University of Plymouth

School of Engineering, Computing, and Mathematics

COMP3000

Computing Project

2020/2021

Back to Work

Evan Ward

10616392

BSc (Hons) Computer Science

# Acknowledgements

I would like to thank my family for the continued support throughout this project, helping me to stay focussed and motivated.

# Abstract

Households across the world, whether that be families or shared homes, tend to struggle to complete chores throughout the home. This is due to a lack of motivation (especially with children), lack of time or lack of organisation. This project aims to solve this problem by providing a universal interactive website accessible by anyone with any device that has access to a web browser.

The main body of this report discusses the project in full, outlining the project’s background, aims and objectives and a discussion about the method of approach. As well as this, this report covers the legal, social and ethical issues that encompass the project, an outline of the project management, the technologies used and system architecture and design. This report also documents the development process of the project and the solutions to some hurdles that entailed.

The final three sections of this report evaluate the final outcome of the project as a whole, outlining the successes and issues that where encountered. Here, the initial aims and objectives are evaluated as well as the developer’s performance throughout the project.

At the end of this report, the references used within the main body can be found, along with an appendix containing all relevant documents relating to the project.

Table of Contents

[Acknowledgements 2](#_Toc71634866)

[Abstract 2](#_Toc71634867)

[1 Introduction 6](#_Toc71634868)

[1.1 Background & Existing Solutions 6](#_Toc71634869)

[1.1.1 Background 6](#_Toc71634870)

[1.1.2 Existing Solutions 6](#_Toc71634871)

[1.2 Aims, Objectives and Deliverables 6](#_Toc71634872)

[1.2.1 Project Aims 7](#_Toc71634873)

[1.2.2 Project Objectives 7](#_Toc71634874)

[1.2.3 Project Deliverables 7](#_Toc71634875)

[2 Methods of Approach 8](#_Toc71634876)

[2.1 Project Structure 8](#_Toc71634877)

[2.2 Agile Project Management 8](#_Toc71634878)

[2.3 Risk Assessment 9](#_Toc71634879)

[2.4 Minimum Viable Product, Optional Requirements & Final Solution Limitations 10](#_Toc71634880)

[2.4.1 Minimum Viable Product 10](#_Toc71634881)

[2.4.2 Optional Requirements 11](#_Toc71634882)

[2.4.3 Final Solution Limitations 11](#_Toc71634883)

[2.4 Tools & Technologies 11](#_Toc71634884)

[2.4.1 HTML, CSS & PHP 11](#_Toc71634885)

[2.4.2 JavaScript 11](#_Toc71634886)

[2.4.3 MySQL 11](#_Toc71634887)

[2.4.4 WAMP 12](#_Toc71634888)

[2.4.5 Amazon Web Services 12](#_Toc71634889)

[3 Legal, Social & Ethical 13](#_Toc71634890)

[3.1 Legal 13](#_Toc71634891)

[3.1.1 Laws 13](#_Toc71634892)

[3.1.2 Licenses & Intellectual Property 13](#_Toc71634893)

[3.2 Social & Ethical 14](#_Toc71634894)

[4 Project Management 15](#_Toc71634895)

[4.1 Microsoft Office 365 Planner 15](#_Toc71634896)

[4.2 Agile Methodology 15](#_Toc71634897)

[4.2.1 Product Backlog 16](#_Toc71634898)

[4.2.2 Sprint Backlog 16](#_Toc71634899)

[4.2.3 Increments 16](#_Toc71634900)

[4.3 GitHub 16](#_Toc71634901)

[5 Architecture & Design 18](#_Toc71634902)

[5.1 Architecture 18](#_Toc71634903)

[5.2 Planning 19](#_Toc71634904)

[5.2.1 Functional Requirements 19](#_Toc71634905)

[5.2.2 Non-Functional Requirements 19](#_Toc71634906)

[5.2.3 User Stories 19](#_Toc71634907)

[5.2.4 Risk Assessment 19](#_Toc71634908)

[5.3 Design 20](#_Toc71634909)

[5.3.1 Entity Relationship Diagrams 20](#_Toc71634910)

[5.3.2 Storyboards 22](#_Toc71634911)

[5.3.3 UML Diagrams 24](#_Toc71634912)

[5.3.4 Use Case Diagrams 25](#_Toc71634913)

[5.4 Design Principles 28](#_Toc71634914)

[5.4.1 Responsiveness 28](#_Toc71634915)

[5.4.2 User Focused Design 28](#_Toc71634916)

[5.4.3 Don’t Repeat Yourself 28](#_Toc71634917)

[5.4.4 Security 29](#_Toc71634918)

[6 Project Development 30](#_Toc71634919)

[6.1 Sprint Zero: Project Initialisation 30](#_Toc71634920)

[6.2 Sprint One: Website & Database Design 30](#_Toc71634921)

[6.3 Sprint Two: User Accounts 31](#_Toc71634922)

[6.4 Sprint Three: Chores 33](#_Toc71634923)

[6.5 Sprint Four: Rewards System 35](#_Toc71634924)

[6.6 Sprint Five: Calendar 37](#_Toc71634925)

[6.7 Sprint Six: Clean Up & Testing 39](#_Toc71634926)

[7 Usability Study 40](#_Toc71634927)

[7.1 Results 40](#_Toc71634928)

[8 End-Project Report 42](#_Toc71634929)

[8.1 End-Project Summary 42](#_Toc71634930)

[8.2 Project Changes 42](#_Toc71634931)

[9 Project Post-Mortem 43](#_Toc71634932)

[9.1 Project Management Approach Evaluation 43](#_Toc71634933)

[9.2 Technologies Evaluation 43](#_Toc71634934)

[9.3 Developer Performance Evaluation 44](#_Toc71634935)

[9.4 Future Work 44](#_Toc71634936)

[10 Conclusion 44](#_Toc71634937)

[References 47](#_Toc71634938)

[Appendix 48](#_Toc71634939)

[Appendix 1: User Guide 48](#_Toc71634940)

[Appendix 2: Project Management 48](#_Toc71634941)

[2.1 Product Vision 48](#_Toc71634942)

[2.2 Functional Requirements 48](#_Toc71634943)

[2.3 Sprint Plans 51](#_Toc71634944)

Word Count: 9729

GitHub: <https://github.com/EvanWard29/COMP3000-BackToWork.git>

Website:

# Introduction

## Background & Existing Solutions

### 1.1.1 Background

This project is designed to support households who struggle to keep their houses clean and tidy on a day-to-day basis and want to be a bit more organised around the home. This can apply to families or student/shared homes. As any person knows, doing chores is not fun and can be a daunting task at times. Children especially find it difficult to complete chores because they may feel it is a waste of their time which could be spent doing more exciting things that interest them. The same can be applied to some adults, however most adults see the importance of doing chores and just get on with it. From personal experience growing up with siblings, chores were one thing that was always disliked, and to this day they can still be tedious.

Some families may also struggle to stay organised with assigning tasks. Methods such as displaying paper-based or whiteboard charts in the kitchen may not always work due to them being too tedious to physically write up a plan for the week or because they are easily destroyed. In an increasingly digital world, outdated methods of organising chores through paper-based and whiteboard charts have become obsolete. Every family nowadays has some access to the internet through various devices such as mobile phones, tablets or PCs. This project aims to take advantage of this by keeping families organised and digitally connected through a single website accessible through any device with an internet browser.

### 1.1.2 Existing Solutions

There are numerous mobile applications that offer some form of chore rosters that can be downloaded and installed from app stores. Each have similar features, such as the ability to assign chores to members of a group or offer rewards as an incentive for completing chores. However, these applications are restricted to mobile phones only, there are very few websites available on the internet that offer this kind of service. This project aims to fill that gap by offering a unique website that functions like mobile versions but with additional features and free of charge. By creating a full-stack website instead of a mobile application, this service can be accessed at any time through any device with an internet browser, making it portable and easy to use.

## Aims, Objectives and Deliverables

### Project Aims

The aim of this project is to produce a fully functioning full-stack website that offers the ability for users to add members to a household group where they can then assign chores to each member to complete within a given time.

This project is mainly aimed at families who struggle to organise and complete chores around the house. This project is also targeted at shared student homes who wish to have some form of chore roster to split chores amongst housemates.

Users will be rewarded with points for completing chores and can be redeemed for rewards of their choice when enough points have accumulated. If chores are not completed on time, points will be deducted from the user.

Users will also be able to set up a calendar that will be synced with the group. On this calendar, users can plan their week ahead by creating new events on the calendar. Users will also be able to view when their assigned chores need to be completed by on this calendar.

### Project Objectives

The following project objectives were established at the beginning of the project:

* Design a full stack interactive website that is user-friendly using storyboards.
* Set out the structure of a relational database for storing user data using entity relationship diagrams.
* Use HTML and CSS to implement the designed website.
* Setup a relational database using MySQL.
* Implement the functionality of the website using PHP as the server-side language and JavaScript as the client-side language.
* Deploy the website and database to Amazon Web Services for hosting.

### 1.2.3 Project Deliverables

The final deliverables of the project will consist of a fully functioning full-stack website and a project report outlining the process of designing, implementing and deploying of this project.

# Methods of Approach

## Project Structure

The structure of this project can be broken down into three parts:

* Planning
* Development
* Analysis

The planning phase at the start of the project was used to identify the problem and outline how a solution to this problem would be approached. From this, an appropriate project management approach was selected that would be used to structure the development of the solution and which technologies would be used to develop the solution. This provided a clear guide as to how the solution would be designed and implemented in the development phase.

The development phase was the execution of the planned approach to the problem identified in the planning phase. The first step of this phase was to design the solution using storyboards, entity relationship diagrams and UML diagrams. The second step was to implement the designed solution. The third step was to test the implemented solution through usability studies to get valuable feedback that would be used to perfect the solution.

The analysis phase is the last phase. This phase would evaluate the project as a whole in the form of this written report.

## Agile Project Management

Agile project management was chosen as the approach to take for structuring this project. Agile project management is an iterative approach to software development where work is delivered in small increments throughout the development process instead of one big launch. Requirements, plans, and results are evaluated continuously so teams have a natural mechanism for responding to change quickly ((i) Atlassian, 2021).

For this project, scrum was chosen as the agile methodology to be followed. The scrum methodology splits a project’s development into three parts:

* The Product Backlog
* The Sprint Backlog
* Increments

The product backlog outlines the master list of work that needs to be done. This is a dynamic list of features, requirements, enhancements, and fixes that acts as the input for sprints ((i) Atlassian, 2021). For this project, my product backlog was outlined using Microsoft Planner.

The sprint backlog is the list of items, user stories, or bug fixes, selected by the development team for implementation in the current sprint cycle ((i) Atlassian, 2021). Scrum breaks the development of a product into short sprints that last 2-4 weeks. Each sprint implements a select number of features chosen at the beginning of each sprint.

Increments are the usable end-product of a sprint ((i) Atlassian, 2021). At the end of each sprint, a review is carried out that goes over what has been completed during the current sprint, any bugs that have been identified and need fixing, and planning for the next sprint cycle.

This methodology has allowed the project to be broken down into five sprints to be worked on over a period of 3-5 weeks each. At the end of each sprint, a sprint review is carried out and planning for next sprint is carried out. Doing this provides a clear guide as which stage the project’s development is in and how to go about developing the project.

## Risk Assessment

At the start of the project during the planning phase, a risk assessment was carried out to identify any risks that may affect the project and how to overcome them.

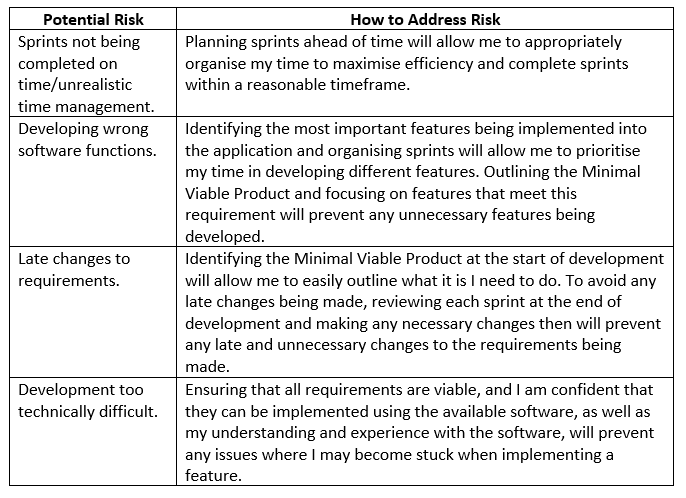


Figure 1 Risk Assessment

## Minimum Viable Product, Optional Requirements & Final Solution Limitations

### Minimum Viable Product

The following requirements are the minimum requirements of the solution and are considered to be the Minimal Viable Product:

* Allow users to create groups to add members of their family/household.
* Allow users to create new chores to be completed around the house.
* Allow users to assign chores to a member of their group to complete.
* Allow users to plan ahead by creating a new event on a calendar synced between the group.
* Allow users to view their assigned chores in a table and on the calendar.
* Allow users to earn reward points when completing a chore or deduct rewards when failing to complete a chore.

### Optional Requirements

The following requirements are considered to be optional and are only to be attempted if time permits it:

* Allow group members to message each other in a group chat.
* Allow group members to private message others within the same household group and outside their household group.
* Allow users to receive notification reminders for completing chores.

### Final Solution Limitations

The following is considered to be the limitations of the final solution:

* The final solution will be a website and so is only accessible through an internet web browser.

## 2.4 Tools & Technologies

### 2.4.1 HTML, CSS & PHP

PHP was used as the main backend language as well as HTML and CSS for the website design. This language was chosen because PHP can be easily embedded in HTML (PHP, 2021). PHP was also chosen as the server language due to the author’s experience of working with PHP, as apposed to NodeJS for example where experience is limited.

### 2.4.2 JavaScript

JavaScript was used as the main frontend language. JavaScript is a scripting language that enables you to create dynamically updating content, control multimedia, animate images, and pretty much everything else (MDN Web Docs, 2021). JavaScript was chosen as the main front-end language because it is widely accepted by most web browsers and the authors experience and knowledge of using the language was much more developed.

### MySQL

A MySQL server was used for storing user data instead of an alternative database such as MongoDB because MySQL allows for relational databases. This allowed the author to easily implement the entity relationship diagrams that were designed for each table of the database. MySQL is also an industry standard language that is widely accepted.

### 2.4.4 WAMP

Windows, Apache, MySQL, PHP (WAMP), is a software stack that acts like a virtual server on your computer (Hostinger Tutorials, 2021). For this reason, as well as the authors extensive experience using this stack, WAMP was chosen as the software stack for this project. WAMP is easy to use as it is a secure locally hosted server that can be easily accessed and does not require files to be uploaded to an external service, saving time and ensuring files are safely stored locally.

### Amazon Web Services

Amazon Web Services (AWS) was used for the hosting of the website and MySQL database. AWS has a variety of services that users can use to build and deploy applications. AWS was used to create an EC2 Windows Server 2019 instance running Apache for hosting the website. AWS was also used to create and host a RDS MySQL database server for storing user data securely.

# Legal, Social & Ethical

## 3.1 Legal

There are various legal requirements that must be considered and adhered to when designing any application. For this project, there are two important laws that must be considered when building the final solution. These are GDPR (General Data Protection Regulation, 2018) and the Data Protection Act (Data Protection Act, 2018).

### 3.1.1 Laws

#### 3.1.1.1 General Data Protection Regulation (GDPR)

The final solution of this project will be GDPR compliant. Cookies will be used by the website. In order to comply with GDBR, the website will ask the user for consent for the use of cookies via a popup. Any data that is collected can be fully controlled by the user. Data collected will be stored in a remote secure database. Users will also be able to delete their accounts which will also delete all user data collected.

#### 3.1.1.2 Data Protection Act

The final solution of this project will also comply with the Data Protection Act. The website will only collect data that is required by the website to function correctly. Users will be able to view and modify this data from the ‘My Account’ page. Users are able to change their email and password but cannot update any other data. All data collected is first validated before being processed in order to ensure the data is correct. Data such as passwords are also encrypted for security reasons. The website is also hosted on a secure encrypted server via Amazon Web Services.

In order to comply fully with the Data Protection Act users must be able to update their information. This is a feature that would need to be implemented if the project were to become commercially viable. However, this is out of the scope of the project and is intended for educational purposes only.

## 3.1.2 Licenses & Intellectual Property

The final solution will make use of a number of third-party features. All licenses have been listed in the project’s ‘README.md’ file where appropriate. The final solution has made use of three third-party libraries/assets:

* jQuery – licenced under the MIT Licence. jQuery is a plugin for JavaScript and has been used in this solution for the client-side coding of the website. jQuery is free to use.
* Bootstrap – licenced under the MIT Licence. Bootstrap was used for easily styling HTML elements via CSS classes. Bootstrap is free to use.
* Merriweather Font Family – licenced under the Open Fonts Licence (OFL). This font has been used for styling the website. This font is free to use, as long as it isn’t sold by itself.
* CryptoJS – licenced under the MIT License. CryptoJS is a plugin for JavaScript. This plugin provides a secure way of encrypting data and was used in this solution for encrypting and decrypting user data. CryptoJS is free to use.

## 3.2 Social & Ethical

One of the biggest social and ethical issues websites face is the collection of user data. User data is collected by ‘Back to Work’ in order to fully function correctly. Data that is collected for the user is the minimum amount needed for website to function. This avoids any issues with collecting any unnecessary data and keeps the social and ethical use of data under control.

Another social and ethical issue is the sharing of user data with third-party businesses. The solution does not make use of any third-party APIs and so all data collected by the solution is not shared with anyone.

Back to Work allows users to create admin accounts. These accounts can only be made by people over the age of 14. This is due to the fact that sensitive user data is collected, and admins of a group have full control of accounts under the age of 14 and so the user needs to be sensible with the data they are handling. If a user is under the age of 14, an account can be made for them via an admin, but the account will be restricted to the group of the admin making the account. Accounts can also be deleted by the admin if need be. This ensures parents have control of their child’s accounts at all times if a family is using the solution.

Test data collected via a usability study must also be acceptable in terms of social and ethical issues. In accordance with the ‘University of Plymouth Ethics Policy’, all data is anonymously collected deleted afterwards. All test data was gathered from students of ‘University of Plymouth’ in line with the ‘University of Plymouth Ethics Policy’.

# 4 Project Management

## 4.1 Microsoft Office 365 Planner

Microsoft Planner was the main online project management tool used for this project. Office 365 Planner provides a hub for team members to create plans, organise and assign tasks to different users and to check updates on progress through dashboards (Curry, 2019). Microsoft Planner allows users to create buckets containing tasks that need to be completed. These tasks can be assigned to specific users within a group along with a start date and deadline for completion. Microsoft Planner and other similar project management tools are widely used for Scrum style workflows across the industry.

Within the developer’s Planner, six buckets were created, one for the Product Backlog and five for sprints. Each bucket contained tasks that needed to be completed before a specified deadline. This allowed the developer to structure their time wisely and ensure there was enough time to focus on fully implementing a feature within the backlog correctly and efficiently.



Figure 2 Microsoft Office 365 Planner

## 4.2 Agile Methodology

Scrum was the chosen Agile Methodology for this project. As such, the project was split into three parts: Product Backlog, Sprint Backlog and Increments.

### 4.2.1 Product Backlog

Within the developer’s Planner, one bucket was created specifically for the product backlog. This bucket contained tasks that represented a feature of Back to Work that needed to be implemented. Each feature was given a colour label, a priority and set a start date and deadline for completion. This allowed the developer to keep track of which features had been implemented already and which features still needed completing.

### 4.2.2 Sprint Backlog

In order to tackle the features within the product backlog, five sprints were set up that lasted 3-5 Weeks. Each sprint contained multiple tasks that focused on completing one or two features within the backlog, depending on the difficulty of implementing each feature. Each task within a sprint was colour labelled which linked to a feature within the product backlog. This allowed the developer to easily check when a feature could be marked as complete and fully implemented within the product backlog. Creating sprints allowed the developer to structure their time appropriately and efficiently, ensuring there was enough time to fully implement a feature correctly.

### 4.2.3 Increments

Increments were the final products of sprints. Once a sprint had been completed and merged with the master product, a sprint review was carried out. This review focused on evaluating what went well during the sprint cycle, what could have gone better, and things to consider when moving onto the next sprint cycle. These reviews allowed the developer to reflect on the work that had been done so far and decide whether extra time would be needed for completing other sprints or whether a features priority needed to be changed depending on time constraints.

## 4.3 GitHub

Version Control is an important aspect of project management as it allows developers to easily structure the development of a product through branches. It also allows developers to revert changes to previous versions of the product if the changes made on the current branch need to be scrapped or in the case of irreversible data loss. GitHub is a widely used version control system across the industry. GitHub allows developers store data remotely, ensuring there is always a backup to the current project in the case of data loss. It is also easy to use through command line or with the desktop app. As such, GitHub was chosen as the version control system for this project.

GitHub allows users to create branches off of a master branch. This allows developers to work on a feature that needs implementing on a separate branch that will not affect the master branch. When a change is made to the product, developers can commit these changes to the current branch. This allows developers to easily revert changes to an older commit if need be.

For this project, the developer created a new branch of the master product whenever work was started on a new sprint from the sprint backlog. Once a sprint had been completed, the current branch would be merged with the master branch and that would become the master product. Overall, there were five branches made, one for each sprint and an additional sixth branch at the end for testing and project clean-up.

# 5 Architecture & Design

Before starting the development of the project, it was essential to plan the architecture and design of the product to allow for a smooth development process which could easily follow the design plans. This was vital in order to ensure the success of the project. Each aspect of the solution was carefully planned ahead in order to outline the key features required to be able to satisfy the minimum viable product, as well as any other features that are optional that could be introduced if there is enough time or at a later date in the future. During planning, the solution’s architecture was chosen, as well as the technologies that would be used to produce the final solution.

## 5.1 Architecture

WAMP was the chosen software stack for this project. This is due to the developer’s experience using the stack and the technologies associated with it. WAMP was also chosen as the project’s software stack because it allows developers to host websites locally which made developing and debugging the website much easier. Apache was chosen as the web server to use because it is the most widely used and accepted web server available (wpbeginner, 2021). MySQL was chosen as the database to use because the developer has the most experience using MySQL than they do any other database, such as MongoDB. MySQL is a relational database management system which is widely used alongside web development for storing relational data (123 Reg, 2021). This would make building the database much easier as the developer planned for it to be mainly relational. PHP was the chosen backend server language to use because it is a widely used and open source which allows developers to easily embed code within HTML (PHP, 2021). This backend language was chosen because the developer has the most experience using PHP than any other backend language such as NodeJS. JavaScript was used for the frontend development of the website because it is easy to use and has very useful plugins, such as jQuery, that allow you to easily manipulate elements on the webpage. The below image is the full system architecture of the website.

Graphical user interface, application

Description automatically generated

Figure 3 System Architecture of Back to Work

## 5.2 Planning

### 5.2.1 Functional Requirements

When planning the project, clear functional requirements of the final solution were carefully chosen and outlined (2.2 Functional Requirements). This ensured that a working and finished solution that satisfied the minimum viable product could be produced before the deadline. This made planning sprints easier as I was able to split the functional requirements amongst the five sprints in order of importance and difficulty, allowing for a smooth implementation of each feature during the development process.

### 5.2.2 Non-Functional Requirements

Some additional non-requirements were also planned at the beginning of the project. These requirements were not necessary in order to satisfy the minimum viable product but were optional features that would further extend the functionality of the final solution (2.3 Non-Functional Requirements). These requirements were only to be implemented if time permitted it or at some point in the future.

### 5.2.3 User Stories

In order to plan the functional requirements and outline the minimum viable product, I came up with some user stories that I envisioned a user would want to do with the final product (2.4 User Stories). The definition of user stories is as follows: “A user story is an informal, general explanation of a software feature written from the perspective of the end user. Its purpose is to articulate how a software feature will provide value to the customer” ((ii) Atlassian, 2021). These user stories helped me to prioritise each requirement and plan the five sprints appropriately based off of each requirement’s priority.

### 5.2.4 Risk Assessment

At the beginning of the project, I performed a risk assessment (2.5 Risk Assessment). Risk assessments are usually performed during a project’s start-up and are essential within the agile framework and when working with a team within an agile project. This is because teams need to know how to avoid risks or overcome a risk if one is encountered during a product’s development. During the initialisation phase of the project, I performed a risk assessment which consisted of the following steps:

* Risk Identification
* Risk Assessment
* Risk Response

During the risk identification step, I identified any risks that could occur during the product’s development. These risks could have been anything from encountering risks of software bugs or risks of completing sprints past the deadline of the sprint. The next step in the risk assessment was to assess what level of threat the potential risk would be to the project. Some risks were low and would have little impact on the project’s development. Other risks were more extreme, such as data getting lost, and would have a much bigger impact on the project’s development. The next step was used to identify solutions to the risks and how I could avoid encountering the risks in the first place. This process allowed me to easily continue development of the project without worrying about unexpected risks that could potentially hinder the development of the product.

## 5.3 Design

Before starting development of a project, development teams need to plan the architecture of the software to be developed. This means designing the structure of the code, databases and interfaces, etc. This allows development teams to have a clear vision as to what they need to be building and how they should go about doing it.

### 5.3.1 Entity Relationship Diagrams

Various diagrams are used to design the structure of code, database and interfaces. One of these diagrams is Entity Relationship Diagrams (ERDs). These diagrams are used for designing the structure of databases as well as the relationships of the tables within the database if using a relational database such as MySQL.

During the planning phase of my project, I used entity relationship diagrams to define the structure of my relational database that would be used to store user data for Back to Work. I had a clear idea of how the data within my tables would relate with one another. Visually representing this idea with an entity relationship diagram made it easier to build the actual database. The below figure is my initial entity relationship diagram when planning the project.

Diagram

Description automatically generated

Figure 4 Initial Entity Relationship Diagram

Throughout the development of the project, more tables were added to the database due to having new features and ideas to add to the website. One of these ideas was the calendar. Originally the calendar was to only be implemented if there was enough time for me to do so. However, after carefully considering it, the calendar functionality was added earlier due to new ideas that would improve the user experience. As a result, the following figure is the final entity relationship of the system.

Diagram

Description automatically generated

Figure 5 Final Entity Relationship Diagram

### 5.3.2 Storyboards

Storyboards are another useful diagram that developers use. Storyboards allow developers to design user interfaces of software, whether that be for mobile applications or websites. This makes coding the interfaces easier for developers as they already have a rough design for the interfaces that they need to create, instead of having to come up with the design on the go whilst coding. Storyboards also allow developers to represent how pages on a website or application will interact with each other through buttons, menu items, etc. They are also used for designing the general theme of the overall application or webpages, making it easier for developers to implement because they have a relatively clear vision of what the colour scheme of a webpage should look like.

During the planning phase I used storyboards to design the interface and general theme of the website. Doing this gave me a clear idea of how I would implement the HTML elements, such as the navbar, buttons and page structures. Creating storyboards also made styling the HTML using CSS much easier because I knew what the website’s theme would look like. A full list of storyboards I made for Back to Work can be found in the appendix (2.4 Storyboards). Below is one of the storyboards I created for the website.



Figure 6 'My Group' Page Storyboard

### 5.3.3 UML Diagrams

Probably the most commonly used diagram by developers is UML diagrams. UML (Unified Modelling Language) diagrams are used to visually represent the different components within a system. This helps developers when coding the application and how the code links together through classes, methods, etc. It also means they can visually see how the system should be structured which makes it clearer to implement within the code.

During the planning phase, I used UML diagrams and more specifically UI-Flow diagrams to plan how a user would how I though a user would navigate the website and what each interaction the user had with the website would do. UI-Flow charts are a great way of visually displaying the actions a user can take and what the system will do in response to those actions. Flowcharts and UI-Flow charts are also a great way to visually represent the logic of how the system works. These diagrams are also useful for visualising user stories and so after coming up with some suitable user stories, I design some UI-Flow charts to go along with them. Below is one of the UI-Flow charts I design to visually display how a user would interact with the ‘Chores’ page of the website, a full list of UI-Flow charts made for Back to Work can be found in the appendix (2.7 UI-Flow Charts).

Diagram, schematic

Description automatically generated

Figure 7 UI-Flow Chart of 'Chores' Page

### 5.3.4 Use Case Diagrams

Another way of visualising user stories is to design Use Case Diagrams. These diagrams are another great way of visualising how a user would interact with the interface of the system and how the system functions would react. Each function or webpage of the system is usually designed around a storyboard or functional requirement. This makes it easier for developers to implement some of the logic behind the system within the code. During the planning phase, I designed a use case diagram alongside the user stories and storyboards so that I had a clear idea of how I could go about implementing the different functions of the website. This diagram can be found in the appendix (2.8 Use Case Diagram).

## 5.4 Design Principles

When development teams plan projects, it is important for them to consider certain design principles to ensure the product being developed is of high standards. Having universal standards for developers within a team to follow makes working in teams a much easier task as everyone should be contributing work of the same standards. This avoids any issues with others being unable to understand the code or code being structured wrong. Sticking to these design principles will also ensure the final product is suitable for users to use and is user friendly.

### 5.4.1 Responsiveness

Back to Work is an interactive website that can be accessed by any user on any device that has access to the internet and a web browser. As a result of this, the website needed to be designed to work on not just desktop browsers but also mobile browsers. To do this, the website was designed with responsiveness in mind. When a website is responsive, it allows elements of the webpage to resize, reorder and structure itself based on the device being used, changing the appearance of the website to be more device friendly. Having the website responsive meant I did not need to focus on building two versions of the website but instead allowed me to focus on just one responsive site. Below is an example of how the website has been made to be responsive.

Graphical user interface, text, application

Description automatically generated

Figure 8 Responsive Login Page

### 5.4.2 User Focused Design

Throughout the development of the website, the user and their experience of using the website is always kept in mind. This was especially important when it came to the user interface of the webpages. If the user struggles to navigate the website, the user would be unlikely to return and so it was vital that the implementation of the webpages and their interfaces was easy to navigate, visually appealing and user interaction, such as inputting data and validating that data, was as simple as possible in order for the user to get the best experience possible.

### 5.4.3 Don’t Repeat Yourself

The Don’t Repeat Yourself (DRY) principle states that duplicate code should be eliminated via abstraction and automation to avoid waste code. Adding wasteful code adds to the maintenance of software and could lead to avoidable bugs (DevIQ, 2016). During development of Back to Work, this principle was followed to the best of my ability. To do this, code was structured appropriately to avoid messy code and reduce the risk of bugs appearing.

### 5.4.4 Security

The user’s security is the most important aspect of a website. To ensure user data was safe and secure throughout the website, only data that was necessary was retrieved from the database when needed and all sensitive data was encrypted appropriately. By encrypting sensitive data, such as passwords, the user’s data is protected when being transmitted from the database to the server and then the client and vice versa. The website also always checks if a user is logged in and if they have the correct permissions to access certain aspects of a page. If a user is not logged in, they are always redirected to the login page and all cookies are erased. All fields that require data input by the user are also validated extensively to ensure the correct data formats are being entered and no malicious data can enter the database and potentially cause serious harm.

# 6 Project Development

During the development of the project, each functional requirement that I set out during the project initialisation phase was split into five sprints that would last 3-5 weeks. This allowed me to prioritise some of the more important features that were necessary to satisfy the minimal viable product. Each sprint was split into smaller tasks to complete along with a checklist of smaller tasks that would need to be completed to finish the main task. This allowed for a clearly structured development process to take in order to complete the current sprint. Each full sprint along with its review can be found within the appendix of this report (2.9 Sprint Plans). Below is an outline of the development process that I took for each sprint.

## 6.1 Sprint Zero: Project Initialisation

Before starting work on the project, the project initialisation phase was used to come up with a project idea. From this, the idea of ‘Back to Work’ was born. The project initialisation phase was used to outline the product vision for my project (2.1 Product Vision) set up my development environment, perform a risk assessment (2.5 Risk Assessment) and generate some user stories (2.4 User Stories). Doing this meant I was prepared and ready to start planning how I would structure the development of the project in sprints of 3-5 weeks. This gave me a clear path to building a final working solution.

## 6.2 Sprint One: Website & Database Design

Sprint one was the start of the development of Back to Work and was completed during the first month of product development.

**Objective**

The objective of sprint one was to design and implement the different webpages that would be used for the website as well as the database structure. By the end of this sprint, a complete first draft of the website and database was expected.

**Development**

During sprint one I focussed on designing the website interface using storyboards and the user stories as a guide. Each functional requirement was broken down into webpages, such as a chores page for assigning chores and a rewards page for redeeming rewards. I then design a rough layout for each webpage that I thought was appropriate. By doing this, I was able to establish a general colour theme and layout for the website.

During sprint one I also focussed on designing the structure and data for the relational database I planned to use using entity relational diagrams. These diagrams allowed me to get a rough idea of how the data within the different tables would be stored and how it would relate to other tables. During this time, I also made the decision to use a remote MySQL server hosted by Amazon Web Services to host my database securely.

Once the website and database’s structure had been designed, I started work on implementing them. Using HTML & PHP I implemented first draft webpages that would be used. I also setup the remote MySQL server and implemented the different tables that I planned to use. During this stage, I made sure there was enough room for me to add new features easily to the webpages and database, in case during the development of the project new ideas for features developed or existing ideas developed further.

## 6.3 Sprint Two: User Accounts

Sprint two was started right after the completion of sprint one and lasted 6 weeks. This was due to taking a break during the Christmas period.

**Objective**

The objective of sprint two was to have a fully integrated registration and login system that supported the creation of admin and user accounts. This feature was of high priority due to the website being user focussed and needing a user to be logged in for it to operate properly.

**Development**

The first step of sprint two was to set up the registration system. To do this, I used a classic registration form containing all relevant fields for the user to input the required user details, such as an email, password and name. I then used JavaScript to validate this data client-side. By validating the data client-side it removed any possible risks of malicious or incorrect data reaching the server. Before sending the data to be processed by the server and added to the database, I ensured all sensitive data, such as passwords, was encrypted using a plugin for JavaScript called ‘CryptoJS’. I then used jQuery’s POST function to send the data to an API that would add the data to the database.

Graphical user interface, application

Description automatically generated

Figure 9 Registration Page

The second step of sprint two was to implement the login functionality. A simple login form was made for users to input their email and password. JavaScript and jQuery were then used to validate the input data. To validate the user’s inputted credentials, an API was sued to query the database to find the user’s email. If the email is found, the encrypted password stored in the database is returned and compared with the user inputted password. If they match, the user is logged in.

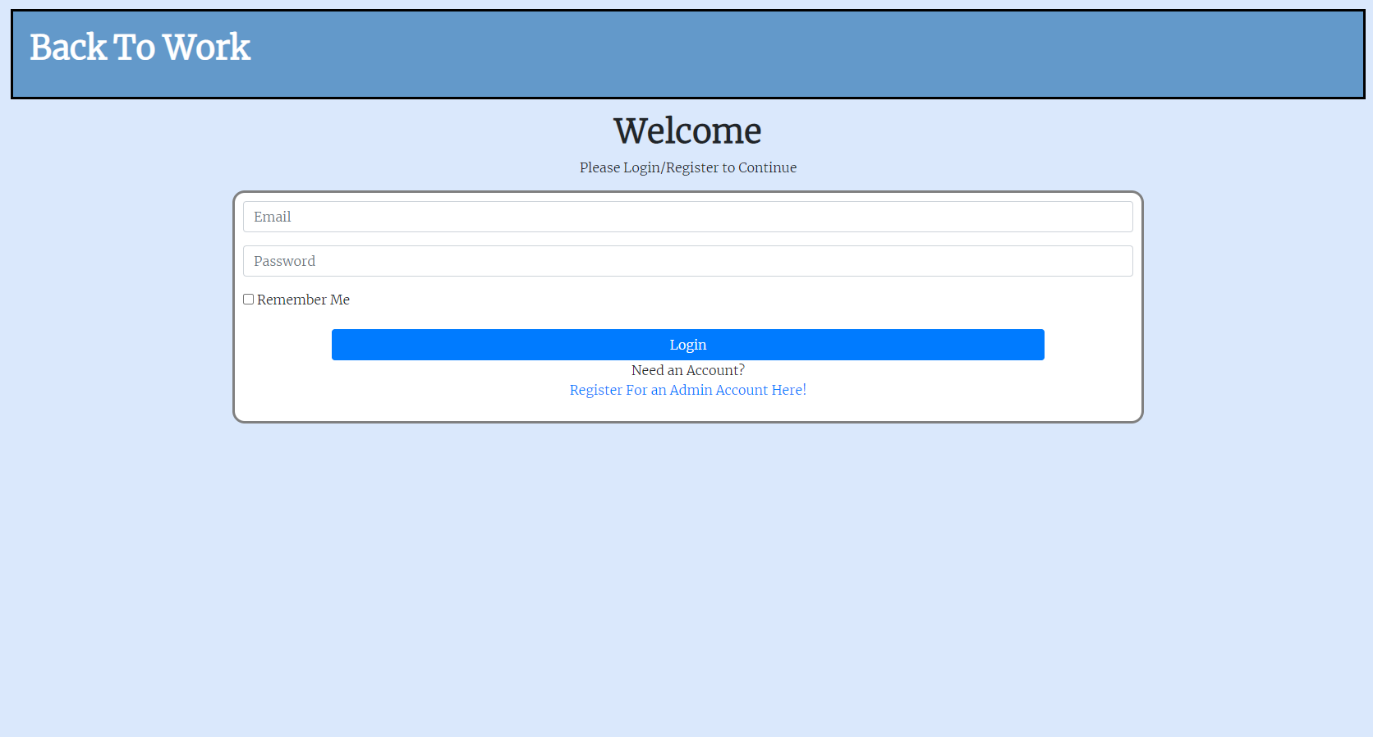


Figure 10 Login Page

The final step of sprint two was to set up the options for user accounts. These options included the ability for users to change their email and password or delete their account and group. This feature was of highest importance due to ensuring the user has full control of their data and following GDPR rules.

Graphical user interface

Description automatically generated with low confidence

Figure 11 My Account Page

## 6.4 Sprint Three: Chores

Sprint three was started after the completion of sprint two and lasted 5 weeks.

**Objective**

The objective of sprint three was to have a fully functional chores system where users could be added to a group and assigned a chore for them to complete within a given timeframe.

**Development**

The first step of sprint three was to implement a group system that allowed users who registered for an account to create a new group and be able to add new users to that group. When a new user registers for an account, they automatically become an admin. This is due to the age restriction of 14+ that I put in place to avoid any issues with younger users having too much control of the system. Admins are able to create new user accounts through the ‘My Group’ page which are automatically added to the admins group. They also have the option to assign an admin role to these new users if they wish to do so.

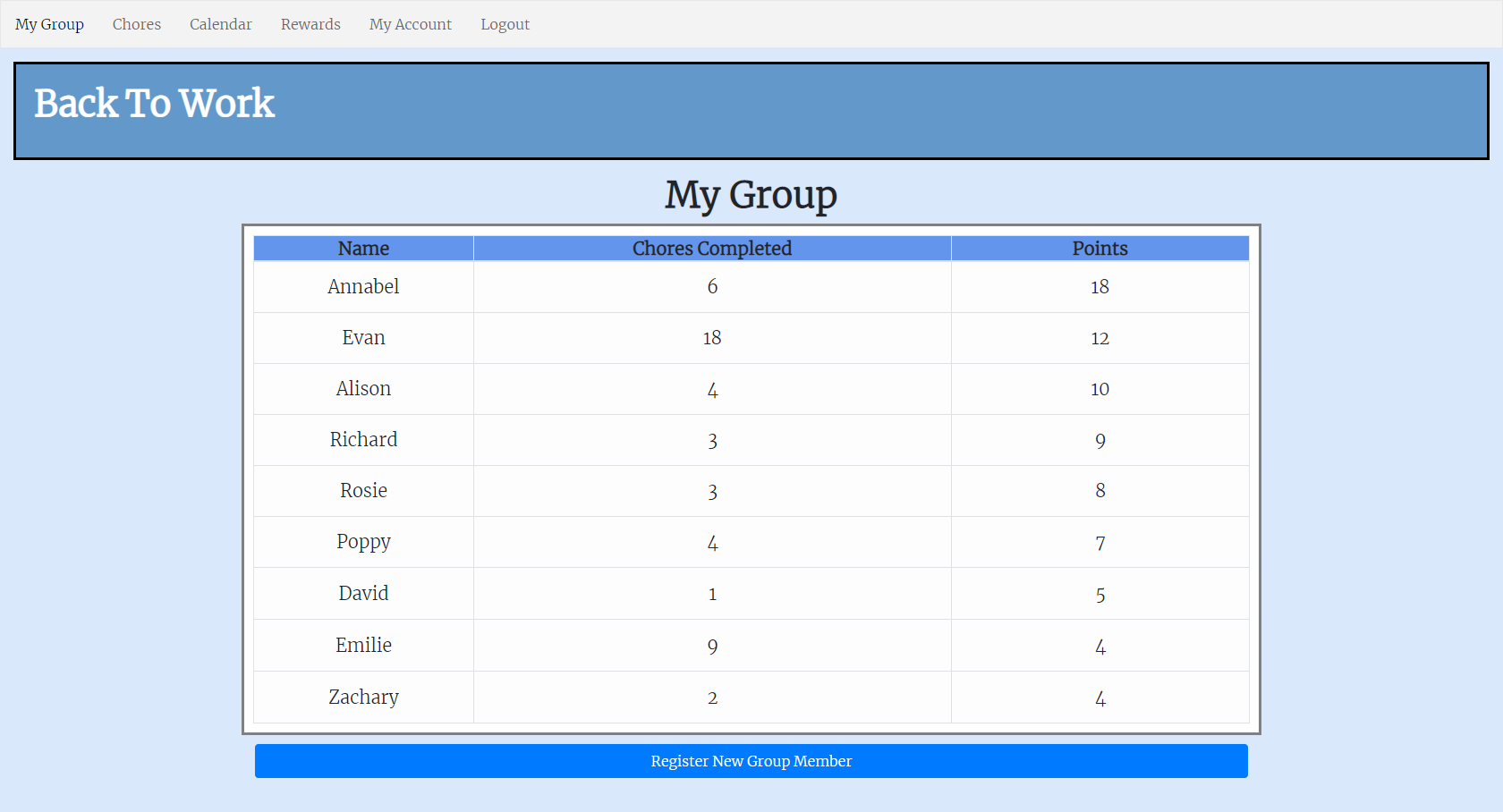


Figure 12 My Group Page

The second step of sprint three was to implement the chores system. Admins have the ability to create new chores for the group which they can then assign to users within their group. To do this, a simple form was made where the admin can input the name and description of the chore with a points and penalty value. This data is then validated using JavaScript and jQuery then sent to an API to be added to the database. Admins can then view the available chores that have not been assigned to a user yet. To assign a chore, the admin chooses a chore and selects a user within the group to assign the chore to. Admins have the ability to view all assigned chores within the group as well as their own. Normal users can only view the chores that they have been assigned to. Admins also have the ability to edit a chore’s details and reassign a chore to another user within the group.

Graphical user interface, application

Description automatically generated

Figure 13 Chores Page

## 6.5 Sprint Four: Rewards System

Sprint was started after the completion of sprint three and lasted 3 weeks.

**Objective**

The objective of sprint four was to have a fully implemented rewards system where users could earn points for completing chores on time and lose points for completing chores past their deadline. User would then be able to redeem those points for a physical reward.

**Development**

The first step of sprint four was focussed on implementing a way for admins to add new rewards for users to redeem. This was done using a from, JavaScript and jQuery for validation and an API to add the new reward to the system. These rewards were group specific and could only be redeemed if a user had enough points.

Graphical user interface, application

Description automatically generated

Figure 14 Form to Add New Reward

The second step of sprint four was to implement the points themselves. Each new user starts with zero points. When a user completes an assigned chore, an API is used to check if the chore being completed was completed on time or late. If the chore is completed on time, the points value of the chore is added to the user’s points in the database and the assigned chore is removed from the database and moved to the available chores. If the chore is completed late, the same process happens but instead of adding the points, the penalty value is subtracted from the user’s points instead.

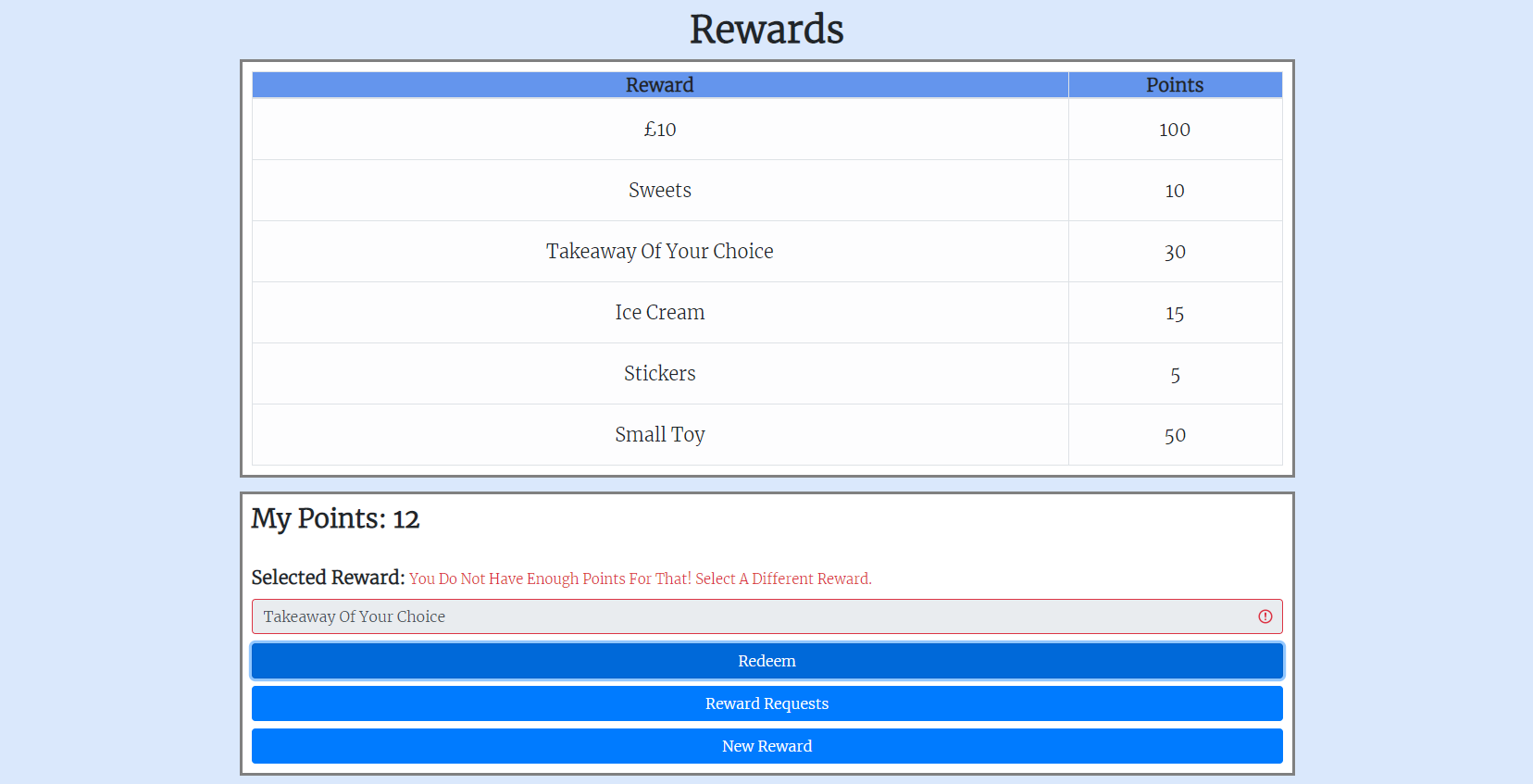


Figure 15 Select Reward

The last step of sprint four was to implement the rewards redemption process. Users can redeem their points for available rewards set up by an admin. A reward request is then added to the database for admins to either approve or decline the request. If the request is approved, the request status is set to ‘PROCESSED’. If the request is declined, the user’s points are refunded, and the status is set to ‘PROCESSED’.

Graphical user interface, application

Description automatically generated

Figure 16 Rewards Page

## 6.6 Sprint Five: Calendar

The final sprint was started after the completion of sprint four and lasted 5 weeks.

**Objective**

The objective of sprint five was to implement the functionality of the calendar by integrating assigned chore deadlines with the calendar and allowing admins to add new events to the calendar.

**Development**

The first step of sprint five was to build the calendar itself. To do this, I designed a calendar class that would generate a calendar for the current date. When the calendar page is loaded, the calendar class builds the calendar on the page for the user to view.

Table

Description automatically generated

Figure 17 Calendar Page

The second step for sprint five was to implement the events functionality. jQuery was used to detect when a user clicks on a date within the calendar. A Bootstrap modal popup then appears for the user to either view events for that date or add a new event to that date. User can view events but only admins can add events to the calendar. To do this, JavaScript and jQuery are used to fetch and validate the inputted data and then send the data to an API to save the event to the database. To show the events on the calendar, an API is called by the server to fetch the events for the group. The dates with events are then coloured to stand out and represent an event. When a user clicks on a date with an event, the vent is displayed within a table.

Graphical user interface, application

Description automatically generated

Figure 18 Calendar Event

The final step of sprint five was to integrate the deadlines of assigned chores with the calendar. To do this, a new calendar event was made whenever a chore is assigned. This event is then saved to the database along with the assigned chore’s id. Chore events are then displayed as an event on the calendar but within a separate table when viewed. Users of a group are able to see all chores assigned to other users within the group. Each chore event shows the details of the chore, including the name of the chore and who it is assigned to. When a chore is completed, the chore event is removed from the database.

Graphical user interface, application

Description automatically generated

Figure 19 Chore Event

## 6.7 Sprint Six: Clean Up & Testing

Sprint six was started after the completion of sprint five and lasted until the end of the project deadline. Sprint six was used only for tidying up the code and performing a usability study to test the website.

**Objective**

The objective of sprint six was to ensure the file and code structure was organised appropriately. Sprint six was also used for gathering feedback on the website through a usability study.

**Development**

Throughout the development of Back to Work, I made sure to keep the file structure and code as clean and tidy as possible, ensuring I used comments to explain the code as well. The aim of sprint six was to just go through the code, adding additional comments and ensuring code is indented correctly and variables are named appropriately. Doing this also gave me the opportunity to find any bugs that I was not aware of and remove any duplicated code.

The usability study gave me the opportunity to receive valuable feedback that I could use to improve the user experience of using Back to Work.

# 7 Usability Study

Usability studies are used by developers to gain valuable feedback from test users. The main goals of a usability study are to identify problems within the application, uncover opportunities for improvement and learn about the target user’s behaviour and preferences whilst using the application (Moran, 2019).

During sprint six, I performed a usability study to receive feedback on Back to Work. The study was a simple questionnaire that asked the user to perform a number of tasks that tested a feature of the website and provide feedback on this task. The user then had the opportunity rate the difficulty of the task performed, provide more written feedback on the task if they wished to do so, provide feedback on the design and layout of the website and provide feedback about the overall experience of using Back to Work and any suggestions they may have for improving the experience.

## 7.1 Results

Below are some of the results of the usability study. The full questionnaire and all responses are available in the appendix **(APPENDIX).**

Chart, pie chart

Description automatically generated

Figure 20 Editing Chore Details – Responses

Some of the feedback I received for editing chore details said that the popup that appears when selected a chore should be made cleaner when editing the chore. For example, the below response said that I should hide the option for assigning a chore when editing it instead of simply disabling it.

A picture containing text

Description automatically generated

Figure 21 Suggestion for Changing the Way Chores Are Edited

After receiving the above response, I changed the way chores were edited to make the UI cleaner and tidier. I also received some other similar response below.

Chart, pie chart

Description automatically generatedChart, pie chart

Description automatically generatedChart, pie chart

Description automatically generated

Figure 22 Responses Relating to Confirmation Popups

The above responses were very clear that some of the user interactions with changing any details needed a confirmation popup to ensure the user is making the right decision. As a result of this valuable feedback, I added a confirmation popup, similar to the way users are asked to confirm whether they wish to delete their group or not.

Overall, the feedback I received from performing this usability study was extremely useful for making changes to the website to be even more user friendly. This usability study also gave me a clear idea as to how I should expect potential users to interact with the website.

# 8 End-Project Report

## 8.1 End-Project Summary

Overall, the project I set out to undertake was a success. The main goal I had from the very beginning of the project to build some form of application that would allow users to create groups and assign chores to members of their group to complete was achieved. All features that I intended to add to the system were implemented successfully with only small alterations to what was initially set out during the planning phase.

## 8.2 Project Changes

During the development of the project, there were only a few changes that were made to the idea I had in mind. Probably the biggest change was the fact that I original planned to build a mobile application instead of a website. However, after starting to attempt to build the application to work on mobile, I decided to pursue an alternative method and instead chose to build a website instead. I made this decision due to two reasons. One, my experience in building mobile applications was very limited and next to none. This meant I would have to learn how to code mobile applications whilst trying to build my application. This was not ideal and so decided to build a website instead because I have much more experience doing this. The second reason I decided to build a website instead of a mobile application was because a mobile application would mean its restricted to mobile users whereas a website is accessible by any device with a web browser. By building a responsive website, users with any device could access the application from anywhere.

My initial idea for the application cantered around families assigning chores to members of the family. During development I decided to change this to make the website more accessible by a variety of users, such as families and student house groups. This is why the project name changed from ‘Mobile Family Planner’ to ‘Back to Work’, to target all users who wished to use the application.

Some additional features that I wanted to implement were never implemented due to time constraints. One of these features was the ability for users to receive notifications to remind them that they have chores to complete. This would have been a useful feature to have but was not possible to implement within the time I had.

# 9 Project Post-Mortem

## 9.1 Project Management Approach Evaluation

The agile methodology was chosen as the approach to this project. This allowed the author to organise the development of the project more efficiently, allowing for a smooth development process between implementing features. The use of Microsoft Office 365 Planner meant that preparing sprints in detail was a simple task. However, due to the author’s lack of experience using this service, sprints were sometimes hard to follow due to the way buckets and tasks are presented on the page. An alternative service for planning sprints and organising the product backlog would have been the GitHub Project Board due to the author’s experience using the service and the easy-to-follow structure of sprints being organised as ‘To Do’, ‘In Progress’ and ‘Done’, along with it’s simple drag and drop system.

Overall, the project was managed well. Throughout the project development, time was managed efficiently and only a few deadlines that were personally set were missed due to unavoidable circumstances. Designing appropriate planning documents, such as storyboards and user stories, made developing the application much easier and clearer to follow along. An effective version control system using GitHub also made implementing features hassle free and provided a smooth development process.

## 9.2 Technologies Evaluation

The decision to use PHP, HTML, MySQL and JavaScript for the backend, frontend and database languages, was the right choice. This was due to the extensive experience and understanding of the author working with these technologies in the past. This avoided any issues with having to learn new languages or using technologies, such as NodeJS and MongoDB, where the author’s experience is very limited, providing a clear path as to how to develop the website.

The decision to switch from working on a mobile application to a website was the right choice. This avoided wasting time learning how to develop mobile applications and instead allowed the author to focus on developing a website where the author has a much more extensive understanding of using the technologies used to develop websites.

## 9.3 Developer Performance Evaluation

Throughout the development of the project, most sprints were completed before the planned deadlines of 3-5 weeks. Only one sprint went past the deadline due to changes in the developer’s accommodation because of the COVID 19 Pandemic. However, this did not effect the development process as the developer ensured there was enough leeway between working on sprints in case a sprint had to be pushed back. An appropriate 15 hours of development each week was achieved for the majority of the time. There were some cases where the developer was unable to work during some weeks due to other commitments or unavoidable circumstances, however, this did not affect the development of the project greatly.

Overall, the developer’s performance was acceptable considering the circumstances. There were few mishaps during the development of the project and the developer was always focussed on producing the highest of quality work. The developer never lost interest in the project due to the personal connection to the project and the rewarding experience that would be gained during the process.

## 9.4 Future Work

The final solution developed for this project is working as intended and is fully functioning. The functional requirements set out at the beginning of the project were all implemented successfully. This however does not mean there is room for improvement and further expansion. For example, with more time, the developer would have liked to implement a notification system within the website but due to time constraints, this was not possible. This feature would be a potential feature that could be implemented sometime in the future if the project were to continue.

To further develop this project in the future, a mobile application could be made to be more mobile friendly. This app would work in parallel with the website but would be its on dedicated application. Although this is not necessary to do due to the website being responsive, a dedicated mobile application would allow mobile users to use the device offline if connection to the internet is lost. It would also make using push notifications easier and more accessible.

# 10 Conclusion

The main objective of this application was to build an interactive website that would allow users to create household groups where they could assign chores to members of the household to complete. This main objective has been achieved as well as all other objectives set out at the beginning of the project, such as the rewards system and calendar.

Overall, I am pleased with the outcome of the project. It has allowed me to further develop my understanding of Full Stack Development, providing me with invaluable experience that will be useful to me when working on future projects.

# References

Atlassian, 2021. *What is Agile?.* [Online]   
Available at: https://www.atlassian.com/agile

Atlassian, n.d. [Online].

Curry, L., 2019. *What is Office 365 Planner? A Beginner's Guide.* [Online]   
Available at: https://www.chorus.co/resources/news/what-is-office-365-planner-a-beginners-guide

Data Protection Act, 2018. Data Protection Act. *Data Protection Act 2018.*

General Data Protection Regulation, 2018. GDPR. *Official Journal of the European Union.*

Hostinger Tutorials, 2021. *What is WAMP?.* [Online]   
Available at: https://www.hostinger.co.uk/tutorials/what-is-wamp

MDN Web Docs, 2021. *What is JavaScript.* [Online]   
Available at: https://developer.mozilla.org/en-US/docs/Learn/JavaScript/First\_steps/What\_is\_JavaScript

PHP, 2021. *PHP.* [Online]   
Available at: https://www.php.net/manual/en/intro-whatis.php

# Appendix

## Appendix 1: User Guide

The following is a general user guide for users to follow along.

## Appendix 2: Project Management

### 2.1 Product Vision

Back to Work is an interactive website that allows users to assign chores for completion to a member of a group. This system is aimed at families who struggle to get their children to their chores or students who are sharing a house and wish to have an organised chore roster. Back to Work uses a reward system where points can be earned for completing a chore and then redeemed for a reward of the user’s choice. This acts as an incentive for kids or housemates to complete their chores. Users also have access to a calendar for being more organised by planning their week ahead and keeping track of when chores need to be completed.

### 2.2 Functional Requirements

Below are the functional requirements that were outlined at the start of the project during the planning phase.

Text

Description automatically generated

Text

Description automatically generated

Figure 23 Functional Requirements

### Non-Functional Requirements

Text

Description automatically generated

Figure 24 Non-Functional Requirements

### 2.4 User Stories

Text

Description automatically generated

Figure 25 User Stories

### 2.5 Risk Assessment

Table

Description automatically generated

Figure 26 Risk Assessment

### 2.6 Storyboards

Graphical user interface

Description automatically generated

Figure 27 Registration Page Storyboard

Graphical user interface

Description automatically generated

Figure 28 Login Page Storyboard

A picture containing table

Description automatically generated

Figure 29 My Group Storyboard

Graphical user interface, application

Description automatically generated

Figure 30 Chores Storyboard

Graphical user interface

Description automatically generated

Figure 31 Rewards Storyboard

Graphical user interface, calendar

Description automatically generated

Figure 32 Calendar Storyboard

Graphical user interface, application

Description automatically generated

Figure 33 My Account Storyboard

### 2.7 UI-Flow Charts

Diagram

Description automatically generated

Figure 34 Calendar UI-Flow Chart

Diagram, schematic

Description automatically generated

Figure 35 Chores UI-Flow Chart

Diagram

Description automatically generated

Figure 36 Rewards UI-Flow Chart

Diagram

Description automatically generated

Figure 37 Login & Registration UI-Flow Chart

Diagram

Description automatically generated

Figure 38 My Account UI-Flow Chart

Diagram

Description automatically generated

Figure 39 My Group UI-Flow Chart

A picture containing map

Description automatically generated

Figure 40 UI-Flow Chart in Full

### 2.8 Use Case Diagram

Diagram

Description automatically generated

Figure 41 Use Case Diagram

### 2.9 Sprint Plans

The following are the plans and reviews for each of the seven sprints (0-6).

#### 2.9.1 Sprint Zero: Project Initialisation

Text

Description automatically generated

Figure 42 Sprint Zero - Plan & Review

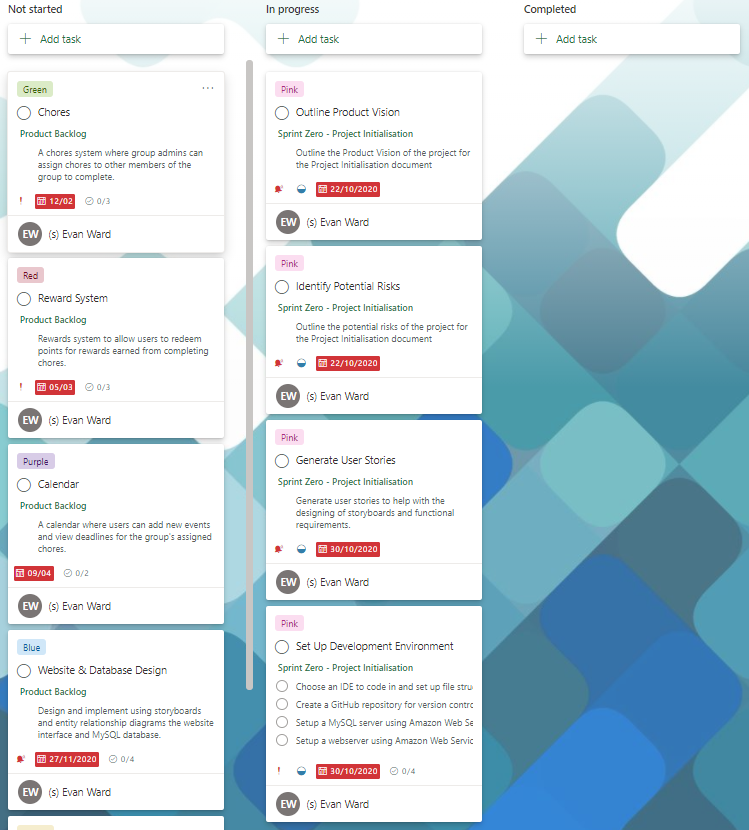


Figure 43 Sprint Zero - Planner Board

#### 2.9.2 Sprint One: Website & Database Design

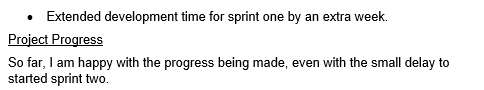
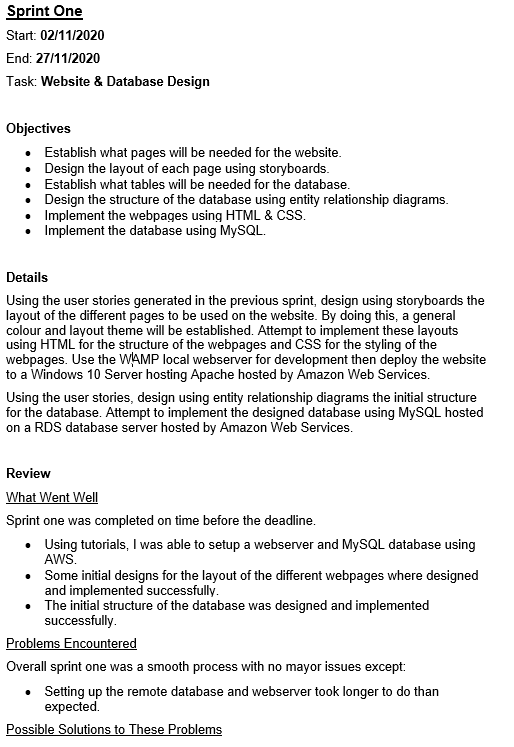


Figure 44 Sprint One - Plan & Review

Graphical user interface, text, application

Description automatically generated

Figure 45 Sprint One - Planner Board

#### 2.9.3 Sprint Two: User Accounts

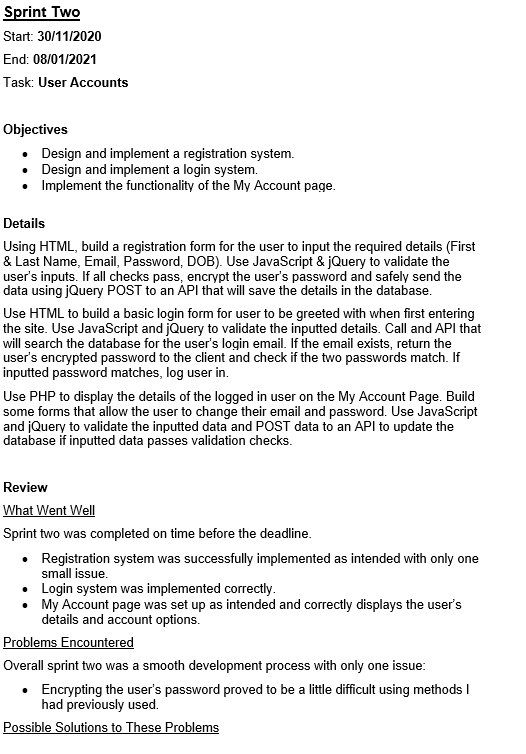


Figure 46 Sprint Two - Plan & Review

Graphical user interface, text, application, chat or text message

Description automatically generated

Figure 47 Sprint Two - Planner Board

#### 2.9.4 Sprint Three: Chores

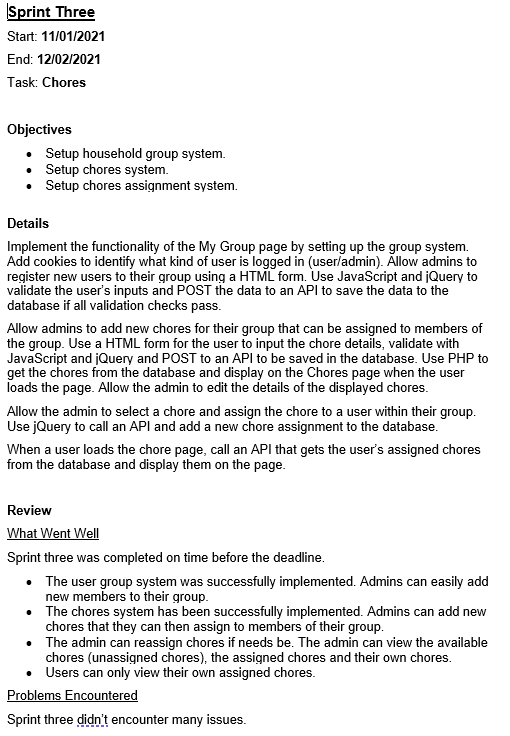
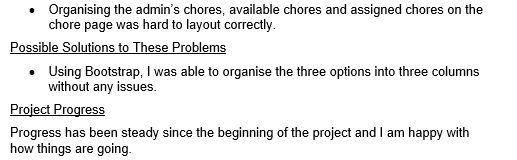


Figure 48 Sprint Three - Plan & Review

Graphical user interface, text, application

Description automatically generated

Figure 49 Sprint Three - Planner Board

#### 2.9.5 Sprint Four: Rewards System

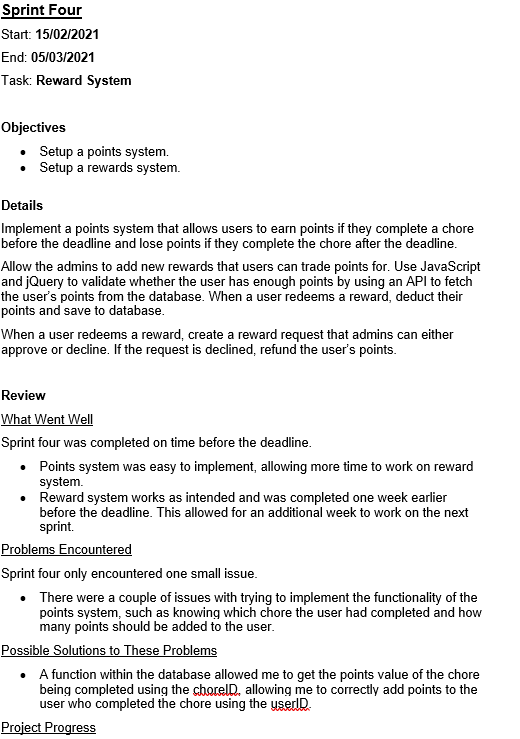
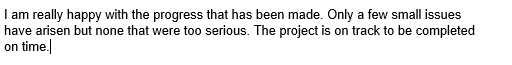


Figure 50 Sprint Four - Plan & Review

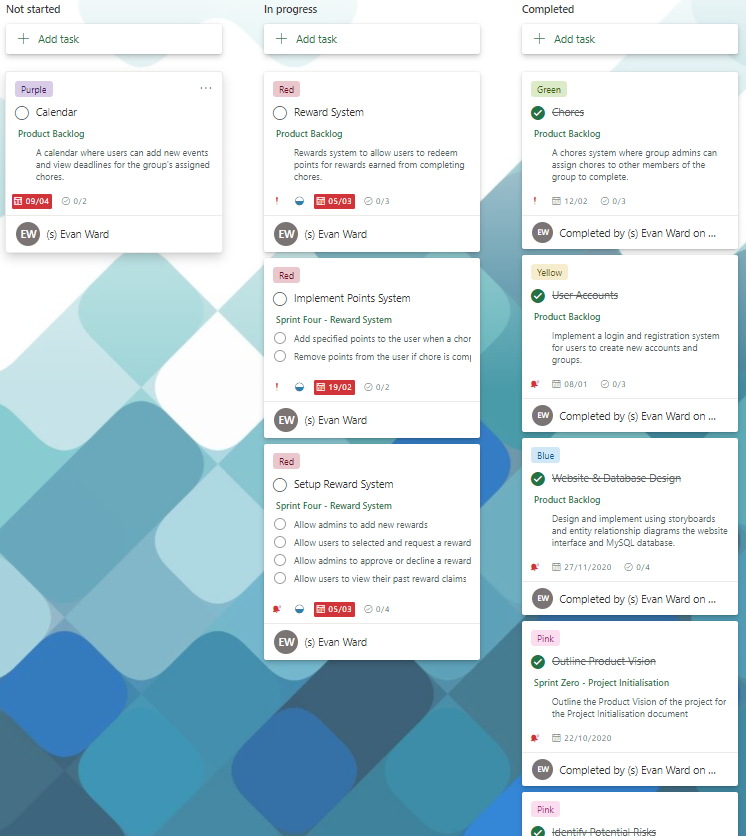


Figure 51 Sprint Four - Planner Board

#### 2.9.6 Sprint Five: Calendar

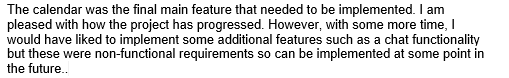
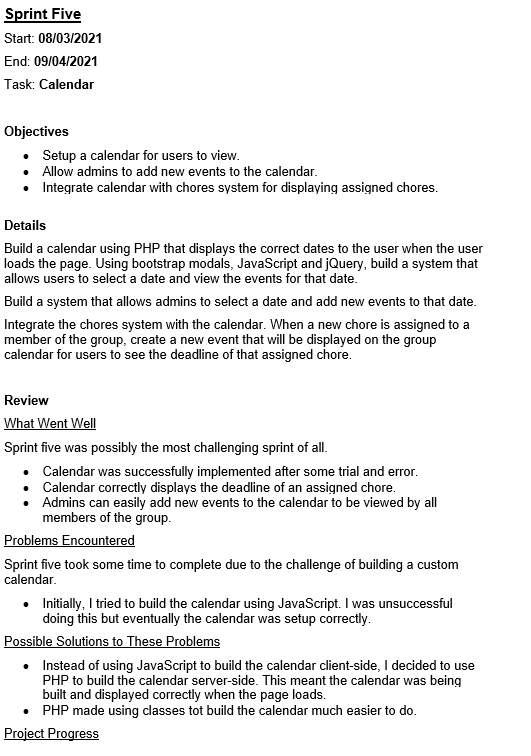


Figure 52 Sprint Five - Plan & Review

Graphical user interface, application

Description automatically generated

#### 2.9.7 Sprint Six: Clean Up & Testing

Text, letter

Description automatically generated

Figure 53 Sprint Six - Plan & Review

A picture containing graphical user interface

Description automatically generated

Figure 54 Sprint Six - Planner Board