users (**F11**) etc, this meant that collectively by introducing requirements I was able to provide the promised **D1** deliverable. By using the waterfall methodology to create a set of requirements to follow at the start, it allowed me to focus on important areas of development without deviating.

The final UI design of the application is something that I'm very happy with as styling websites is not my strong suit, and so to have the final design look professional and tidy is a success. The process of designing started with plain wireframes as I didn't have an initial idea of what I wanted my site to look like. But after researching how other sites looked, this influenced me to use a minimalistic design which is aesthetically pleasing.

The development tools I decided to use were well suited to achieving and implementing requirements, made possible from different areas of the application. Specifically, they were useful for implementation of security and privacy techniques. This is another area of success as security was a major aspect of my requirements that I have effectively introduced within my application.

Finally, learning API development was difficult to understand with no prior knowledge however, I was able to overcome implications in order to successfully integrate a functional REST API for my project.

Areas of improvement

To improve overall I would familiarise myself with all tools to gain a more furlough understanding of how they function. Due to my lack of experience of using Tello it was a learning curve to prioritise which important operations needed to be completed regularly. However, I mitigated this issue by checking management tools consistently.

Another area of development I would like to improve is the process of condensing my PHP development into less files and remove code repetition. Primarily, my PHP operations used repetitive blocks of code, this approach could have been easily solved if I were to put the required code into its own file and then called it instead of putting it into every PHP file. This was done to save time as I treated it as a template for each data retrieval operation.

This is also similar to some of my HTML & CSS code, consistency can be improved as working on a project for a long time can get repetitive in terms of styling. I would use short cuts to quickly design different pages, which is functional in the short term but, hinders the responsiveness meaning I would back track and alter the native display before doing mobile display.

My final improvement is in regard to my own time management skills and the need adapt to be more flexible when dealing with multiple assignments. I learnt when it was necessary to prioritise certain work, in semester two I equally balance the workload between different assignments.

What have I learnt

During the development of my project, I learnt various topics but more specifically I gained a better insight into API's. Learning and using APIs for this project has been a new experience as I have never used one before. This was an interesting topic to learn and was highly beneficial as now I am confident developing RESTful APIs in other projects.

Express is a Node.js framework that I learnt to develop my backend server. This was a useful and easy tool to use, it is something I've never used before for web development, but it made the process of developing quick as it doesn't take a lot to learn how the library works. There is plenty of documentation to read to better understand how to perform certain functions such as creating different servers within a Node application. This is also another tool I will use for future projects.

One thing I learnt that I could take away with me is the importance of project management and how beneficial it is. Most projects I do in my own leisure time, so I code as I go, as I typically don't have a plan. However, for my final year project I followed the process of correctly project managing, this taught me how effective it is to plan and manage my time as it made developing straight forward and less stressful, whilst it is time consuming it makes the process easier of overall.

How would I approach future projects

In future I would explore using another public environment that would provide me with more control. Using cPanel through Brighton domains made development harder as certain resources required admin privileges. An example was database configuration due to the fact that I had low level privilege I wasn't able to configure a database as much as I would as an administrator. Another problem with cPanel is it doesn't provide the basic features that an editor would provide. This made testing difficult as cPanel doesn't provide a console interface to display data from your applications like most smart editors and so this made testing hard as I was not able to see if they would pass or fail. I would instead use another public environment that I could have full control over.

Another change would be to implement more than one API service as this provides the ability to not be reliant on one service. As previously mentioned, one of my implications was that the True Layer data API was unavailable for a certain time. By introducing another API service, I could switch between the two so that I could provide a constantly available service.

My final change that I would make is for future web development, to create websites from a single node server instead of having two separate servers. This condenses everything into one location making managing and developing significantly easier. This process would make accessing everything easy and I wouldn't need to incorporate as many development tools as I did.

Research

Open banking

This section of research was collected to provide insight into what open banking is, how it works and the potential risks that can occur.

What is open banking ?

These are financial services that implement data collecting APIs to allow third party applications to access consumer data from official financial institutions. This means that they can access consumer banking details and other financial data (Investopedia, 2022).

How does it work ?

These services work with the use of Data collecting API's these are integrated interfaces that provides a secure way of accessing and exposing data (Finextra, 2022).

What are some of the risks with open banking ?

The main risk that can occur from these types of services is the integrity of data being affected. Unreliable services can exploit consumers and steal or modify data which is unlawful. Additionally, data can be prone to interceptions like man in the middle attacks if the development of the API is not secure (Teradata!, 2022).

Popular services like Monzo and starling bank are both approved by FSCS (Financial Services Compensation Scheme), who ensure these services are reliable and can protect customers data.

Influence

The risk section of this research provides awareness of the potential threats and problems that can arise with open banking applications. This influenced me to make sure that I implemented a substantial amount of security and privacy techniques to mitigate any threat. Primarily focusing on maintaining the confidentiality and integrity of data would be a number one priority. With this can I readjust my requirements to introduce data privacy and security. By introducing different security methods such as encryption and hashing sensitive data I was able to achieve a high level of security.

Competitor UI design

This section looked into the design and layout of both Monzo and Starling.

Web design

Basic and clean navigation bar that has a logo and only necessary links. Background colour of banner makes it clear to create text. Text size and colour works perfectly well with the dark blue to offer clarity.

Sign up button which accessible through the webpage (Fixed position). However , no sign in button for those who are already a member. Clear and visual to all. Big to offer better accessibility for those.

A clear brief display of what they are trying to achieve and sell to consumers. Again, text is clear and stand out with the blue background.

Visualizations used to display what the application looks like. Image informs of mobile accessibility, which means that the site is responsive.

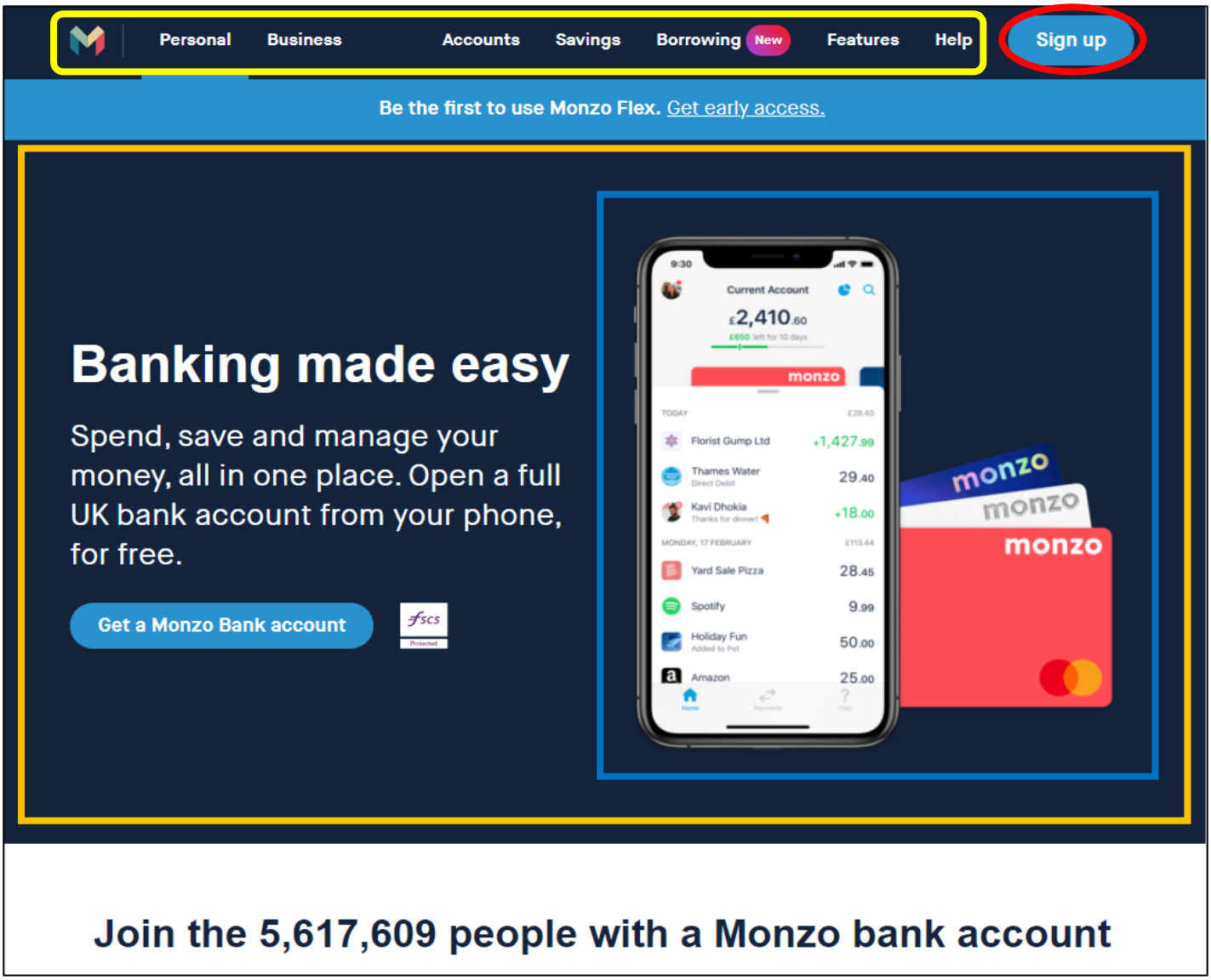


Figure 65 – Monzo's landing page

Information displayed and maintained within different sections. Shows and explains what the application provides. Provides a smart and professional look to the application.

Use of images to help those who may be illiterate and or struggle with reading visualise the features they offer. Makes understanding information easier and how it can be accessed through the mobile application.

Clear and concise explanation of what you are able to do and what they intend this feature to do.

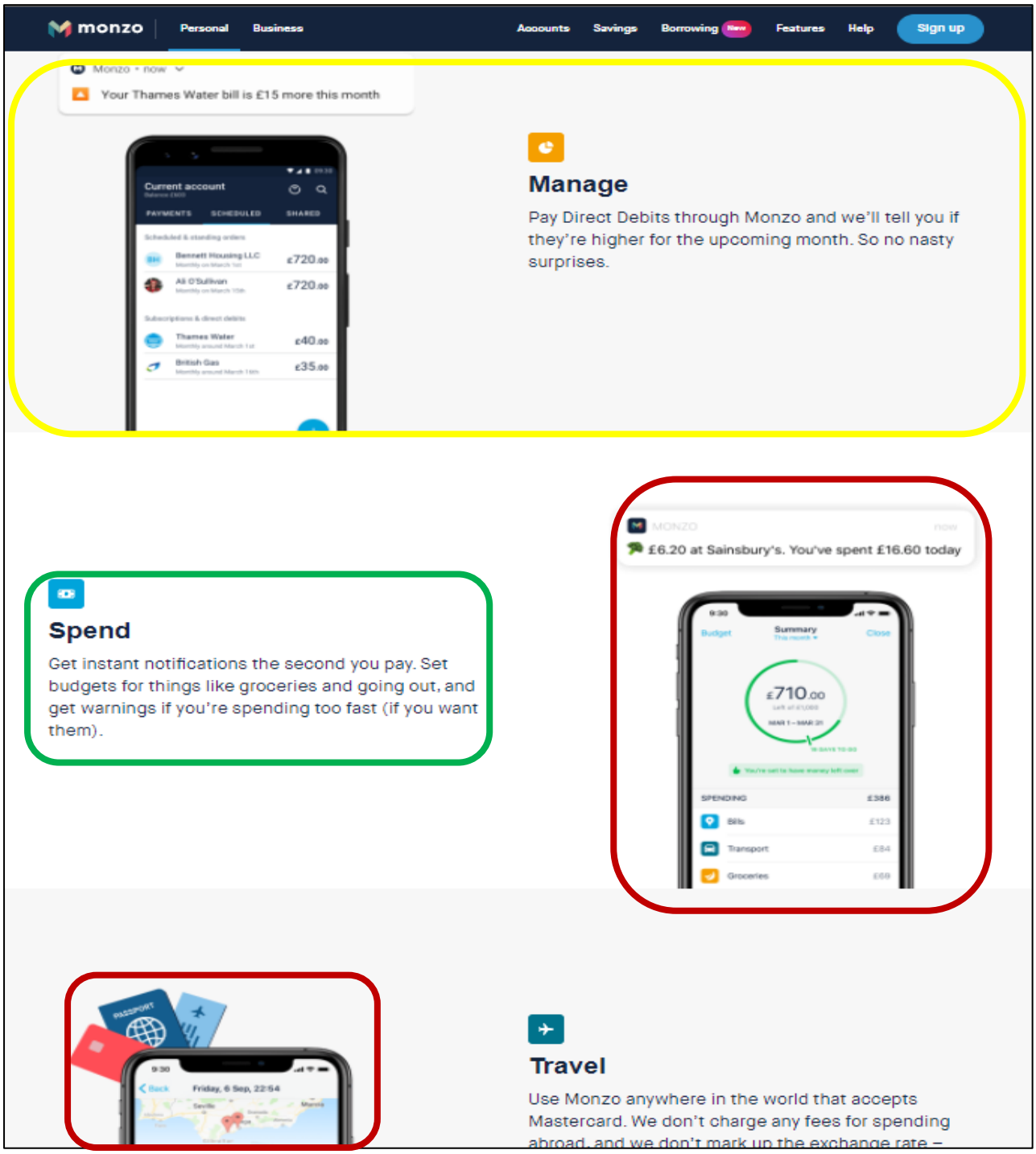


Figure 65 – Monzo personal page

Provides insight of the simple process of obtaining a Monzo card.

QR link development is a unique approach for users to access the application

Direct links to both android and apple stores to download this application quickly.

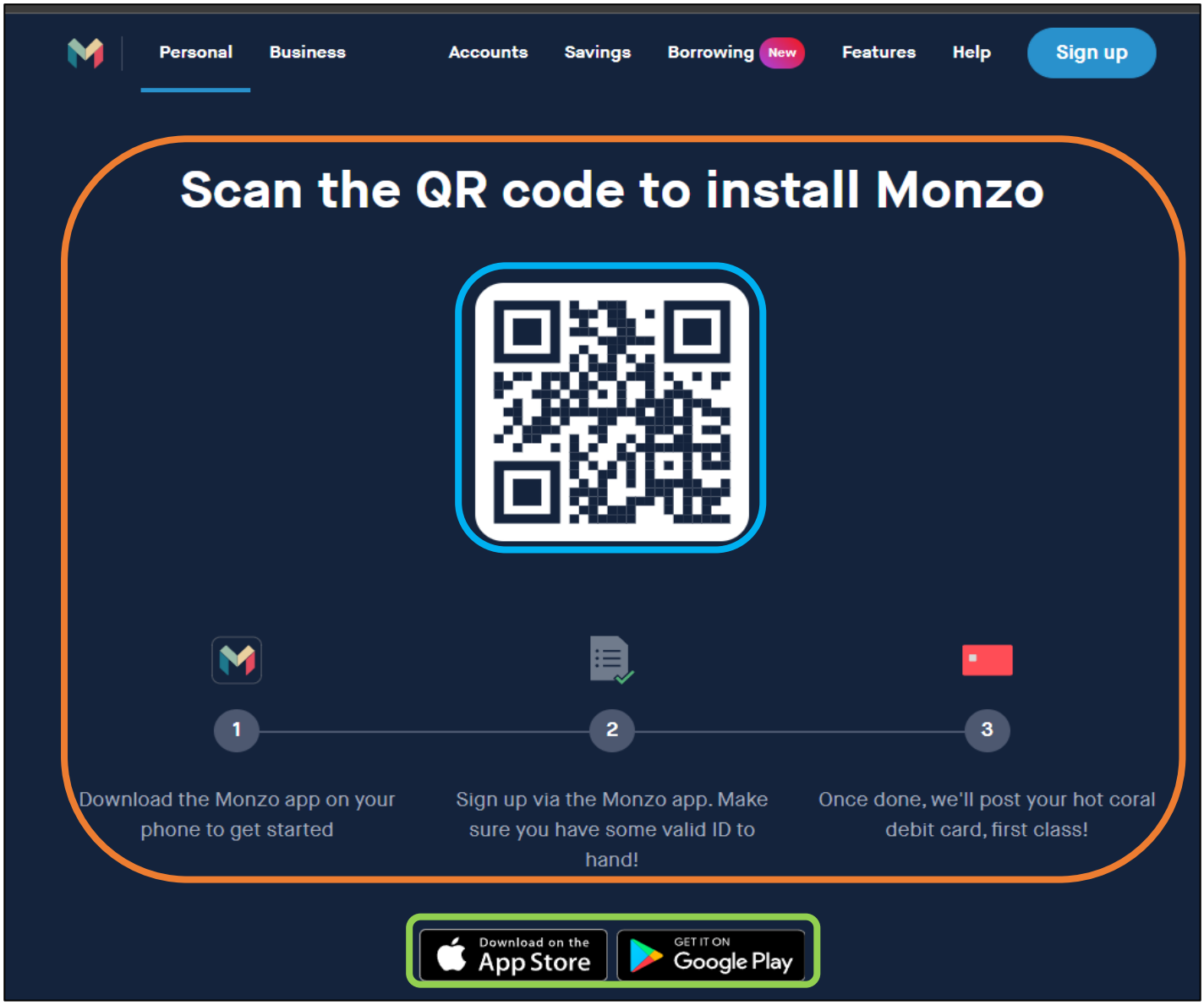


Figure 66 – Unique Monzo QR scannable feature

Mobile design

Mobile format displays a compact summary of user's expenditure using charts for better understanding.

Allows user to display different information for different account that they own.

Profile display which allows for easy access. An activity display to use when user is active.

Charts and messages to help users understand their expenditure. Messages used reassure consumers.

Transaction history which displays the ins and outs of user's different accounts.

Mobile notifications which displays purchase is a useful security feature to make sure users are making legitimate purchases. Additionally using transaction data to obtain data such as location of purchase to better analyse and tailor user experience.

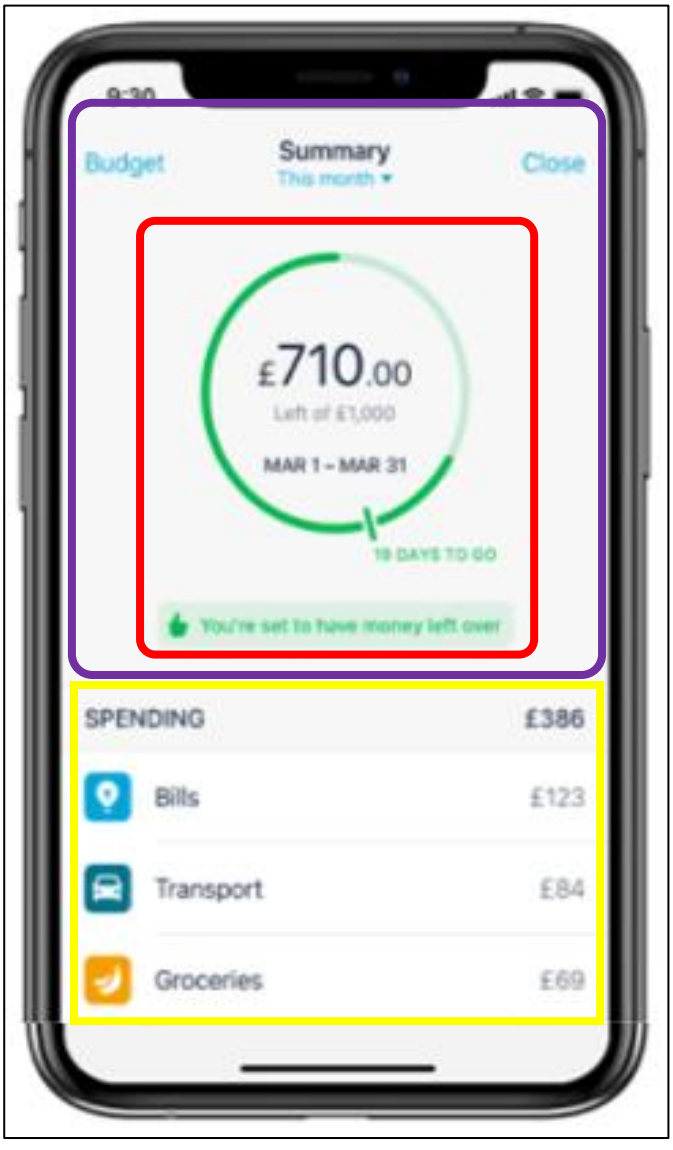


Figure 67 – Monzo mobile view 1

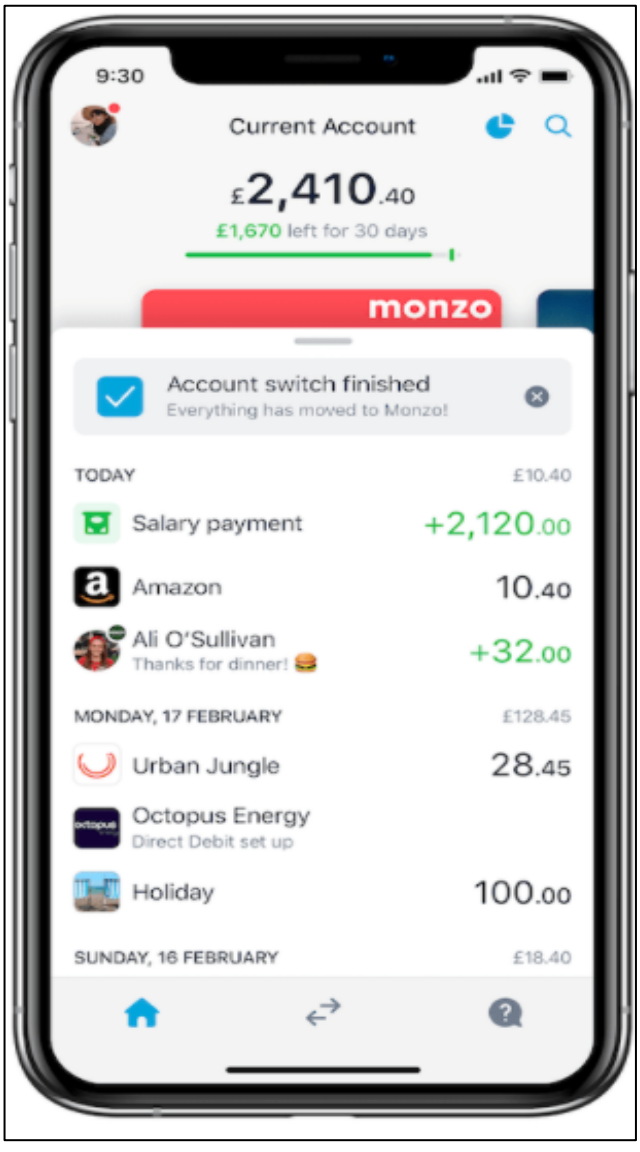


Figure 68 - Monzo mobile view 2

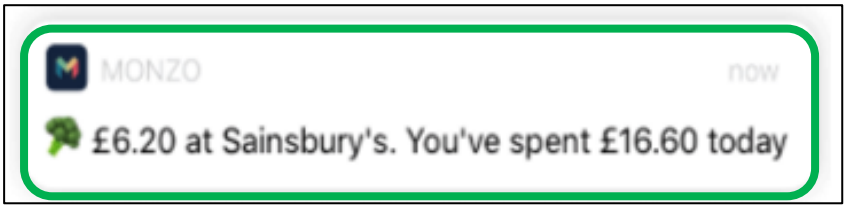


Figure 69 – Monzo Mobile notification

Influence

After researching the different competitor application designs this gave me insight in the general direction of how I wanted to style my application. Some specific details that both sites incorporated was the balance of dark colours for their background and light colours for their text, I wanted to introduce this same approach as this made reading text clear. Both application offered a minimalistic and professional design. This research has influenced me to adopt a similar UI design and to provide me with a example that I can work to.

Some specific design aspect particularly influenced me was the colour scheme used within the Monzo index page and the why it explains clearly the different operations and how they work.

Monzo and other applications provide an approach to visualise data with the use of charts .This influenced me into also applying the same approach, as introducing this feature will contribute to my requirements.

Security threats and mitigation methods

This section looks into the common web related attacks that could potentially occur on my application. This will be useful as it will help me correctly implement security methods to prevent these are threats.

OWASP TOP Ten web application security risks

OWASP Is an open community application created to help improve software security. This site was able to provide the top common web threats (OWASP, 2022).

1. Broken Access Control (94% of applications were tested for some form of broken access control)
2. Cryptographic Failures
3. Injection (94% of the applications were tested for some form of injection or cross-site Scripting)
4. Insecure Design
5. Security Misconfiguration (90% of applications were tested for some form of misconfiguration)

Mitigation methods

Broken Access Control (OWASP, 2022)

- Deny access to functionality by default.
- Use Access control lists and role-based authentication mechanisms.
- Do not just hide functions.

Cryptographic Failures (OWASP, 2022)

- Make sure to encrypt all sensitive data at rest.
- Ensure up-to-date and strong standard algorithms, protocols, and keys are in place; use proper key management.
- Always use authenticated encryption instead of just encryption.

Injection (OWASP, 2022)

- Input Validation
- Prepared statements
- Escaping All User Supplied Input

Insecure Design (OWASP, 2022).

- Segregate tier layers on the system and network layers depending on the exposure and protection needs
- Use threat modelling for critical authentication, access control, business logic, and key flows
- Establish and use a library of secure design patterns or paved road ready to use components.

Security Misconfiguration (OWASP, 2022)

- Disable administration interfaces.
- Disable debugging.
- Disable use of default accounts/passwords.

Influence

Understanding and learning the different types of a web application threats influenced me to research the types of mitigation methods I can implement within my application, so that I can deliver a secure artifact. It also made me caution when developing as I had to double check that I didn't introduce any security flaws with certain operations. Overall, this section help influence my ability to successfully development and implement a safe application to use. Some security aspects this research section help was the integrating of a 128-bit hash function which was used to encrypt sensitive data.

API research

What is an API

An API is an Application Programming Interface which is a software that allows two applications to talk to each other. They are used to access data and interact with external software components, operating systems, or microservices (Wyatt, 2022).

What is RESTful API

A REST API is a variation of a traditional API but is an application programming interface that conforms to the constraints of a REST architectural style and allows for interaction with RESTful web services. These types of API's communicate and send JSON data using HTTPS URLs (redhat, 2022).

What is a banking API

An API with an interface through which a financial institution provides data about customers, accounts, and transactions. They can make use of third-party financial services, which, in turn, access the data required by the original bank via the banking API (Rudolf, 2022). Some benefits with the use of banking APIs:

1. API banking improves customer experience.
2. Provide continuously adapting technology.
3. Provide the opportunity to expand payment services.

Influence

Gaining a better insight into API's helped influence me to uncover the correct API to use for my project this meant that I was looking for specific characteristics such as reliability and or availability. As this is a service that will perform critical operations within my application and should be required to continuously be available for end users. It also made me consider the type of API wanted to use, such as a basic API which communicate with an array of prebuilt functions or, A REST API which communicates and retrieves data through URL routes.

References

Developer.mozilla.org. 2022. Structuring the web with HTML - Learn web development | MDN. [online] Available at: <https://developer.mozilla.org/en-US/docs/Learn/HTML> [Accessed 5 March 2022].

Developer.mozilla.org. 2022. What is JavaScript? - Learn web development | MDN. [online] Available at: <https://developer.mozilla.org/en-US/docs/Learn/JavaScript/First_steps/What_is_JavaScript> [Accessed 5 April 2022].

W3techs.com. 2022. jQuery vs. Vue.js vs. Angular usage statistics, May 2022. [online] Available at: <https://w3techs.com/technologies/comparison/js-angularjs,js-jquery,js-vuejs> [Accessed 8 April 2022].

Hackr.io. 2022. [online] Available at: <https://hackr.io/blog/web-development-ide> [Accessed 6 April 2022].

Developer.mozilla.org. 2022. Express web framework (Node.js/JavaScript) - Learn web development | MDN. [online] Available at: <https://developer.mozilla.org/en-US/docs/Learn/Server-side/Express_Nodejs> [Accessed 7 April 2022].

Forbes.com. 2022. [online] Available at: <https://www.forbes.com/advisor/business/agile-vs-waterfall-methodology/> [Accessed 8 April 2022].

Meenan, P., 2022. WebPageTest - Website Performance and Optimization Test. [online] Webpagetest.org. Available at: <https://webpagetest.org/> [Accessed 9 April 2022].

Ui.dev. 2022. Am I Responsive?. [online] Available at: <https://ui.dev/amiresponsive> [Accessed 5 May 2022].

W3.org. 2022. World Wide Web Consortium (W3C). [online] Available at: <https://www.w3.org/> [Accessed 5 May 2022].

Appdevtools.com. 2022. HTML Validator Online - AppDevTools. [online] Available at: <https://appdevtools.com/html-validator> [Accessed 10 April 2022].

Jestjs.io. 2022. Jest · JavaScript Testing. [online] Available at: <https://jestjs.io/> [Accessed 5 May 2022].

Phpunit.de. 2022. PHPUnit – The PHP Testing Framework. [online] Available at: <https://phpunit.de/> [Accessed 5 May 2022].

GOV.UK. 2022. Data protection. [online] Available at: <https://www.gov.uk/data-protection> [Accessed 13 April 2022].

Cs.uwaterloo.ca. 2022. [online] Available at: <https://cs.uwaterloo.ca/~m2nagapp/courses/CS446/1195/Arch_Design_Activity/ClientServer.pdf> [Accessed 5 May 2022].

Web Accessibility of the Top Research-Intensive Universities in the UK. 2022. [online] Available at: <https://journals.sagepub.com/doi/full/10.1177/21582440211056614> [Accessed 15 April 2022].

Siteimprove. 2022. Accessible fonts: How to choose a font for web accessibility. [online] Available at: <https://www.siteimprove.com/glossary/accessible-fonts/#:~:text=Top%20accessible%20fonts&text=The%20most%20accessible%20fonts%20are,rather%20than%20the%20body%20text.]> [Accessed 12 May 2022].

size?, M., 2022. Minimum font size? | Accessible Web. [online] Accessible Web. Available at: <https://accessibleweb.com/question-answer/minimum-font-size/> [Accessed 5 May 2022].

PUMPKIN. 2022. ADVANTAGES OF USING BLUE IN YOUR WEB DESIGN. [online] Available at: <https://www.pumpkinwebdesign.com/web-design-manchester/advantages-of-using-blue-in-your-web-design/#:~:text=It%20is%20thought%20to%20evoke,a%20colour%20associated%20with%20optimism> [Accessed 24 April 2022].

Monzo.com. 2022. Banking made easy. [online] Available at: <https://monzo.com/> [Accessed 5 May 2022].

Owasp.org. 2022. OWASP Top Ten Web Application Security Risks | OWASP. [online] Available at: <https://owasp.org/www-project-top-ten/> [Accessed 5 May 2022].

Ptsecurity.com. 2022. How to prevent SQL injection attacks. [online] Available at: <https://www.ptsecurity.com/ww-en/analytics/knowledge-base/how-to-prevent-sql-injection-attacks/> [Accessed 5 May 2022].

Php.net. 2022. PHP: Prepared Statements - Manual. [online] Available at: <https://www.php.net/manual/en/mysqli.quickstart.prepared-statements.php> [Accessed 5 May 2022].

Chartjs.org. 2022. Chart.js | Open source HTML5 Charts for your website. [online] Available at: <https://www.chartjs.org/> [Accessed 5 May 2022].

Node.js. 2022. Node.js. [online] Available at: <https://nodejs.org/en/> [Accessed 5 May 2022].

Rudolf, D., 2022. The Basics of API Banking: What Every Bank Manager Should Know About Banking APIs. [online] Knowledge.fintecsystems.com. Available at: <https://knowledge.fintecsystems.com/en/blog/the-basics-of-api-banking-what-every-bank-manager-should-know-about-banking-apis#:~:text=A%20banking%20API%20is%20an,offered%20by%20their%20own%20bank> [Accessed 6 April 2022].

Appendices

Appendix 1 - Demonstration account and relevant information

URL location	https://jb1828.brighton.domains/FinalYearProject/index.php
GitHub source code link	JosephBatchelor/FinalYearProject: Here is the source code for my Open banking application (github.com)
Demonstration account Username and password for my web site login.	Username: 123 Password: 123 Email address: 123
Username and password for accessing mock data within the data API interface	Username: john Password: doe

Appendix 2– Record of supervisory meetings

#	Date	Discussion
1	12/10/21	<F2F> this was an initial meeting discussing what my project and what im going to do.
2	08/11/21	<Online> Intern report meeting discussing how to write it, what elements to include and how to write and design deliverables and requirements.
3	01/12/21	<Online> Viva meeting, we discussed different aspects of the interim report, which helped me correctly align and introduce my requirements and deliverables.
4	08/2/22	<F2F> meeting with supervisor to help me fix a database related problem.
5	15/02/22	<F2F> Progress check meeting making sure that I am on track.
6	21/02/22	<Online> This was a group meeting discussing the approach to writing the project report.
7	01/03/22	<Online> Progress check meeting making sure that I am on track.
8	22/3/22	<F2F> Final progress check before breaking up for easter.
9	04/5/2022	<Online> project report review meeting

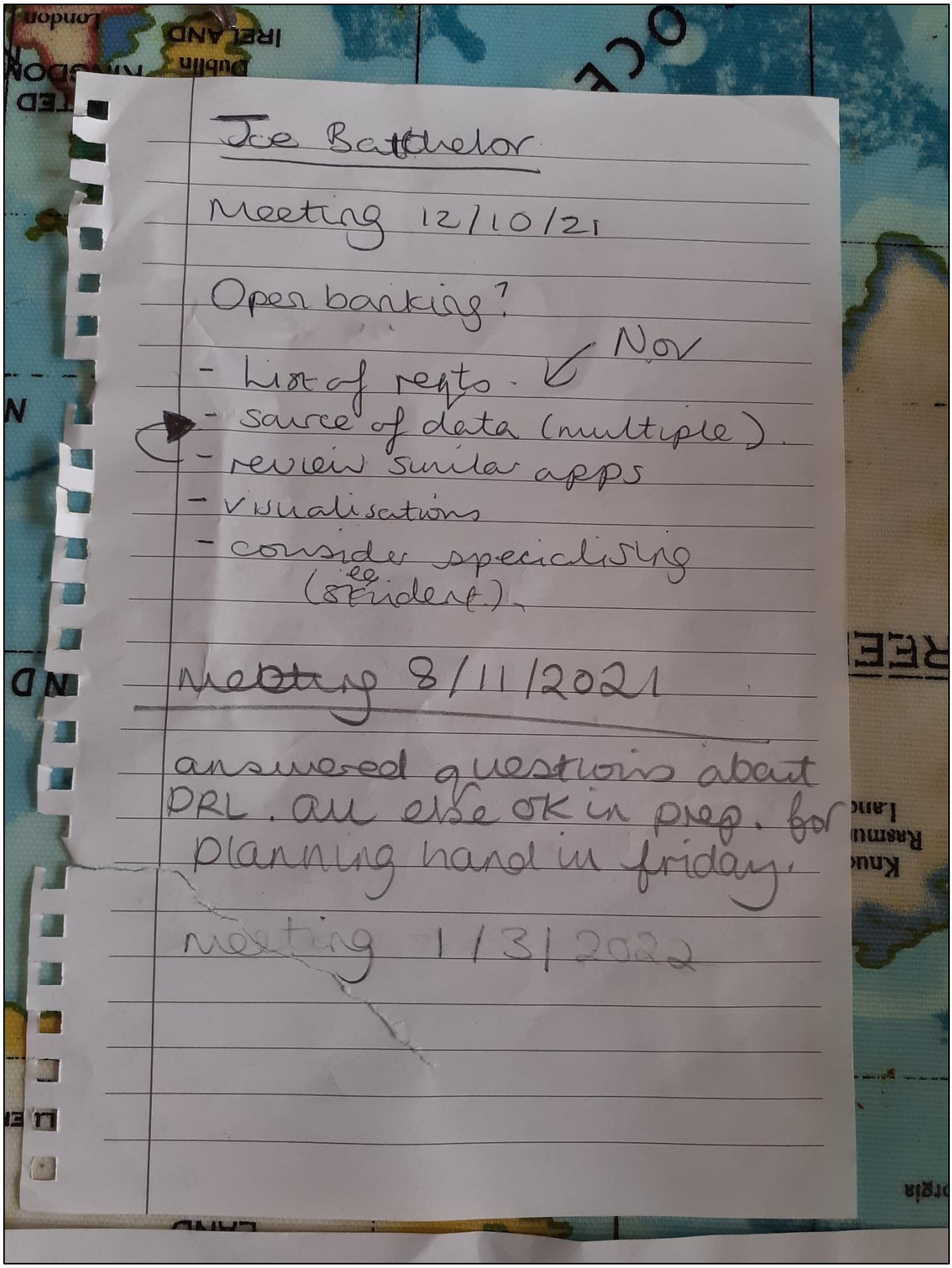


Figure 70 – 12/10/21 project meeting discussing potential features to consider for my application.

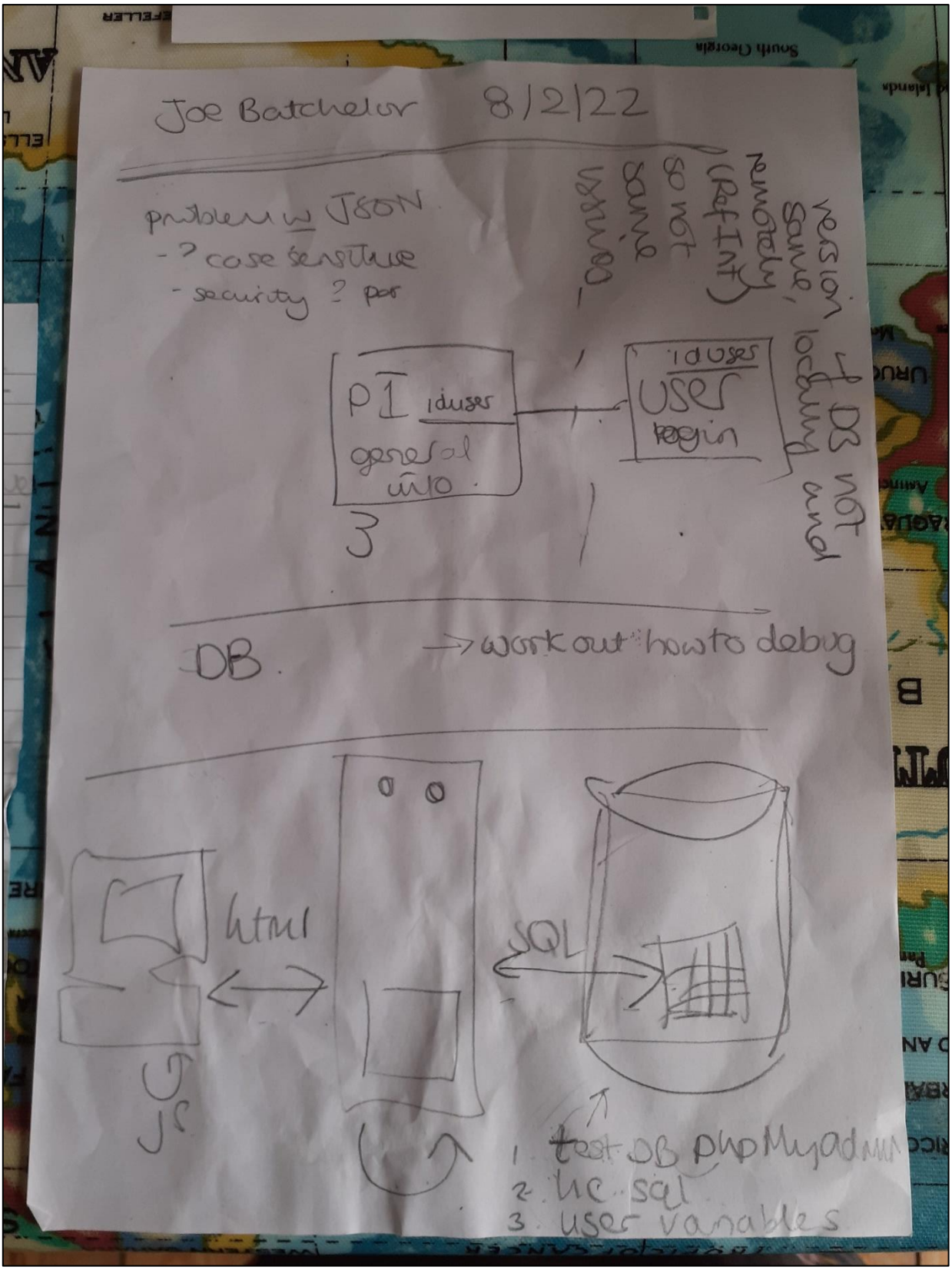


Figure 71 – 8/2/22 project meeting notes about a database problem that occurred. Listing how to overcome it.

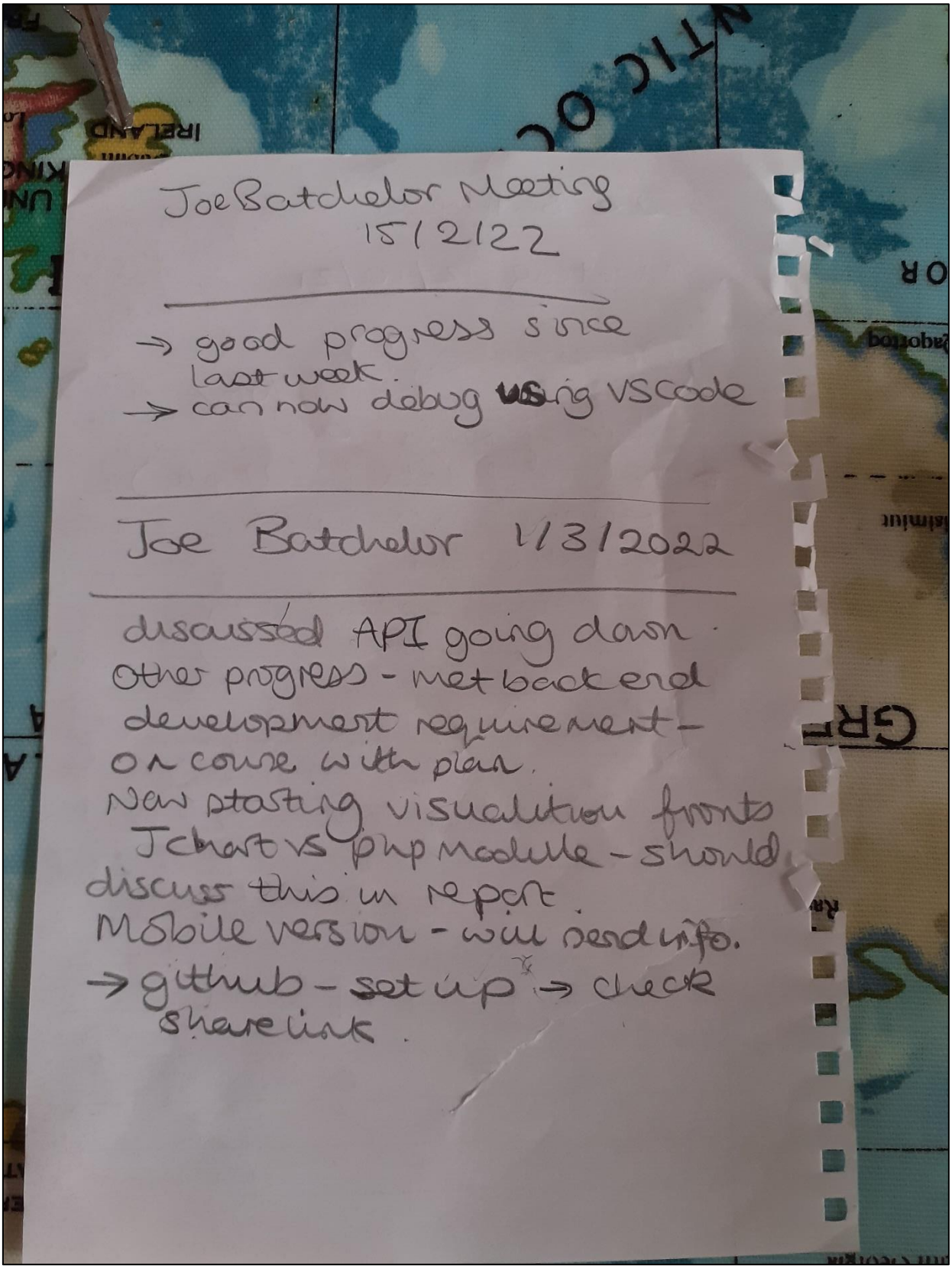


Figure 72 – 15/2/22 progress meeting listing all currently achieved features

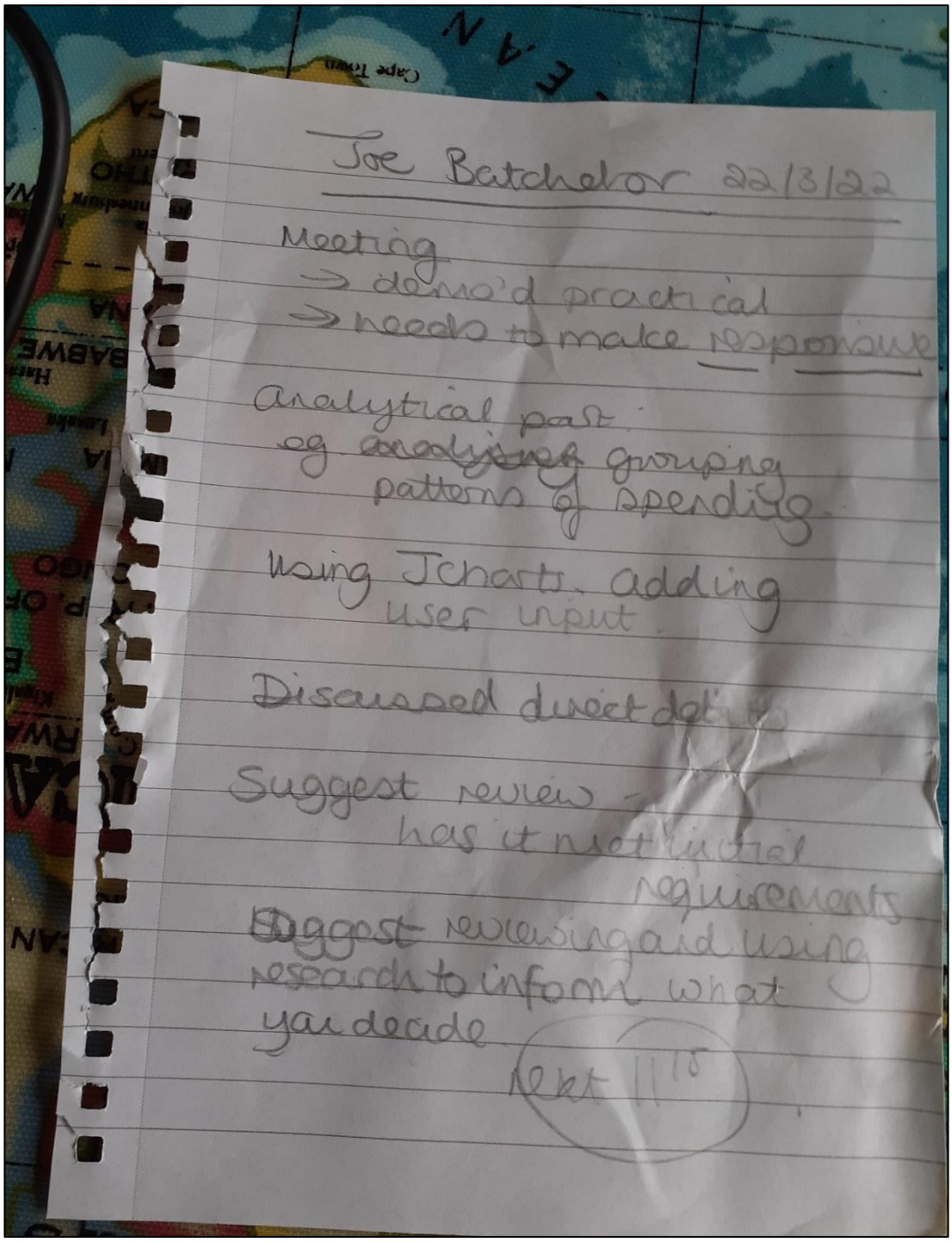


Figure 73 – 22/3/2022 – final progress meeting about what I have added.

Appendix 4 – Website operation tutorial as a new user

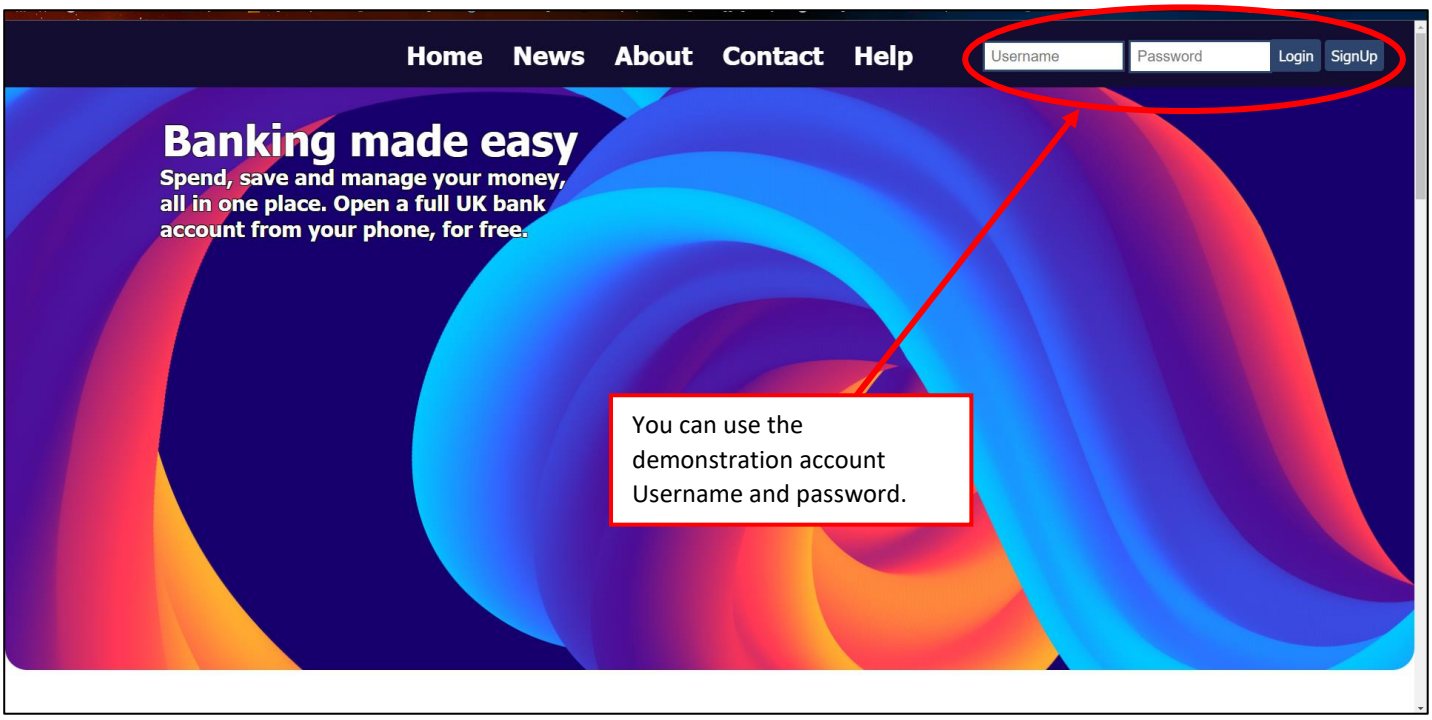


Figure 74 – My landing index page.

Figure 75 – close view of login section.

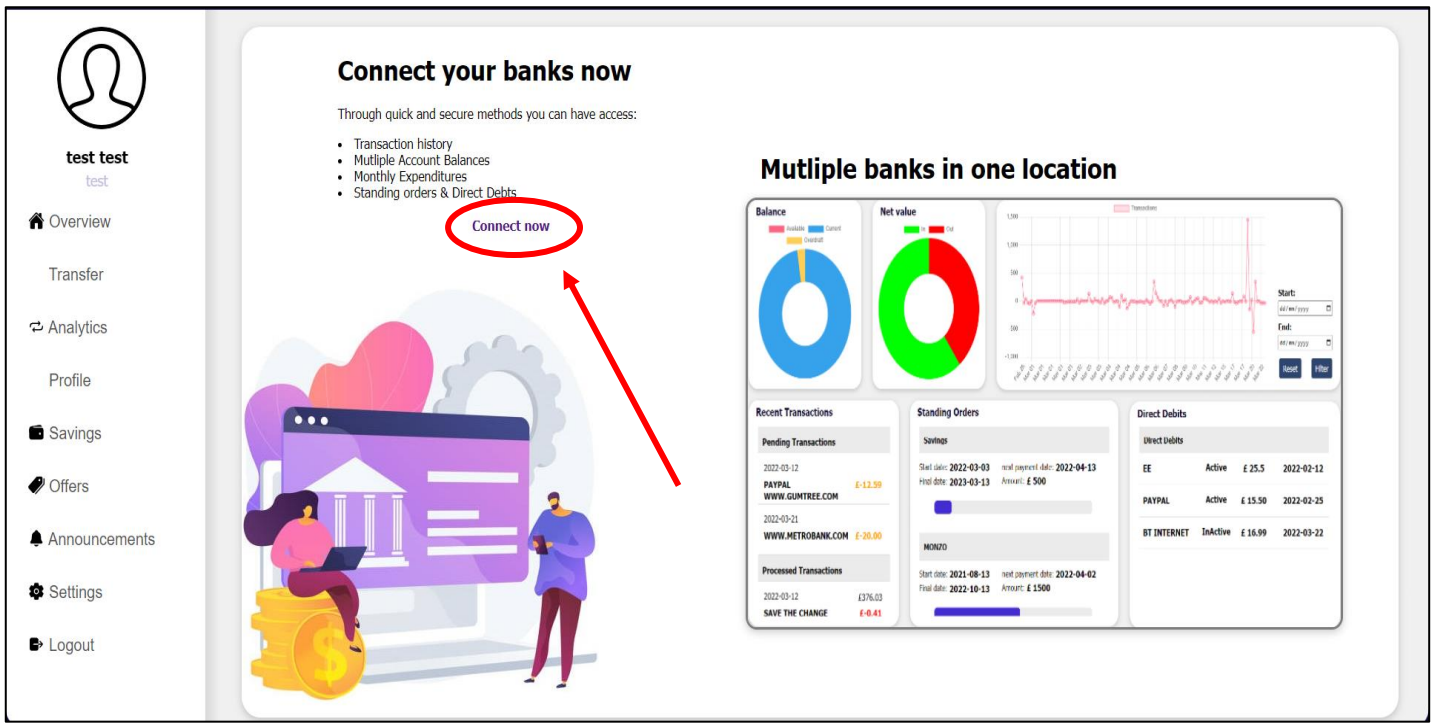


Figure 76 – Overview page before establishing banks.

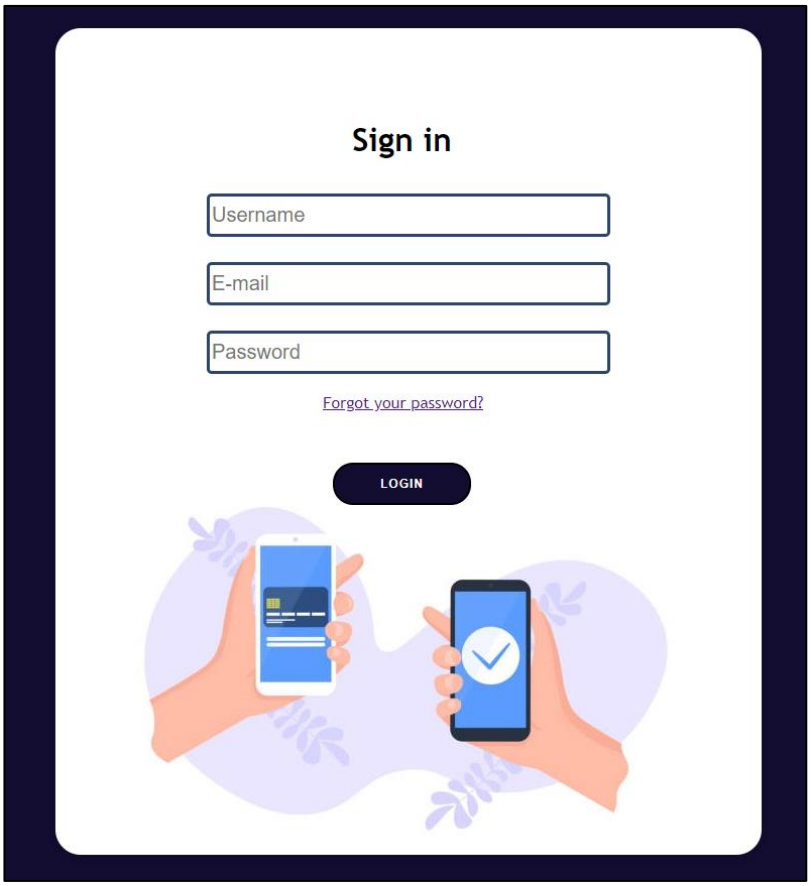


Figure 77 – Verification form.

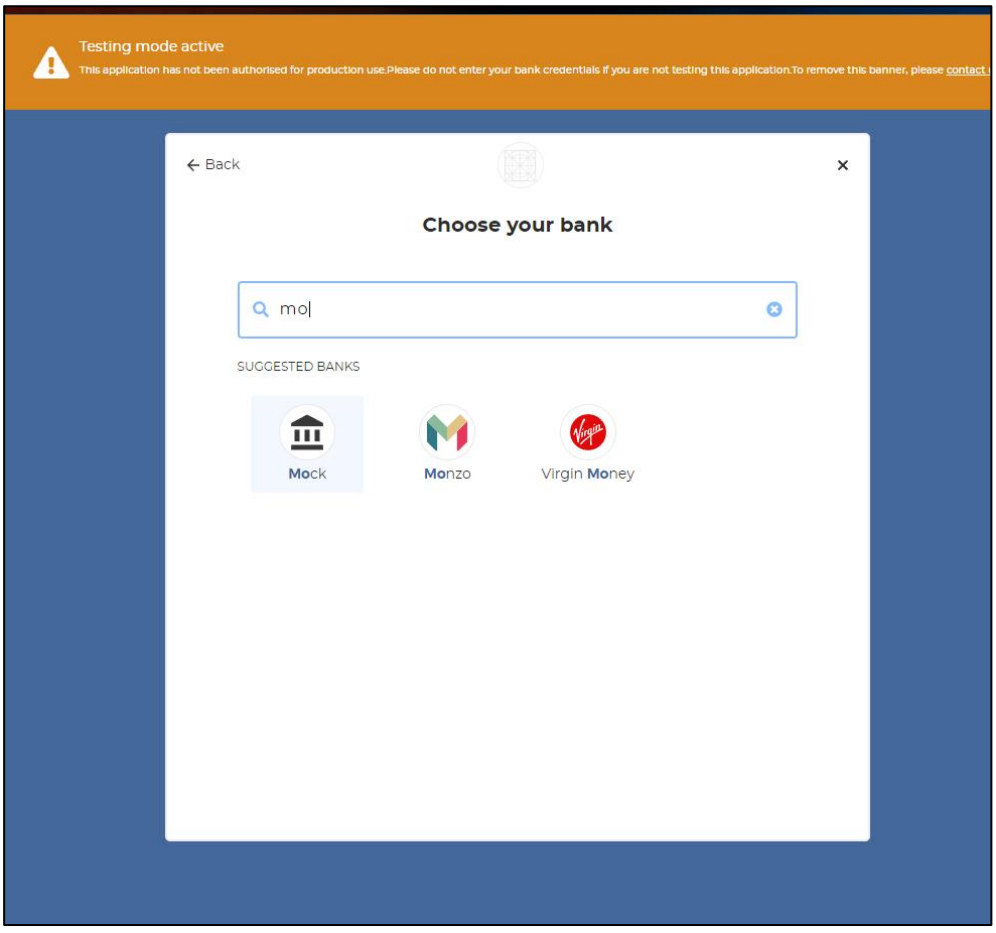


Figure 78 – API interface (choosing bank section)

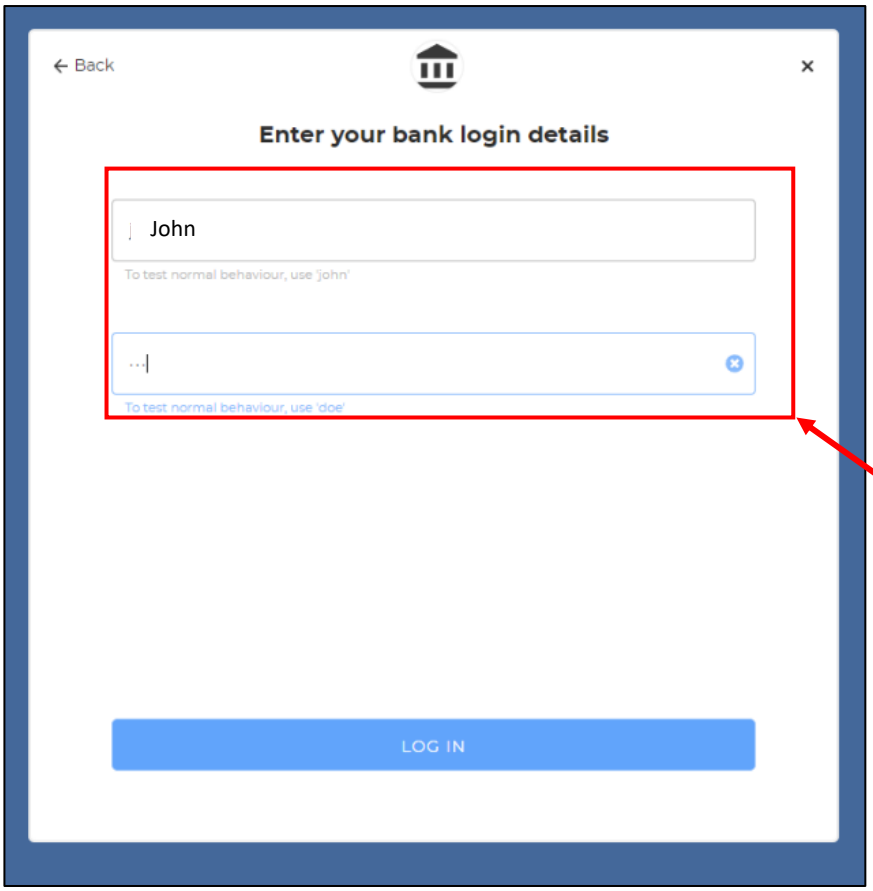


Figure 79 – API interface (authentication section)

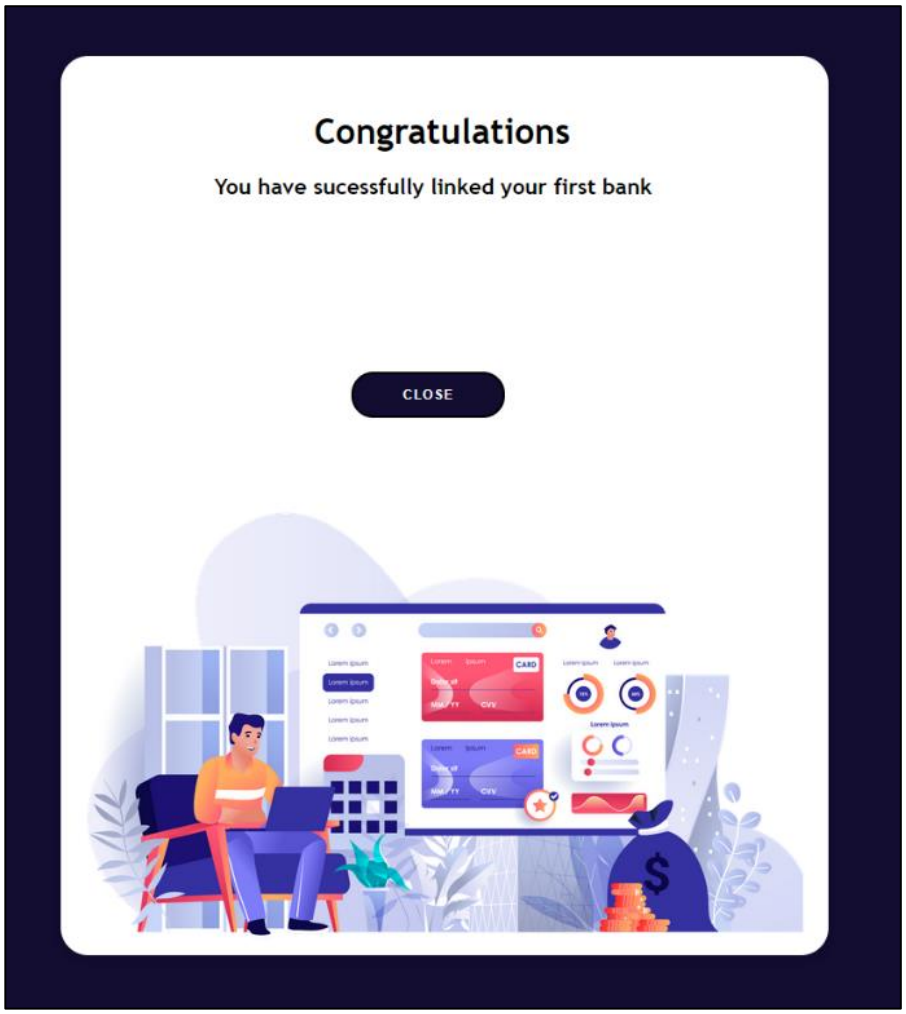


Figure 80 – Confirmation page

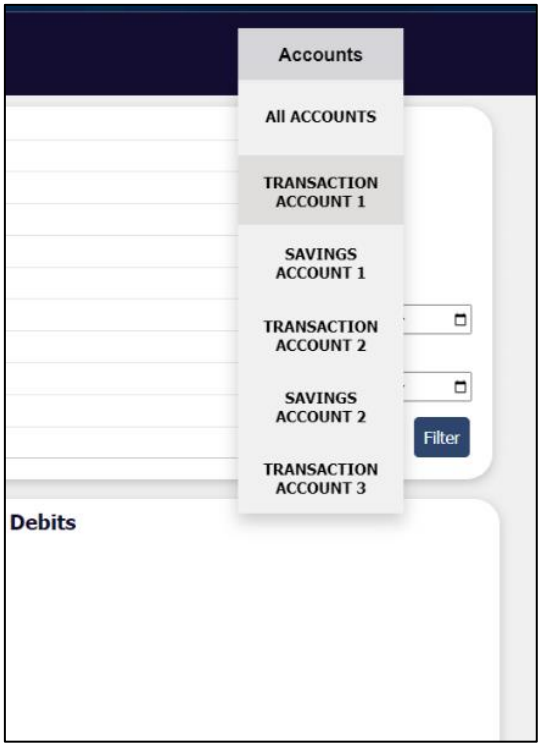


Figure 81 – Desired account select from drop down image.

Figure 82 - Overview page with established banks.