



Coeliac Disease Symptom Tracker

Functional Specification

Student: Niamh Coleman (C00205225)

Supervisor: Joseph Kehoe

Date: 30/11/2018

Abstract

The aim of this document is to illustrate the internal workings of this application to the reader. After completing this document, the reader should be able to successfully create the application to the described specifications. This document will also describe all application functionality and how the completed application will look. The primary target user of this application is a teenage coeliac patient. Being a coeliac is burdensome, restrictive and challenging. This application will provide information that is crucial to their continued wellbeing by allowing the individual to keep track of the symptoms that they possess and the severity of those symptoms.

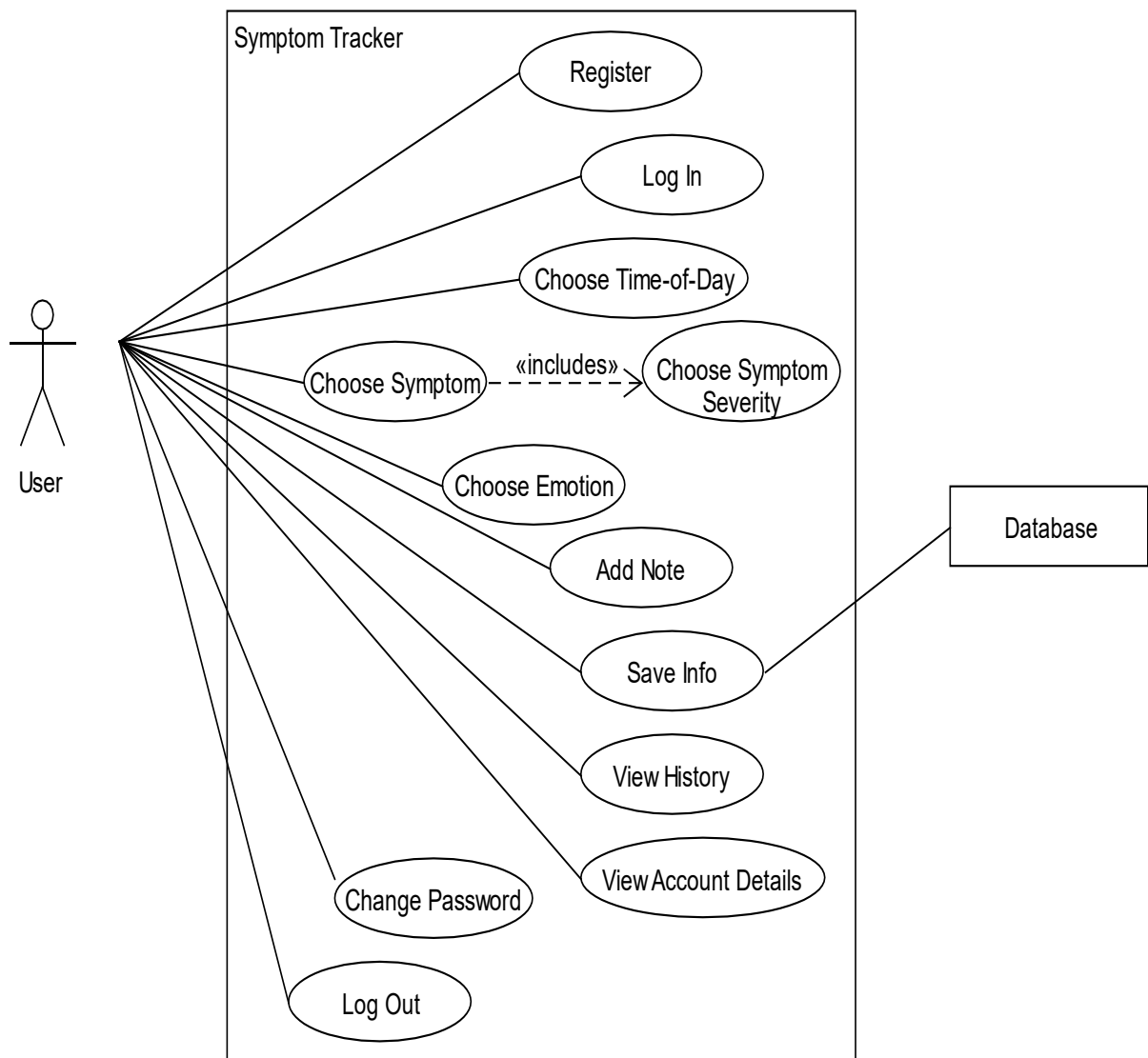
Table of Contents

Abstract	2
Introduction to the Application	4
Use Case Diagram.....	5
Brief Use Cases	6
Architecture Diagram.....	11
Wireframes.....	12
Project Development Timeline	24
Gantt Chart	25
Supplementary Specification (FURPS)	27

Introduction to the Application

The symptom tracker will provide users with an application that is user friendly and intuitive to use. The user will have the ability to add both positive and negative symptoms once a day. The user will be able to add any additional symptoms that are not already present in the database. When adding a symptom, the user will be able to enter the severity of the symptom. This will be particularly helpful for showing to a doctor. The application will encourage adherence to a gluten-free diet. It will do this by illustrating to the user the lack of symptoms that occur when a coeliac-friendly lifestyle is adapted. It also gives the user an opportunity to take note of any spikes in symptoms and may help them correlate that information e.g. with a restaurant falsely advertising coeliac friendly meals.

Use Case Diagram



Brief Use Cases

Register

Name	Register
Actors	User, Mobile Application, Database
Preconditions	The user has successfully downloaded the application.
Activity	This use case begins when the user downloads the application and chooses to register with the system. The user will enter the information that the system requires i.e. name, email address and password. The user is required to confirm their password. This information will be securely stored in the database.
Consequences	The user is registered with the system.
Alternative(s)	<ol style="list-style-type: none">1. The user enters invalid information e.g. user enters an incorrectly formatted email address.2. The user confirms their password and the passwords do not match upon comparison.

Log In

Name	Log In
Actors	User, Mobile Application, Database
Preconditions	The user has successfully registered with the system.
Activity	This use case begins when a registered user wants to log into the application. The user enters their email address and password. This information is validated by the system. If this data is correct the user gains entry to the application.
Consequences	The user successfully logs into the application and is presented with the home screen.
Alternative(s)	<ol style="list-style-type: none">1. The user enters incorrect information and must re-enter this data.2. The user enters incorrect information four times and is locked out of the application. At this point the user must follow the given steps to unlock their account.

Choose Time of Day

Name	Choose Time of Day
Actors	User, Mobile Application
Preconditions	The user has successfully logged into the application.
Activity	This use case begins when a logged in user wants to log some symptoms. The user chooses a time of day from a list. The options available to the user are: morning, afternoon, evening and night. This information is recorded by the application.
Consequences	The user has successfully chosen a time of day and recorded it with the application.

Choose Symptom

Name	Choose Symptom
Actors	User, Mobile Application
Preconditions	The user has successfully logged into the application.
Activity	This use case begins when a logged in user wants to add a symptom. The user chooses a symptom from a list. The user also chooses the severity of the symptom from a choice of low, moderate and severe. These severities are represented through colours; yellow (low), orange (moderate) and red (severe). This information is recorded by the application.
Consequences	The user has successfully recorded a symptom and its severity with the system.

Choose Emotion

Name	Choose Emotion
Actors	User, Mobile Application
Preconditions	The user has successfully logged into the application.
Activity	This use case begins when a logged in user wants to add an emotion. The user chooses an emotion from a list. The list consists of happy, okay and unhappy. This information is recorded by the application.
Consequences	The user has successfully recorded an emotion with the system.

Add Note

Name	Add Note
Actors	User, Mobile Application
Preconditions	The user has successfully logged into the system. The user has added various symptoms and emotions.
Activity	This use case begins when a logged in user wishes to add information about a day. The user may add anything that they feel relevant e.g. <i>I ate chicken nuggets in McDonalds today for lunch.</i>
Consequences	The user successfully adds a note to a day.

Save Info

Name	Save Info
Actors	User, Mobile Application, Database
Preconditions	The user has added all elements that make up an entry i.e. time of day, symptom(s), emotion and notes. The user now wishes to save the information.
Activity	This use case begins when a logged in user has successfully added all the information that they wish to add for the present day. The user presses the 'save' button and the information is stored in the database.
Consequences	The user successfully saves the information.
Alternative(s)	1. The user has not entered all the required information e.g. no symptoms/emotions have been entered. The user cannot press save until all required information has been added.

View History

Name	View History
Actors	User, Mobile Application
Preconditions	The user has successfully logged into the application.
Activity	This use case begins when a logged in user enters the history page in the application. From here, the user may view symptoms that they logged in the past. Users may not access days in the future.
Consequences	The user is presented with the calendar and in turn, a list of symptoms associated with a specific day.

View Account Details

Name	View Account Details
Actors	User, Mobile Application
Preconditions	The user has successfully logged into the application.
Activity	This use case begins when a logged in user enters the account page in the application and wants to view the information associated with their account. The user chooses the option “view account info” from among the options on the page.
Consequences	The user is presented with the name and email address associated with the account.

Change Password

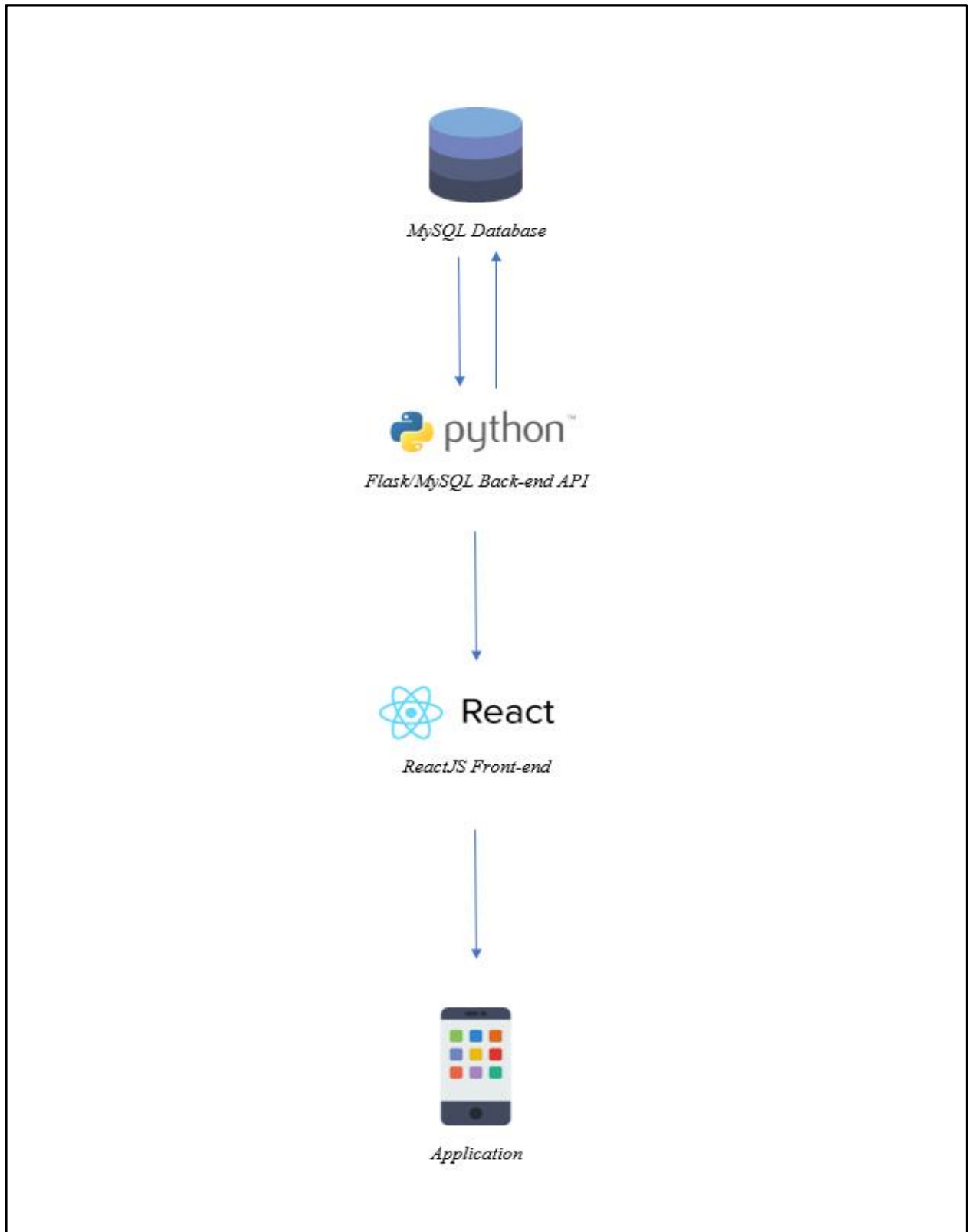
Name	View Account Details
Actors	User, Mobile Application
Preconditions	The user has successfully logged into the application.
Activity	This use case begins when a logged in user enters the account page in the application and wants to change the password associated with their account. The user chooses the option “view account info” from among the options on the page. The user must enter the current password associated with the account and the new password that they wish to use.
Consequences	The user successfully changes the password associated with the account
Alternatives	The user does not enter the correct current password and cannot change the password associated with the account.

Log Out

Name	Log Out
Actors	User, Mobile Application
Preconditions	The user wishes to log out of the application.
Activity	This use case begins when a logged in user wishes to log out of the system. The user chooses the ‘log out’ button and is logged out.
Consequences	The user successfully logs out of the application.

Architecture Diagram

The architecture for this application consists of a ReactJS user interface which is connected to a MySQL database via a Python MySQL back-end API using Flask.

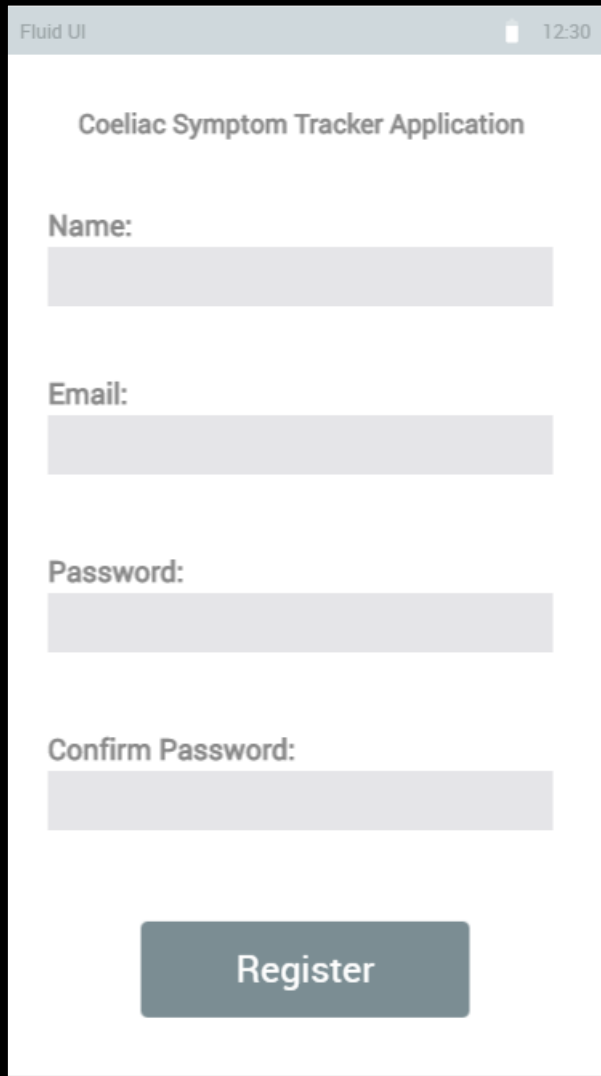


Wireframes

Wireframing is a technique used to plan the layout and structure of interface elements. It aims to provide an understanding of the application before the development phase gets underway. The following wireframes make up the user experience of the application.

Registration

The user sees the registration page first upon downloading the application. The user sees a simple form (figure 1) which requires their name, email address and a password. The user must confirm their password before they can register for the application.



The image shows a mobile application registration screen. At the top, there is a status bar with 'Fluid UI' on the left and '12:30' on the right. Below the status bar, the title 'Coeliac Symptom Tracker Application' is centered. The form consists of four vertically stacked input fields, each with a label to its left: 'Name:', 'Email:', 'Password:', and 'Confirm Password:'. Each input field is a light gray rectangle. At the bottom of the screen, there is a dark gray button with the text 'Register' in white.

Figure 1. Registration Screen

Log In

Once a user has successfully registered with the application (figure 1), they can log in. The following screen (figure 2) is the log in screen that will be presented to the user.

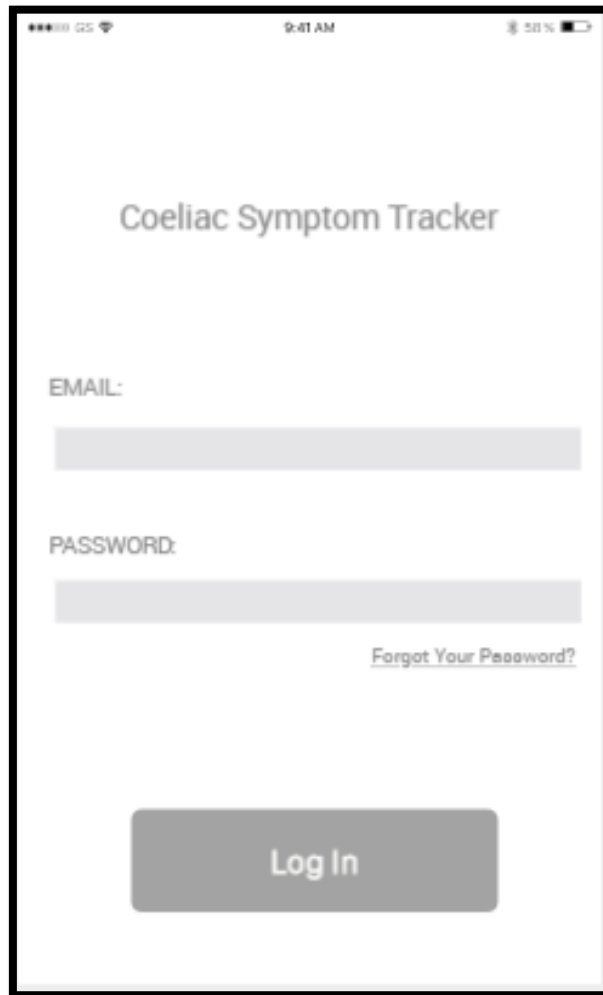


Figure 2. Log In Screen

Symptom Tracking Tab

During the Erasmus+ project, the user interface of the application underwent numerous refinements and changes. The primary function of the application i.e. to log symptoms, went through substantial changes. This affected the timeline of the development process as the refinement took place March 2019, at which point the user interface had been completed and had to be started again. The following documents both the initial and final design for the interface.

Initial Design

This screen represents the primary function of the application, allowing the user to choose the symptoms and emotions that they would like to log. The page will show the current day by default. However, users may view past days if they choose. Users may not choose future days. Users select symptoms and emotions by pressing on the symptom/emotion that they wish to add. Users may also add any relevant notes to the day. The user may add to the application as many times as they like per day, by choosing from one of the time options (i.e. morning, afternoon, evening or night).

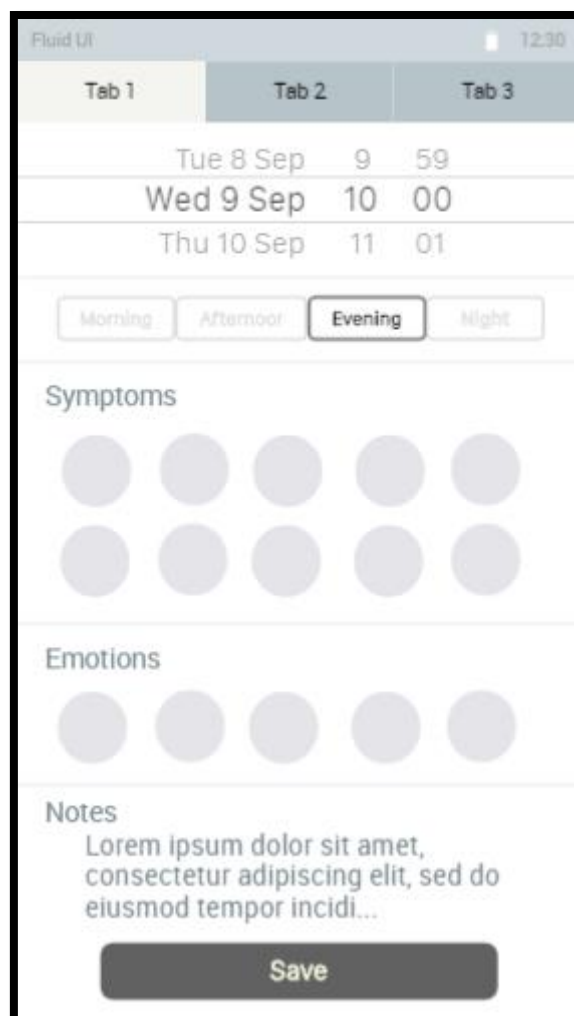


Figure 3.1. Symptom Tracking

When a user presses and holds down a symptom they may select the severity of the symptom. Symptom severity ranges from low (yellow), moderate (orange) and severe (red). This will help the application to collect more meaningful data. If the user just taps the symptom, a moderate severity is automatically applied. Figure 3.3 illustrates when symptom has been chosen.

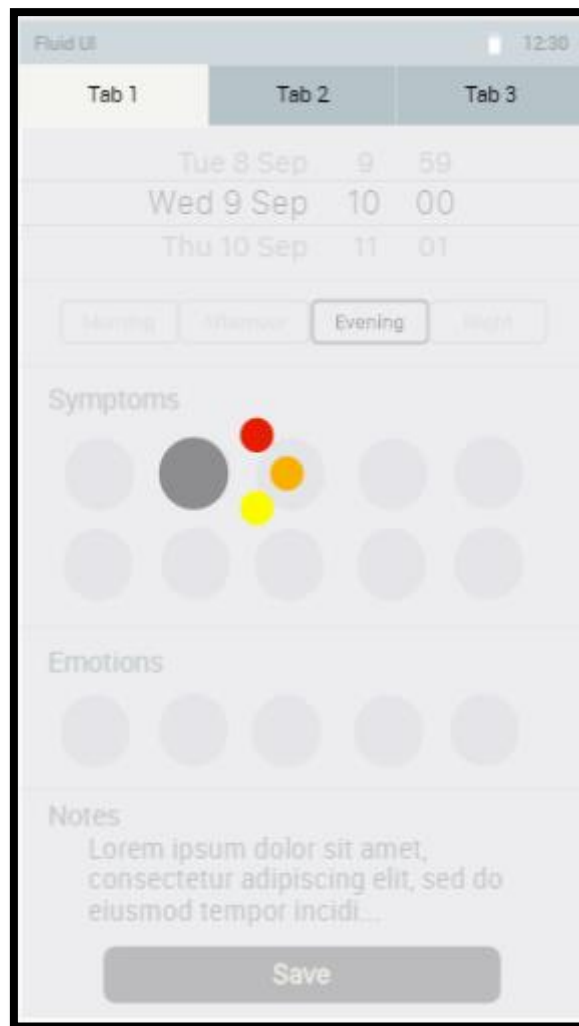


Figure 3.2. Choosing Symptom Severity

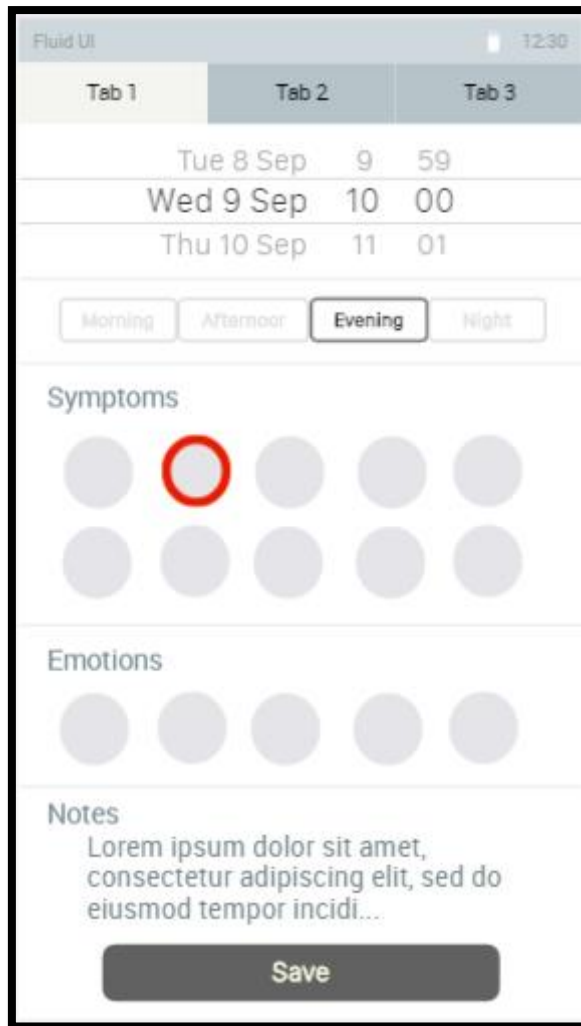


Figure 3.3. Severe Symptom Chosen

Final Design

Following feedback that had been received during the Erasmus+ project and advice received from user interface developers, the logging process was altered to take place through a multi-step form. This simplifies the process and makes the application easier to use. The form consists of the steps time of day, symptoms, emotion and notes.

The first step in the form requires the user to choose the time of day associated with the symptoms being logged. The initial design allowed the user to choose a specific date as well as a time of day. However, this was altered in the final design due to feedback from user interface designers. The final design is much simpler to use and encourages the user to continuous input their symptoms. Figure 4 illustrates the first step in the form.

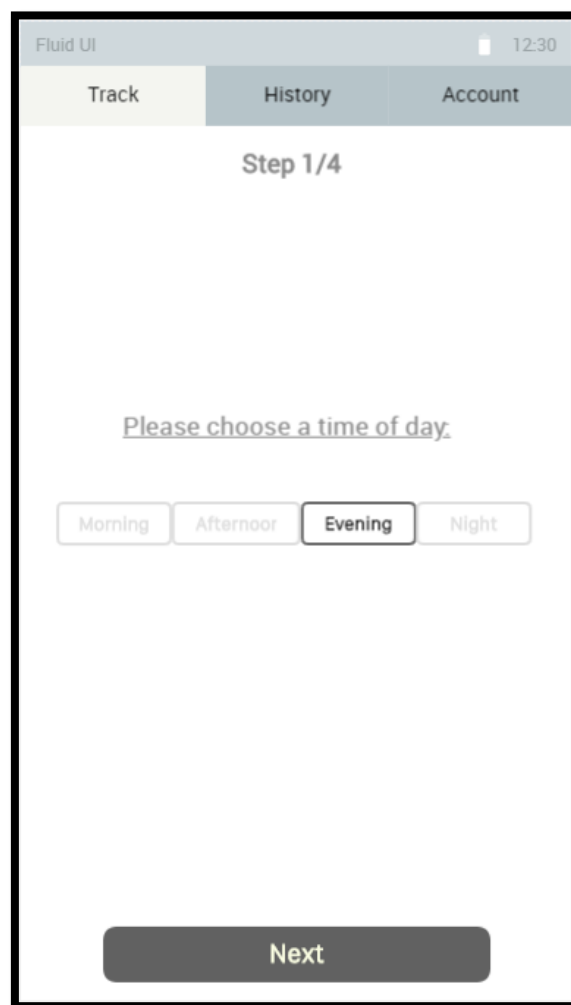
The image shows a mobile application interface for the first step of a tracking form. At the top, there is a header with the text "Fluid UI" on the left and "12:30" on the right. Below the header is a navigation bar with three tabs: "Track" (highlighted in light green), "History", and "Account". The main content area is titled "Step 1/4" and contains the instruction "Please choose a time of day." Below this instruction are four buttons: "Morning", "Afternoon", "Evening" (which is selected and highlighted with a dark border), and "Night". At the bottom of the screen is a large, dark grey button labeled "Next".

Figure 4. First step of the tracking form.

The second step in the form allows the user to choose the symptoms they want to log. All the symptoms are listed here. The user chooses only the symptoms that they want to log. The severities are defined using colour representations i.e. yellow (low), orange (moderate) and red (severe).

The screenshot shows a mobile application interface for tracking symptoms. At the top, there is a status bar with 'Fluid UI' and '12:30'. Below that is a navigation bar with three tabs: 'Track' (highlighted), 'History', and 'Account'. The main content area is titled 'Step 2/4' and contains the question 'What are your Symptoms Today?'. There are four symptom categories listed: 'Acne', 'Nausea', 'Joint Pain', and 'Headache'. Each category has three colored circles (red, orange, yellow) for selection. A 'Next' button is located at the bottom of the screen.

Figure 4.1. Second step of the tracking form.

The third step of the form allows the user to choose the emotion that they want to associate with the entry. The user can choose from the options of happy, okay and unhappy.

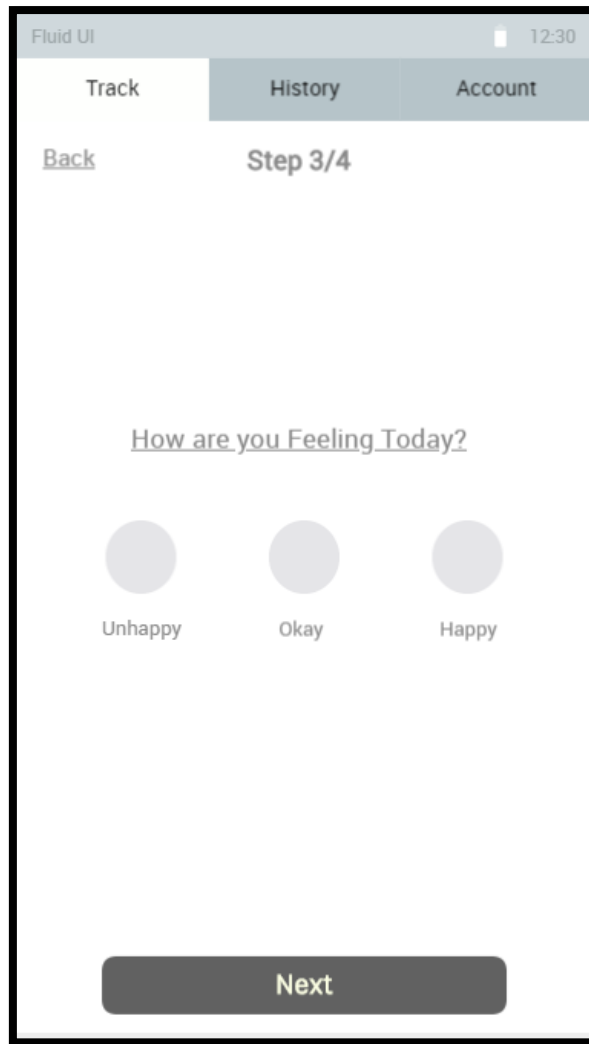


Figure 4.2. Third step of the tracking form.

The fourth (and final) step in the form allows the user to enter any notes that they wish to associate with the current entry. The user can add any relevant information such as eating something out of the ordinary.

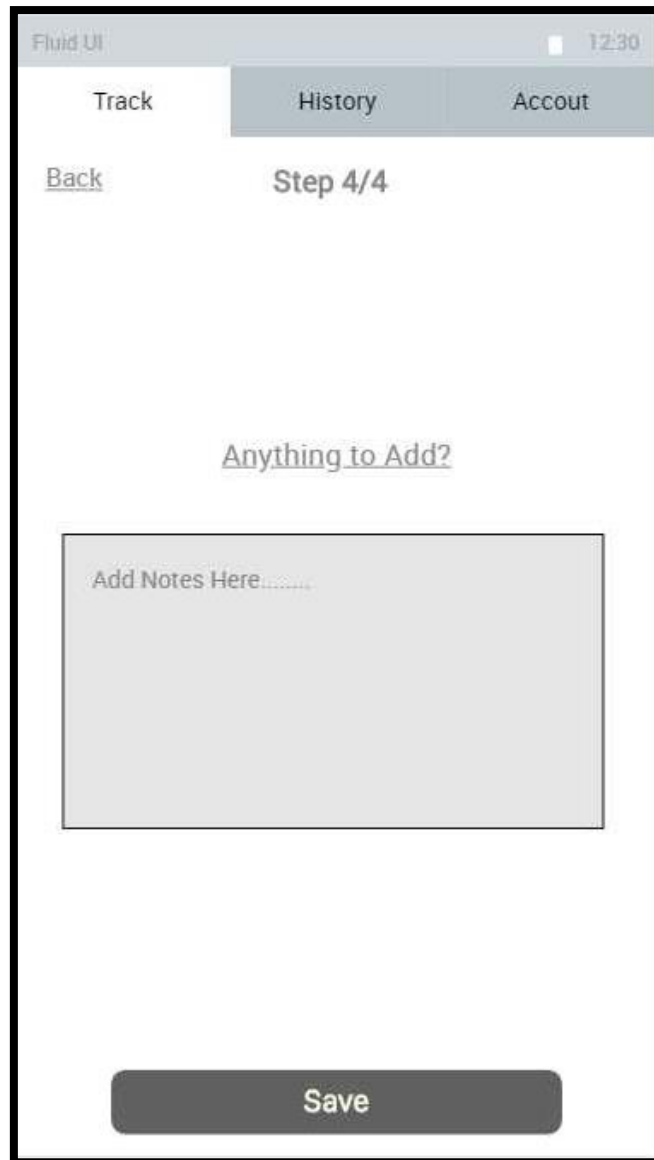


Figure 4.3. Fourth step of the tracking form.

The final design for the symptom tracking form was an improvement on the original in various ways. The addition of a multi-step form increases the simplicity and usability of the function and improves on the cluttered look and feel of the original. The Erasmus+ application is targeted primarily towards teenagers; however coeliac disease can affect individuals at any age and the multi-step form makes it easier for everyone to use and understand.

History Tab

The user may also access the history tab (figure 5) and see a summary of whatever day they choose. The user may not access future days. When a user selects a time from the list, they will see the view of that day (figure 5.1).

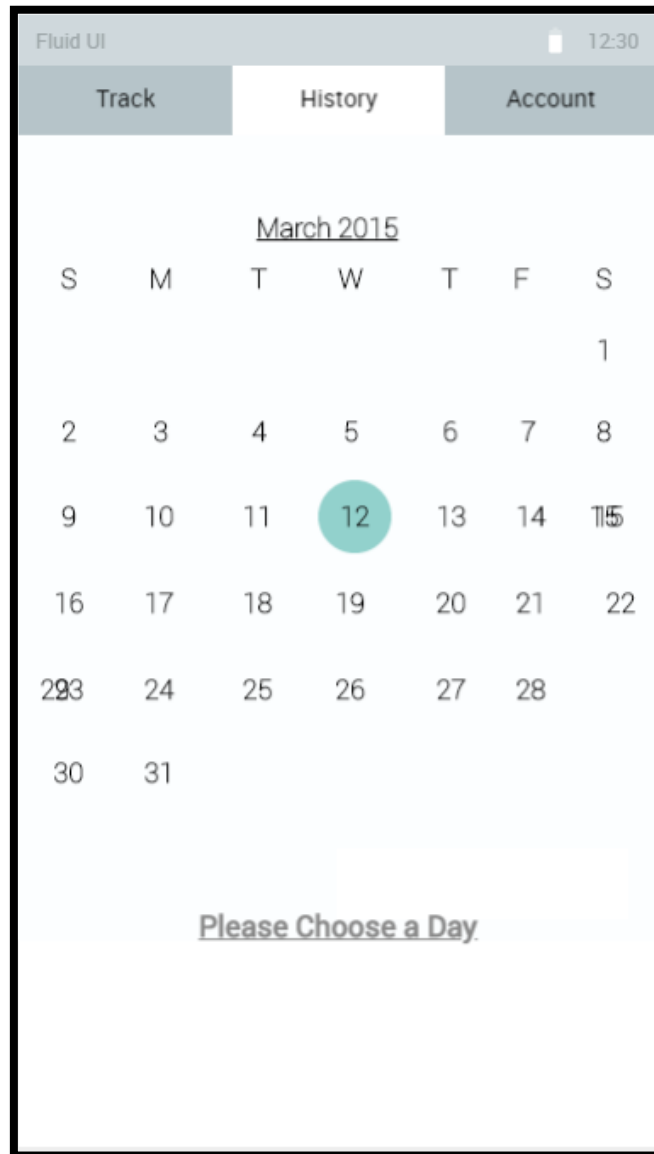


Figure 5. History Tab

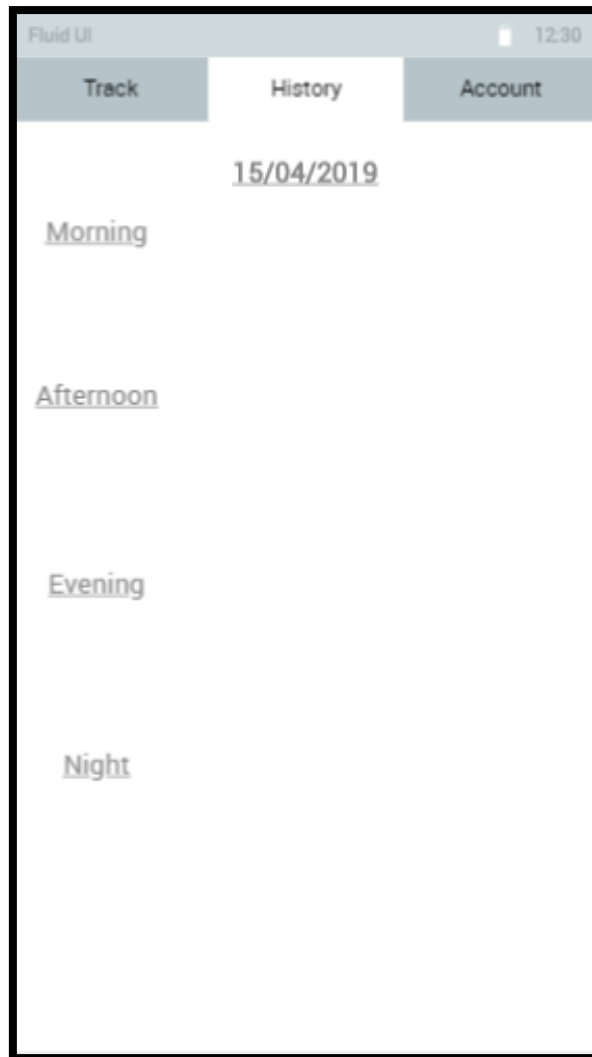


Figure 5.1. History Screen Day View

Account Tab

The user may access the remaining features using the account tab. The user may choose from the options of viewing account information, changing account password or logging out.

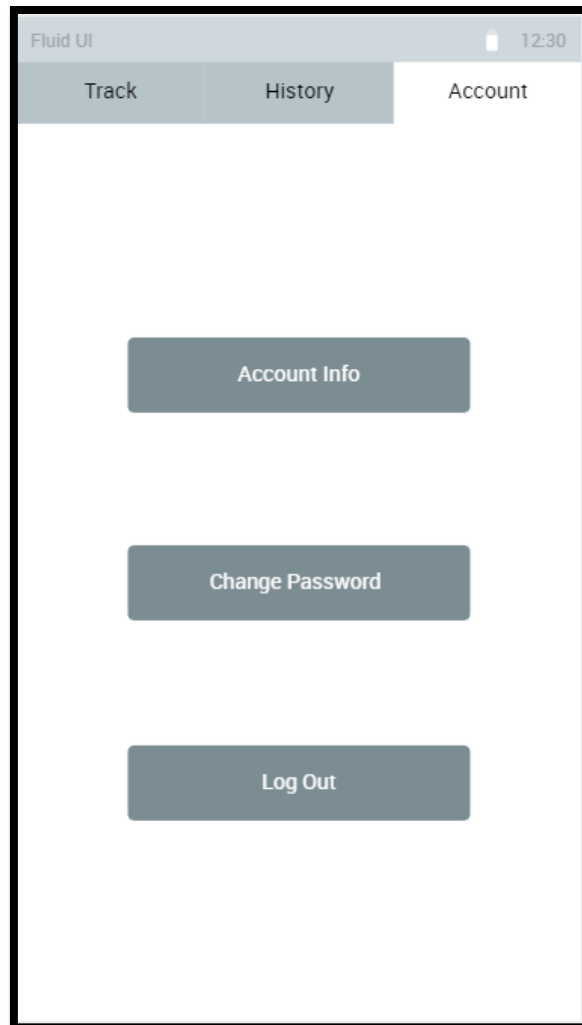


Figure 6. Account tab

Project Development Timeline

The project is taking an agile approach, with development taking place over four iterations/sprints. The following Gantt chart outlines the timeline associated with each iteration. Due to the user interface changes that took place in February 2019, the progress that had already been made with the application i.e. a complete user interface, was no longer relevant and as such development had to begin again. The iterations below illustrate the resulting timeline.

Iteration One (Feb 15th – Mar 1st)

Over the first iteration, development will begin to take place. This will include the initial act of becoming comfortable with the development technologies involved and the formation of the back-end.

As such, the first iteration should produce the following deliverables:

- Set up a MySQL database.
- Familiarisation with development technologies i.e. ReactJS, Python MySQL, Postman.
- Initial user interface layout and navigation development.
- Write API calls for history tab.

Iteration Two (Mar 1st – Mar 15th)

Over the second iteration, most functions will be developed.

As such, the second iteration should produce the following deliverables:

- The history tab and associated functionality.
- Step two of the tracking form (symptoms).
- Step three of the tracking form (emotions).
- Write API calls for account tab.

Iteration Three (Mar 15th – Mar 29th)

The third iteration will consist of the development of all remaining functions.

This iteration should produce the following deliverables:

- Step four of the tracking form (adding a note).
- Step one of the tracking form (time of day).
- Write remaining API calls.

Iteration Four (Mar 29th – Apr 5th)

The final iteration will consist of bridging the gap between the completed API and the completed user interface, as well as dealing with any tasks that had to be pushed into the following sprint previously. This iteration will also contain the testing of the API and the completed application.

- Test the completed API using Postman.
- Connect the user interface to the API.
- Test the functionality of the completed application.
- Deployment of the API (PythonAnywhere)
- Deployment of the UI (Github Pages)

Gantt Chart

A Gantt chart is a graphic representation of the duration of tasks within a project. The following Gantt chart illustrates the timeline of this project over the entire year i.e. from September 2018 until submission April 2019. The purpose of this Gantt chart is to show the continuous work that took place on this project over the year, as well as various deadlines and deliverables. It will allow the reader to have a detailed understanding of the timeline of various aspects of the projects. The Gantt chart is colour coded for the readability of the user. The legend is as follows:

- **Green** –trips taken with the *Erasmus+* project.
- **Blue** – deliverables consisting of documentation
- **Grey**- work that was done prior to specification changes, this work was not used in the final application. Regardless, knowledge gained from this work contributed to the project.
- **Orange**- The four iterations where the bulk of the work was carried out after final changes were made to the project spec in February 2019 (as detailed above).

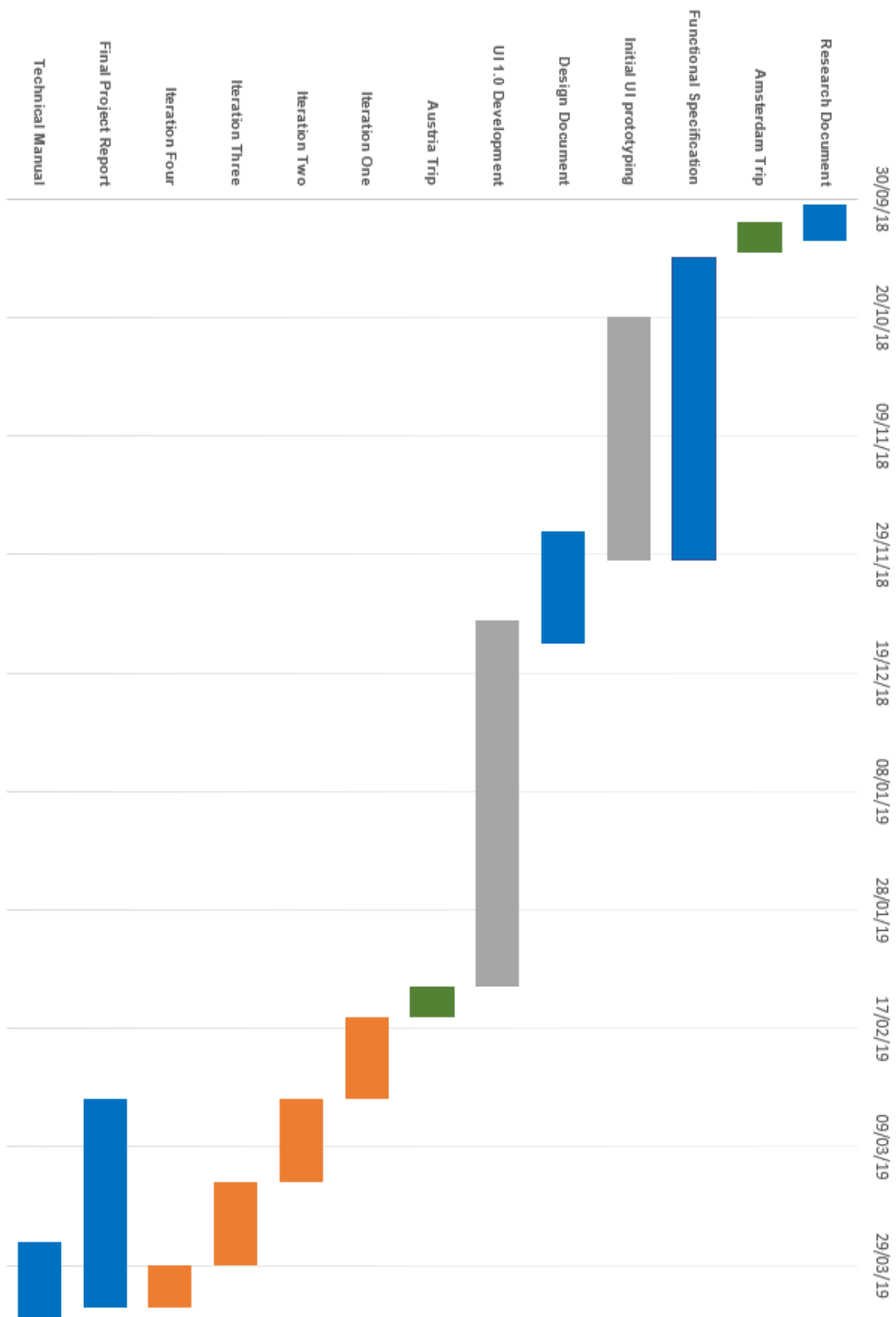


Figure 7. Project Gantt Chart

Supplementary Specification (FURPS)

Functionality

The mobile application described in this document must allow the user to track symptoms and emotions. This information must serve the primary purpose of encouraging teenage coeliac patients to comply with a gluten free diet. This encouragement should come in the form of positive reinforcement of the fact that a gluten free diet will lead to a happier, healthier life.

Usability

- The user must be able to register with the application in less than five minutes.
- Once logged in, the user should remain logged in on the application until they choose to log out.
- The user should be able to instinctively navigate the application without completing any usage tutorials.
- The user should be able to track symptom information in less than two minutes.

Reliability

If the application loses internet connection, the user should still be able to navigate the user interface. Any actions carried out by the user while the application is offline should be implemented by the system once the device regains a connection.

Performance

The application user interface should load for the user in under thirty seconds. As the application aims to allow quick input of data, this is critical to the success of the application.

Supportability

The application must be cross-platform and allow access from any iOS or Android mobile device. To allow this, the device is a web application that can be accessed from any smart device regardless of platform.