GROUP 7 PROJECT – HABIT TRACKER APP
PROJECT DOCUMENT

HOW TO RUN THE HABIT TRACKER APP

1. Clone this repository to your local machine.
2. Create a '.env' file in the backend folder which includes the following details:
   DB_HOST ="localhost" //add in your own db localhost
   DB_USER="root" //keep as is
   DB_PASSWORD="password" //your own db password
   DB_NAME="habit_tracker" //keep as is
   API_KEY= //NOT INCLUDED FOR SECURITY PURPOSES
3. Run the following commands in the terminal:
   cd Group-7-project
   npm start
4. User can now interact with the app.

INTRODUCTION

For the group project, we have created a habit tracker application (app). Our target audience is anyone who needs support in regularly completing tasks, goals or self-care. We wanted to cater to neurodivergent users in particular, and kept this in mind when designing and creating the app.

This document will explain the specifications, design, implementation and execution of the habit tracker app. This report also includes the outcomes of our user testing processes and areas where the app can be further developed in the future.

BACKGROUND

The concept is a habit tracker app that can be used to track a range of habits or goals. After creating a new account and logging in, the user is presented with a homepage which includes their profile. The user is then able to navigate through several screens which allows them to track specific habits, document their goals and review their habits.

The habits that we chose to develop as main components are:

- Hydration
- Nutrition
- Movement
- Reading
- Sleep
- Medication
- Hobbies
- Self-care
- Pets
- Plants
- Socialising
- Social Media

Please see below a walkthrough of how we would expect a user to interact with our app:

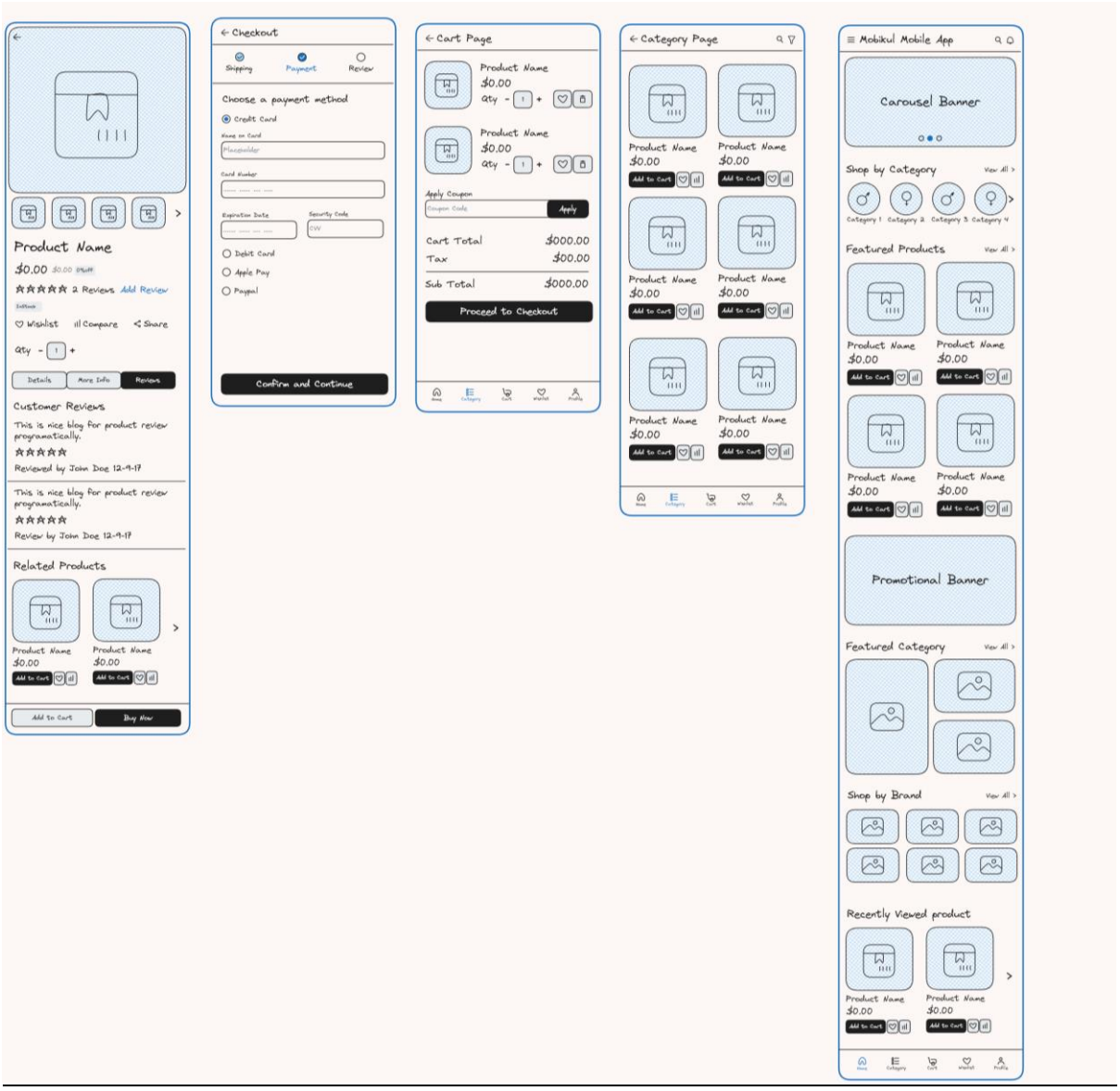
1. Login page: user can create an account, or login with an existing account.
2. Homepage: this is where the main dashboard is located.
3. Habits page: displays the habits the user wants to track and allows user to enter completion data.
4. Goal setting page: user can click on individual habits and set their goals.
5. Settings page: user can change their avatar.
6. ‘About’ page: provides the user more information about the app and its purpose.
7. Navigation bars: user can navigate through to different parts of the app using the navigation bars located at the top and bottom of each screen.

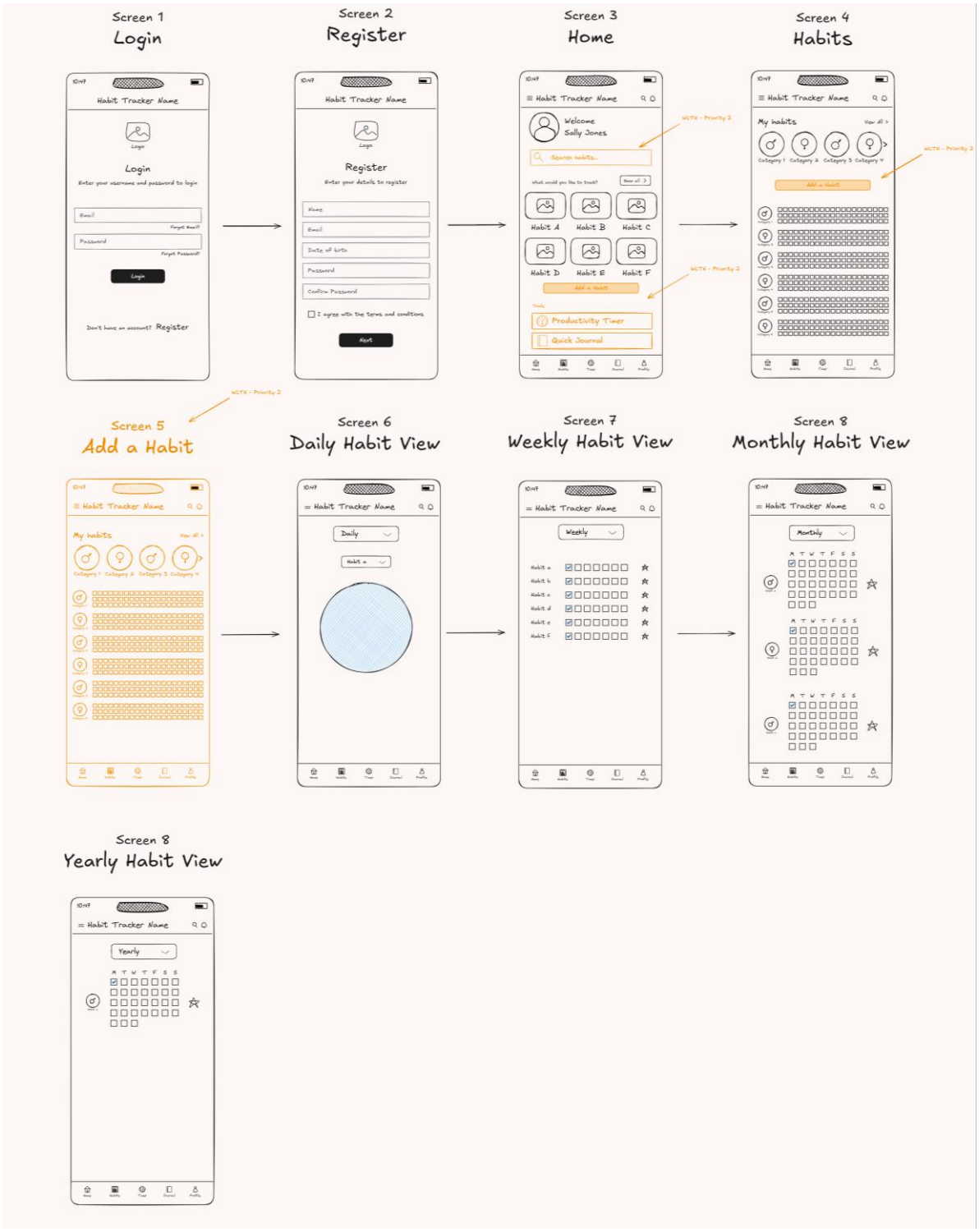
The additional features we integrated into our app are:

- Social media sharing links: user can share the app and their progress on social media platforms directly from the app.
- Word of the day: third-party API integrated to provide a ‘word of the day’ for users.

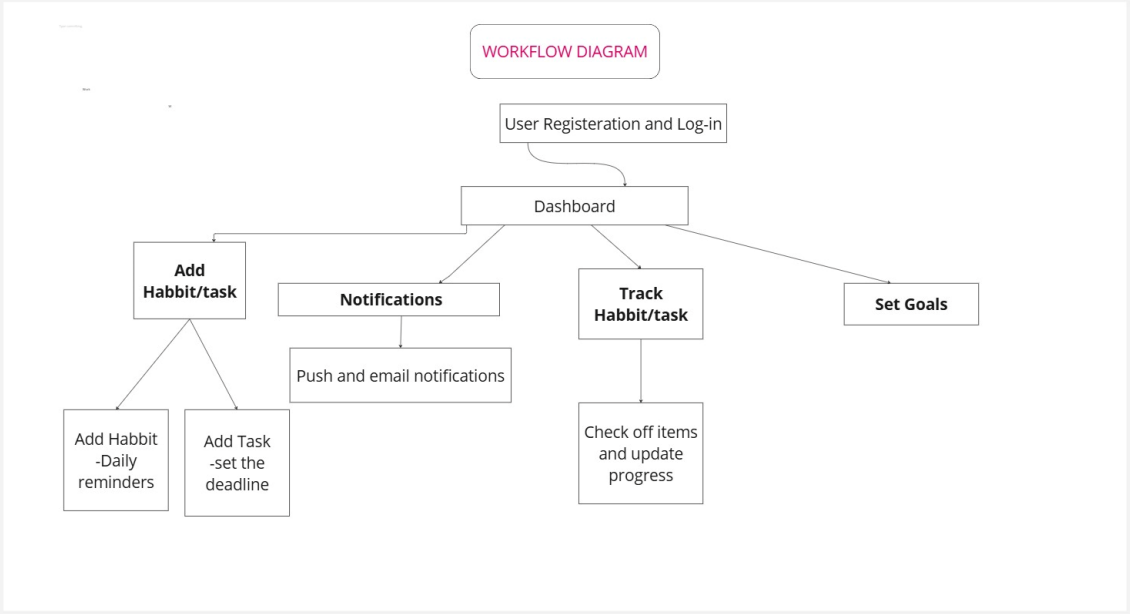
We have designed this app with neurodivergent users in mind. For example, we have simplified the way that habit goals are marked as 'completed', and avoided having a variety of options to document partial completion. We have avoided the use of multiple options/dropdown menus where possible. We have used a simple and consistent design across all screens. In relation to the habit tracking components, we used icons that would be easily interpreted, to avoid the need for additional text.

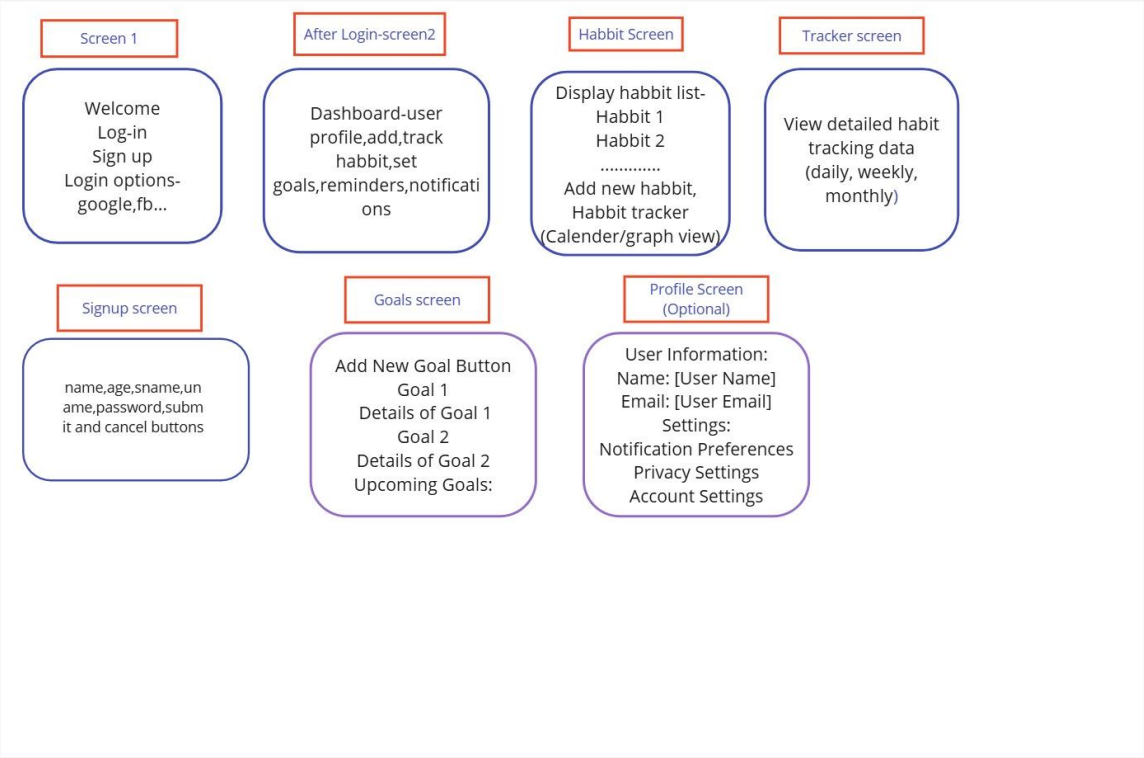
Wireframes for habit tracker app



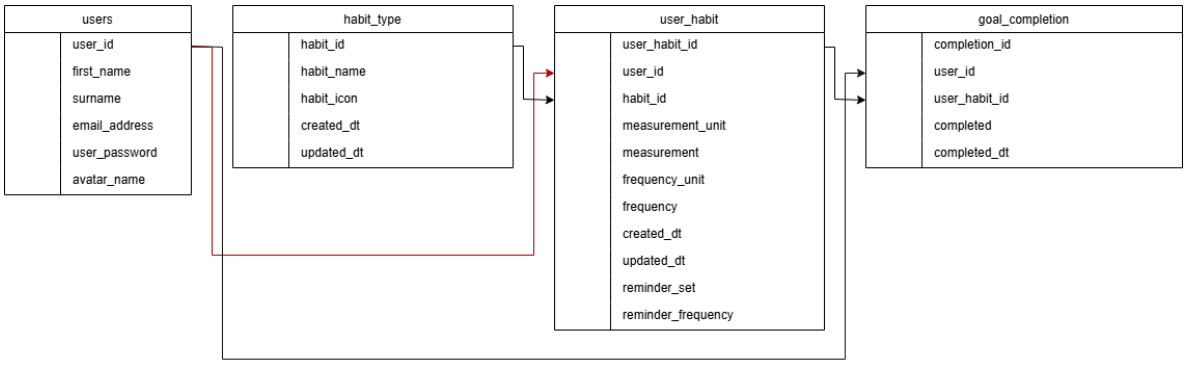

Screen 1
Login
Habit Tracker Name
Logo
Login
Enter your username and password to login
Email
Password
Login
Don't have an account? Register
Screen 2
Register
Habit Tracker Name
Logo
Register
Enter your details to register
Name
Email
Date of birth
Password
Confirm Password
I agree with the terms and conditions
Next
Screen 3
Home
Habit Tracker Name
Welcome
Sally Jones
Search habits...
What would you like to track?
View all
Habit A
Habit B
Habit C
Habit D
Habit E
Habit F
Add a Habit
Tools
Productivity Timer
Quick Journal
WLTH - Priority 2
Screen 4
Habits
Habit Tracker Name
My habits
View all
Category 1
Category 2
Category 3
Category 4
Add a Habit
WLTH - Priority 2
WLTH - Priority 2
Screen 5
Add a Habit
Habit Tracker Name
My habits
View all
Screen 6
Daily Habit View
Habit Tracker Name
Daily
Habit a
Screen 7
Weekly Habit View
Habit Tracker Name
Weekly
Habit a
Habit b
Habit c
Habit d
Habit e
Habit f
Screen 8
Monthly Habit View
Habit Tracker Name
Monthly
M T W T F S S
Screen 8
Yearly Habit View
Habit Tracker Name
Yearly
M T W T F S S

Workflow diagram and mapping





Database diagram



SPECIFICATIONS AND DESIGN

This app was built as a React app, which provides flexibility in relation to use on desktops, laptops and mobile devices. We used the following coding languages and softwares to create the app:

- React.js: used to build the app for deployment on a web browser. React.js allows for better functionality and smoother running on web and mobile browsers.
- Tailwind/CSS: for app design and branding. Since Tailwind was not included in the course, this was a new language for our team to learn and gain experience of using via the project.
- Node.js with Express and Middleware: this allows the app to interact with APIs to capture/display data and a MySQL database to store the data. Within this app, we have built two APIs - one for managing user login details and one for habit tracking.
- MySQL: for database infrastructure and management.
- Git and Github: for version control and collaboration.
- Postman and ThunderClient: to test API connectivity.
- VSCode: to write and edit the code.
- Deployment: app will be used on a web browser which allows users to use this app on desktops, laptops and mobile devices.

Tools and libraries

- Node.js packages
- React.js, react-dom and react-router-dom packages
- Nodemon
- Express.js, Middleware and express-validator packages
- jsonwebtoken
- Cors, body-parser
- MySQL2
- Axios
- Dotenv
- Bcrypt, Bcryptjs
- Jest
- concurrently
- date-fns
- react-share
- Tools for integrating fonts and colours, including Google Fonts and Adobe Contrast Checker
- FontAwesome – for easily recognisable icons which will display across various browsers/devices

Please refer to the package.json files in the Github repository for further details and versions used.

Minimum viable product

Must have:

- Login page which includes frontend and backend validation for email address and password.

- Registration page which includes frontend and backend validation for first name, surname, email address and password.
- Ability for user to log out of the app.
- The following habits must be integrated as defaults for user to choose from: hydration, nutrition, movement, reading, sleep, medication, hobbies, self-care, pets, plants, socialising and social media.
- Navigation bars (header and footer).
- Branding and design which is suitable for neurodivergent users. See Appendix 3 for the mock-ups that were created for the design of the app at the start of the project.
- SQL database which is suitably structured to store user login details securely (i.e. hashing for passwords) and habit tracking data.
- Functioning API which connects to the React app and facilitates storage of data.

Nice to have:
- Having daily, weekly, monthly and yearly views of habit tracking data, which allows the user to view the progress they have made.
- Inclusion of third-party API.
- Sharing on social media.
- 'About' page.

IMPLEMENTATION AND EXECUTION

Development approach

We used 'Sprint' methodology to ensure that project tasks were being completed in a timely manner. As per our SWOT analysis, we identified that some of our team members have availability during the week, whereas others do not (due to working hours, family commitments, etc.). Therefore, we allocated tasks in accordance with availability, individual strengths and areas of knowledge and experience. We used our SWOT analysis to identify which team members would be best placed to complete particular tasks. For example, those who have identified front-end/creativity as strengths are working on the app's design.

We used the following tools to develop this project and work as a team:
- Slack: we used our Slack channel to communicate with each other regularly.
- Notion: we used a Notion board to collaborate and facilitate team working regardless of timing/availability. Edits/changes could be tracked on the Notion board, which allowed us to see each team member's contributions. In relation to tasks for the project, we created tasks (similar to a ticketing system used in Jira) and allocated specific team members to complete these tasks.
- Google Meet: we scheduled regular group meetings to discuss progress, follow up outstanding tasks and allocate new tasks for completion. Smaller meetings were facilitated between the relevant team members who are working together on specific tasks. All meetings either recorded, or minutes documented on our Notion page.
- Excalidraw: we used these tools for wireframing our app.
- Canva: we used this tool to create a mock-up of the front-end design of the app.
- MIRO: we used this tool to work collaboratively for our team's SWOT analysis and brainstorming for project ideas.

In keeping with Agile working, we used Git and GitHub to keep an audit trail of each team member's contributions. Our repository was set up to prevent collaborators from merging branches directly into the main branch. Each pull request required at least one reviewer to check the changes before merging with the main branch. This allowed any errors and queries to be identified prior to the changes being merged into the main branch. Some team members also opted to pair program to make the process of building code more efficient and bring in a more proactive approach to troubleshooting.

We used an iterative approach to developing our components. We would start by creating a basic functioning page within the React app, then developing the design and backend functions further with

each iteration. We also used this approach when connecting different parts of our app together. For example, for the user login page, we had different team members developing the frontend and backend components. After code reviews for both parts of the component, we connected the frontend and backend components. Following peer review, testing and troubleshooting, the user login function was confirmed to be functional and merged into the main branch.

With each iteration, we aimed to implement code refactoring. With further commits, we were able to add comments to the code to ensure that external reviewers/users could understand the purpose of each function/section of code. We also introduced Redux to reduce the need for code repetition, thus promoting consistency throughout the app and improving code readability.

All tasks were prioritised based on urgency and were allocated to the relevant 'sprints' to ensure that these were completed in a timely manner. Please see below the dates of the sprints that we set up for this project.

- Sprint 1: 22/07/24 to 02/08/24 – group assignment deadline was 02/08/24.
- Sprint 2: 03/08/24 to 11/08/24.
- Sprint 3: 12/08/24 to 18/08/24.
- Sprint 4: 19/08/24 to 25/08/24 – group project submission deadline was 25/08/24.

At the start of the project, we split into a frontend team (DS, KM) and a backend team (HM, MN, SP) to ensure that both sides of the app were being developed. Towards the end of the project, we worked collaboratively on outstanding tasks to ensure that these were completed before the deadline. Due to the collaborative nature of the project, all team members were involved in reviewing the code and making contributions/amendments to various parts of the project. Please refer to the project activity logs for further clarification regarding roles, responsibilities and tasks undertaken by each team member.

Implementation

In relation to implementation, we have managed to successfully integrate all the 'must have' requirements for our project and ensure that it is functioning as expected. We have created an app that is in the most part consistent with our wireframes and front-end mock-up designs. Due to time and 'woman-power' limitations, we have not been able to add all the features from our 'nice to have' list, but we were aware that we were being ambitious from the start of the project. In relation to implementation challenges, we have had to simplify the view options and the links between the app and database to ensure that we were able to achieve the minimum viable product. Having a 'design-led' approach meant that the backend functions needed to be integrated into the existing code. Adjustments were required to the component structure, function/const names and database to avoid overlap and confusion in relation to nomenclature. A component spreadsheet was created to allow team members to keep track of nomenclature used in the code.

Execution

Only one terminal is required to start the app. When the user runs the app, the React app will automatically open in the browser and will run on localhost:3000. The API will run on localhost:3001 alongside the React app.

TESTING AND EVALUATION

Testing strategy

We used a range of testing methods to ensure that the app was functioning as expected:

- Jest: to test the code and ensure that it was functioning as expected. We have included an example of a test that we created for the login page (see login.test.js file in the frontend folder).
- We integrated error handling in the code to confirm that the app was functioning and sending/receiving data as expected. For example, with the API, we have integrated error messages which are specific to the server or the database – these can be seen in the console and console.log messages. This will allow developers to understand the nature of the error if these occur.

- We have included a Postman collection to demonstrate the various parts of the API and confirm that they are working.

Feedback from user testing:
Throughout the design process, we used our experience of using similar apps to develop something we feel we would want to use. We also showed elements to friends and family to ensure our use of design was accessible and appealed to a range of audiences. The feedback we received was that the app had a great design, and that the icons were easily recognisable in relation to which habits they represented (e.g. leaf icon indicated that this habit was looking after plants).

Strengths of the app:
As our project has been design-led, we feel we have created strong branding and that the look and feel is very professional and will appeal to users with its simplicity of design and bold use of colour. The navigation is very simple to follow, and the user experience is uncluttered and straightforward. This makes it accessible to a wide range of users including those who are neurodivergent or easily distracted. We have been able to integrate validation features into the login/registration functions which provides additional layers of security.

Weaknesses of the app:
One of the 'nice to have' requirements was to allow the user to view habit completion data and track their progress in daily, weekly, monthly and yearly view pages. This needs further development to ensure that this function is working and is available for users to benefit from. Given that the app was created with neurodivergent users in mind, the app should also include accessibility features. This could include colour contrast, large font and enabling connectivity to text to speech apps. We also recognise that apps of this nature are usually intended for use on mobile devices via apps that are native to smartphone operating systems – accessing the app via a browser would not be convenient for most users. This may lead to users choosing alternative habit tracker apps.

Opportunities for the app:
If this were to be an ongoing project, there are additional features we would love to add as further releases. This includes connecting with calendar apps and wearable devices to issue reminders, a productivity timer feature to allow users to time how long they spend on their habits, and to auto-update their records. More personalisation options could also be integrated, including adding your own habit, choosing colours, icons, profile picture, etc. We could also develop the potential for social media sharing and include a wider range of platforms. For example, we could integrate third-party API functionality for Spotify to allow users to share productivity boosting playlists.

Threats to the app:
Some of the packages used to build the app will be deprecated in the near future, which means that some of the components may no longer function as expected. The code will need to be updated with current packages to ensure that it remains functional. For the app to be publicly available, we would need to find a suitable hosting platform which supports the code we have written. To be competitive with other habit tracking apps available on the market, we would need to create a version of this app that functions natively on smartphone operating systems (iOS and Android). Furthermore, the app is not yet designed to handle a large number of users. The code and infrastructure will need to be developed further to ensure that the app can handle a higher level of traffic/usage.

CONCLUSION
Overall, we have been able to work together as a team to create a functional app which allows users to track a range of habits. However as mentioned above, there is scope to develop this app further to develop its existing functions and integrate more useful features for users. In particular, there are adaptations that can be made to ensure that the app is suitable for neurodivergent users. The experience has given us a chance to implement what we have learnt from the course, learn new skills and gain experience of working in an agile manner to build a functioning full-stack app.

Appendix 1 – Front-end mock ups