

Mind Care

Interim Report

TU856

BSc in Computer Science

**Dan Russuleac**

**C21390151**

**Pierpaolo Dondio**

School of Computer Science

Technological University, Dublin

**04/11/2024**

Abstract

My application Mindcare will be a web-based application that is designed to provide users with easily accessible tools to help them improve their mental health. This is done by providing tools for managing stress, tracking moods, creating positive mental health habits and sharing their journey/progression with a warm community. Using technologies such as React, Express.js and PostgresSQL, my application Mindcare will provide an AI assistant powered by AI that serves as a friend throughout a user’s journey using the application, allowing them to engage in meaningful conversations, receive feedback, foster growth and also not be alone in the process. Mindcare will also offer features like journalling, daily tasks generated by AI, mood tracking, sleep monitoring/logging, and a platform for sharing growth. With the multiple studies and reports done on the benefits of Chatbots on mental health and the tools provided to a user Mindcare has the potential to easily change someone’s life. That is the ultimate goal. By looking at what Mental Health applications lack currently I want to build the best tool to empower user’s in this journey and with extensive research on what can help people most I believe this is a task that my application Mindcare will be able to achieve.

Declaration

I hereby declare that the work described in this dissertation is, except where otherwise stated, entirely my own work and has not been submitted as an exercise for a degree at this or any other university.

Signed:

\_\_\_\_\_Dan Russuleac\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

04/11/2024

Acknowledgements

Table of Contents

[1. Introduction 7](#_Toc119925914)

[1.1. Project Background 7](#_Toc119925915)

[1.2. Project Description 7](#_Toc119925916)

[1.3. Project Aims and Objectives 7](#_Toc119925917)

[1.4. Project Scope 7](#_Toc119925918)

[1.5. Thesis Roadmap 7](#_Toc119925919)

[2. Literature Review 8](#_Toc119925920)

[2.1. Introduction 8](#_Toc119925921)

[2.2. Alternative Existing Solutions to Your Problem 8](#_Toc119925922)

[2.3. Technologies you’ve researched 8](#_Toc119925923)

[2.4. Other Research you’ve done 8](#_Toc119925924)

[2.5. Existing Final Year Projects 8](#_Toc119925925)

[2.6. Conclusions 8](#_Toc119925926)

[3. System Design 9](#_Toc119925927)

[3.1. Introduction 9](#_Toc119925928)

[3.2. Software Methodology 9](#_Toc119925929)

[3.3. Overview of System 9](#_Toc119925930)

[3.X. Other Sections 9](#_Toc119925931)

[3.X. Conclusions 9](#_Toc119925932)

[4. Testing and Evaluation 10](#_Toc119925933)

[4.1. Introduction 10](#_Toc119925934)

[4.2. Plan for Testing 10](#_Toc119925935)

[4.3. Plan for Evaluation 10](#_Toc119925936)

[4.4. Conclusions 10](#_Toc119925937)

[5. Prototype Development 11](#_Toc119925938)

[5.1. Introduction 11](#_Toc119925939)

[5.2. Prototype Development 11](#_Toc119925940)

[5.3. Other Sections 11](#_Toc119925941)

[5.4. Conclusions 11](#_Toc119925942)

[6. Issues and Future Work 12](#_Toc119925943)

[6.1. Introduction 12](#_Toc119925944)

[6.2. Issues and Risks 12](#_Toc119925945)

[6.3. Plans and Future Work 12](#_Toc119925946)

[6.3.1. GANTT Chart 12](#_Toc119925947)

[Bibliography 13](#_Toc119925948)

# 1. Introduction

## Project Background

Mindcare is a web application that is aimed at improving a person’s mental wellbeing by providing a multitude of tools to help users manage stress, track their emotions and build healthy mental health habits. It is quite obvious that in the last few years especially after Covid that mental health awareness has gained a significant amount of attention globally, TU-Dublin being no exception with the student union trying their best to spread awareness for example. According to recent studies, rates of mental health conditions such as anxiety and depression have been rising, with one in four people likely to experience mental health issues at some point in their life [1]. Despite the efforts of increasing awareness, many individuals still face barriers to accessing mental health support due to factors such as cost, social stigma and a limited availability of traditional therapy [2].

The development and ideation of Mindcare was inspired by the growing adoption of technology for personal support and mental health management. AI-driven chatbots such as those introduced by popular social media platforms (snapchat [3]), have attempted to provide users with a “virtual friend”. However, these tools are often very general and lack the approach needed to provide mental health support to their users. Many existing mental health apps also focus on one or two tools sometimes requiring payment or displaying ads to monetize the application which can distract a user from their outcome on the application [4].

Mindcare is designed to fill in these gaps between applications by creating a fully dedicated and accessible space for Mental Health support that incorporates multiple key features. By combining it’s features Mindcare will provide a well rounded experience that addresses various aspects of mental health [5].

## Project Description

Mindcare is a fully fledged web based application for mental health designed to provide users with the ability to manage and control their mental wellbeing. The platform will provide a safe and accessible space for users to engage in activities that support mental health including journaling, tracking moods, setting daily tasks, interacting with an AI powered assistant and a levelling system.

The project aims to address the limitations of traditional mental health applications/methods by offering multiple features in a single platform tailored to the needs of users seeking this type of application. Mindcare is built to be user-friendly, secure and scalable allowing it to offer a consistent and supportive experience for users at various stages of their mental health journey and user experience.

The core functions of Mindcare will be an AI-powered assistant to provide conversations with users, journalling to allow users to write in privately what they are feeling and doing and also to store daily reflections, mood tracking with graphs in order to make it easier for users to see trends in their mental health and how they are feeling, daily tasks to encourage a positive and health routine to users to strength the results of their journey, sleep tracking to recognise patterns in sleep that could lead to understanding of certain emotions, positive moments sharing section with the community of Mindcare, lastly a SOS button in case of emergency’s users can click this button to help them out.

To aid in the seamless functionality and data security of this application the system architecture of Mindcare will be divided into 5 main components. The user interface(Front-end), Backend Server, External Services(API’s) and a cloud hosting environment. The frontend created with react will provide a responsive interface for user’s to work with, research on what type of styling to provide a calming application will be vital to provide user’s with a well rounded application. The backend constructed in Express.js will manage all processing, user requests and secure authentication throughout the application. User data will then be stored in a PostSQL database and details such as a user’s password will be stored securely with encryption. External Services such as an AI api for the chatbot will enhance the application and provide user’s with a powerful assistant to work with. Lastly a cloud hosting provider will allow me to demonstrate my project in action.

The image below demonstrates a Mindcare’s system architecture and how these components interact with one another highlighting the flow between frontend, backend, database and external services. With a strong and robust structure, Mindcare will be able to offer a seamless, secure and effective Mental Health solution to user’s in need.



Figure :Basic System Overview

## Project Aims and Objectives

The overall aim of Mindcare is to create a well functioning and user-friendly web application that allows users to manage their mental well-being. This goal is achieved by integrating technologies and mental health support tools within a secure, professional, accessible platform therefore providing a digital environment for mental well-being.

* Objectives:
* Review and Research Relevant Literature: Conducting an in-depth review of exiting mental health applications and AI based chatbots systems to identify fault/weaknesses to provide Mindcare with opportunities to grow.
* Designing of a well structured system architecture: Developing a clear and thorough system design that outlines the front-end, back-end, database, external services and ensuring scalability, easy of use and versatility.
* Developing and integrating core features: Implementing Mindcare’s core features including the AI assistant, secure login/register, authentication, journalling, mood tracking, daily tasks, sos mode.
* Security: Due to the personal information that will be contained in Mindcare, efforts to secure and authenticate user’s will be vital.
* Test and evaluation of system performance: Conduct testing on the system to see how well and effective it performs, long tokened messages from AI could take awhile to be returned, performance will need to be optimised to keep user’s attention.
* Gathering and Reflection on User Feedback: Collecting user feedback and changing the application based on a reflection of user feedback will be vital for Mindcare to succeed.

## Project Scope

The scope of Mindcare is to develop a web based mental health application that supports users in managing stress, tracking moods, journalling, forming a healthy routine and building positive mental health habits. Mindcare’s focus is on providing accessible AI assisted tools that allow users to engange in mental wellness activites in a secure environment. Key features such as the AI powered assistant, daily tasks, sleep tracking, SOS mode all aim to offer users a variety of resources for mental health support.

Mindcare is not designed to replace professional mental health services or replace therapy with licensed practitioners. It is a platform for user’s without accessibility to licensed therapists to have at least something to benefit them. It also allows users to gain positive habits and help in time of need when there are no other options around. It is easily accessible so it serves as an application that anyone can use in any situation at anytime for any possible reason. That is the goal of Mindcare, to take care of someone’s mind whenever possible. Mindcare is not capable of engaging in remote monitoring or real-time intervention. It is not able to diagnose people with any sort of conditions and only mention symptoms and possible reasons.

By defining these boundaries about Mindcare’s scope we can ensure a focused and clear approach to it’s primary goal; offering accessible digital tools that assist users in managing and reflecting on their mental health in a private and secure setting.

## Thesis Roadmap

This thesis provides a comprehensive overview of the research, design development, testing, prototype development and issues and future work.

* Chapter 2: Literature Review; this chapter reviews relevant literature and research to establish the background of Mindcare, including a full analysis of existing solutions, technologies and academic articles/studies about research in Mental Health applications and AI in mental health settings.
* Chapter 3: System Design; this chapter will provide all details about the design of the Mindcare application, covering the chosen software technology, system architecture and design choices that influenced the project’s development.
* Chapter 4: Testing and Evaluation; this chapter presents the testing strategies and evaluation criteria used to ensure Mindcare’s functionality, usability and security of the application.
* Chapter 5: Prototype Development; this chapter documents the development process for the Mindcare prototype, including code samples and explanations of implemented features.
* Chapter 6: Issues and future work; this chapter will discuss the challenges faced throughout development and mention potential future enhancements and areas for improvement.

# Literature Review

## 2.1. Introduction

In this chapter, numerous areas of research that contributed to the development of Mindcare are explored along with the software solutions and technologies researched and used for the creation of Mindcare. The chapter will begin by examining alternative existing solutions that provided very valuable insights to addressing how Mindcare will function and what features will be implemented. It then covers the key technologies researched and applied in Mindcare’s development. Following this the chapter will then delve into domain-specific research focusing on mental health practices and research on digital solutions for wellness/mental health. Lastly it reviews past final-year projects with similar goals, offering insights that informed the design and functionality of Mindcare.

## 2.2. Alternative Existing Solutions to Your Problem

In the development of Mindcare it is essential to look at existing solutions and examine them to understand their features, strengths, weaknesses and limitations when compared to Mindcare in order to foster more ideas. This analysis provides insights into current trends, user needs and design/styling of Mindcare. After looking at a multitude of applications the three that suit Mindcare goals most are Elomia [6], Earkick [7] and Youper [8].

Elomia is an AI-powered mental health chatbot that offers 24/7 support for multiple issues ranging from daily stress to more serious conditions. It provides a realistic conversational experience aiming to make users feel as though they are speaking with a human therapist. Elomia’s focus is mainly on the chatbot experience, without any other tools like journaling or daily tasks which Mindcare will include.

Earkick is another AI-driven mental health application that measures and improves mental health in real-time. It offers personalized insights and daily content tailored to the user’s needs. Earkick emphasizes real-time tracking and personalized but lacks features such as journaling and mood tracking with graphing. It promises that the chatbot remembers your previous conversations but from testing the chatbot is not setup with remembering previous inputs from the user.

Youper combines an AI chatbot with a few self-help tools, including mood tracking, journaling and goal setting. Youper’s interactive and flexible approach allows user to explore multiple methods of support making it a versatile tool which is more inline with what a mental health application should include. However many of it’s advanced features are behind a paywall therefore limiting accessibility for some user’s which is not ideal.

These existing solutions seriously highlight the need for an accessible user friendly platform that combines multiple supportive tools without limiting any access to features with subscriptions or paywalls. Mindcare aims to bridge these gaps between all these applications by implementing all their features and more and allowing user’s to freely use the application. This allows Mindcare to be ahead of these applications if designed correctly as it’s functionality will be much further ahead.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Feature | Mindcare | Elomia | Earkick | Youper |
| AI Chatbot | Yes – Powerful AI(conversational/  guiding) | Yes – realistic AI | Yes – personalised but less conversational depth | Yes – powerful conversational |
| 24/7 Support | Yes – always accessible | Yes – always accessible | Yes – real-time tracking | Yes – always accessible |
| Mood Tracking | Yes – tracking mood with graphs showing visual trends | No | Yes – real time tracking of emotions and mental health | Yes – mood tracking |
| Journalling | Yes – private space for users to log their thoughts and reflect | No | No | Yes – journaling option available |
| AI Daily Tasks | Yes – your own created/ AI generated available | No | No | Limited – integrates goal setting but not custom daily tasks |
| Sleep Tracking | Yes – user provides sleep time and tracks sleep patterns | No | No | No |
| Positive Moments Sharing | Yes – allows users to share moments with others in the Mindcare community | No | No | No |
| SOS Mode | Yes – support with emergency numbers provided incase | No | No | No |
| Personalized Functions | Yes – AI recommends actions based on user behaviour in chats and preferences/ daily tasks | No | Yes – real time insights and feedback | Yes – personalised recommendations for self-care |
| Cost | Free | Free(limited features) | Free(limited features) | Free with subscription for advanced features) |
| Privacy | High – no collection of personal data except for what user inputs – securely stored in DB | High – anonymous chatbot | High – focuses on anonymous real time tracking | Medium – collects personal data especially for advanced features locked behind the paywall |
| Target Audience | Everyone who can access the internet and may need any components of the application. | Individuals seeking conversational AI support | People interested in real-time mental health insights based on their conversation | Individuals looking for a chatbot with a few other features but not many. |

**Table 1: Comparison of existing solutions to Mindcare**

## 2.3. Technologies you’ve researched

Before the development of Mindcare multiple technologies and languages were researched in order to try chose the best for the development of Mindcare. Each choice for the technology was carefully done to ensure for scalability, manageable workloads, efficiency and overall look/feel of the application.

### 2.3.1 Front-End

React: React was chosen as the front-end for building user interfaces due to the way you can build and reuse multiple components throughout the application. This allows for an application to have a modular and reuseable design. React’s virtual DOM improves performance in dynamic applications making it suitable for Mindcare’s interactive features such as the chatbot that will be implemented and other components.

Alternative – Angular: Angular is another framework somewhat similar to react. It allows for two way data binding and built in services but it has a steep learning curve that I have just started for a module. It also has a larger structure to react and can slow down development. React’s flexibility and extensive support and documentation made it a more practical choice for this project.

### 2.3.2 Backend-End

Express.js with Node.js: Express.js was chosen as the backend framework for Mindcare providing a simple, flexible and effective approach to API routing, middleware management and HTTP request handling. This allows for real-time features being implemented with ease such as mood’s being inputted and then a graph updating based on user changes. Combined with Node.js, Express offer a very efficient and scalable platform for developing a powerful and lightweight backend.

Alternative – Django with python: Django is a robust full stack framework with built-in features like an ORM and admin panel. While Django is very structured it’ synchronous nature and design are less suited for real-time applications like Mindcare. The asynchronous capabilities of Express made it a better choice.

### 2.3.3 Database Management

PostgreSQL: PostgreSQL was selected for its support of ACID compliance, complex queries and structured data types. It’s SQL based relational database structure is highly secure and well suited to manage the variety of data that Mindcare will be working with(Journal Entries, User Info, Chat logs).

Alternative – MySQL: MySQL is widely used and easy to setup but it handles complex queries less efficiently than PostgresSQL. Given the data variety and complexity expected to be in Mindcare, PostgreSQL advanced features such as efficient query completion and enhanced indexing would be more beneficial.

Alternative – MongoDB: MongoDB is a noSQL database and offers flexibility in handling unstructured data but lacks the relational integrity that is needed for managing structured data and sensitive user information. PostgreSQL’s relational structure and support for data security were a lot more beneficial for Mindcare’s development.

### 2.3.4 API and AI Implementation

API for AI: To implement Mindcare’s conversational AI chatbot an external API from AI/ML API was selected rather than building a custom model from scratch or hosting a free to use model on a python flask server. This API approach offers robust natural language processing thereby reducing development time and possible issues. The use of Llama 3.1 as the conversational bot means any sort of query or concern raised by the user to the Model will easily be responded to with answer that are very clear and concise.

Alternative – Self hosted AI model: Building my own model is something I had researched about and also attempted however the resources required to train a model with even a low amount of conversations/data is very intensive and time consuming. Also hosting a somewhat large language model on a python flask server and connecting it to my backend is a possibility however the performance is questionable as a response can take upwards of 5 minutes to be generated and parsed to the front-end to the user which would just not work on my application. This is why utilising an API was chosen.

### 2.3.5 Server-side scripting and Middleware

Express.js with JWT authentication: Express.js allows for the integration of JSON web tokens (JWT) for authentication ensuring that sensitive data such as chat and journal entires are securely accessed and protected. This lightweight approach to middleware allows for simple yet effective session management without compromising security or performance degradation which is crucial for Mindcare due to the sensitive information that will be held.

### 2.3.6 Programming Language

Javascript: is the main programming language and is used to implement the core logic in both the front-end and back-end of Mindcare. Javascript is what powers react, enabling the creation of interactive UI components while also being able to support backend logic with Express.js for routing, authentication and data handling. The language is perfect for real time logic such as chatbots and journal entries, etc.

SQL serves as the only language for database management in Mindcare specifically in PostgreSQL. SQL’s ability to handle complex queries allows for secure and efficient data handling especially for storing sensitive information about users. It’s relational structure makes it perfect for applications where data consistency and security are vital.

HTML (Hypertext Markup Language): provides the structure of Mindcare’s web interface. All elements in the app are ultimately represented as HTML due to react’s rendering of JSX components to HTML. HTML allows content to be organized hierarchically and defines the skeleton of each page and component.

CSS (Cascading Style Sheets): controls the visual presentation of the application ensuring consistency and enhancing user experience. For Mindcare, CSS allows for layouts to be adjust, colour themes, animations and helping make the interface engaging and accessible. CSS rules are defined either globally or at component level.

Material UI sx prop: is used in combination with CSS to provide inline styling in react components. The sx prop from material UI is a powerful styling tool that allows inline CSS customization directly within components allowing for rapid context-specific styling adjustments without the need for external style sheets. Using Material UI sx can allow for enhanced styling and for less pages to be used as external style sheets are rendered useless.

## 2.4. Other Research you’ve done

AI Chatbots in Mental Health:

AI Chatbots are now viewed as accessible tools for mental health support by being utilized as an assistant or friend for example. A scoping review measured their effectiveness and feasibility by highlighting benefits in improving mental and emotional well-being. However, challenges related to usability, user engagement and integration with existing healthcare systems were identified [9].

AI-powered Therapy vs Traditional Methods:

There is a lot of research when it comes to comparing AI-driven mental health tools with traditional methods such as reading books or speaking to a therapist for example. A study explored the effects of AI based therapy revealing that although AI chatbots may lack the depth of traditional therapy methods they provide a convenient and accessible form of support that many users find extremely helpful for initial engagement in mental health care [10]

Challenges in Mental Health Applications:

Research also has identified that engagement with mental health apps can significantly decrease if the AI models are underdeveloped or if the application is not well built and intriguing to users. Making an application difficult to navigate or using a model that is not capable of providing humane responses will lead to user’s becoming disengaged with the application. This showcases the importance of making sure an application is simple yet effective in what it does and provides the user with full control [11]

Mood Tracking and Emotional Awareness:

Mood tracking technologies offer a structured way to reconnect with our inner selves and see how we are truly feeling, therefore providing greater self-awareness. By understanding our emotional patterns and seeing them visualised we can empower ourselves to make informed decisions about our mental health and what we want to work on which has been reported by multiple sources [12]

## 2.5. Existing Final Year Projects

An existing final year project is “Productive Gym – Fitness Web Application” by Klaudiusz Nowakowski.

The application focuses on using technology to enhance personal well-being through a user friendly web application just like Mindcare. Both projects put an emphasis on accessibility and engagement from it’s users, with the gym application providing features like calorie tracking, workout logging and a blogging platform. Both applications leverage components and features to keep users engaged. Mindcare just puts an emphasis on mental health components instead of physically benefiting components like the gym application.

Both Projects use React for the front-end, showcasing that React is great for building a dynamic and interactive user interface that enhances the user experience. React’s component based architecture and virtual DOM provides scalability and responsiveness that is needed for delivering a seamless interaction with features like workout logs in the Productive gym application. To add on, both projects emphasise a modular design which ensures maintainability and efficiency in their respective systems.

On the backend, Productive Gym employs Node.js and MySQL where Mindcare uses Express.js and PostgreSQL. Both applications choose robust technologies suited for scalable and secure data management alongside simple yet effective routing to external services. The technology choices reflect a shared desire for creating a reliable application capable of handling user specific data such as workout logs or sensitive mental health information from users in Mindcare.

Overall, both projects demonstrate the potential of using modern web technologies to create applications that improve a person's well-being. While the “Productive Gym” application is focused on physical fitness, Mindcare focuses on the mental health of it’s users. The “Productive Gym” application showcases that the technologies chosen for Mindcare’s development will be beneficial in reaching the goal Mindcare has been set out to reach.

## 2.6. Conclusion

Concluding from this chapter, it is evident that the role of AI-powered tools, Mental Health derived tools and overall a welcoming and safe web-based environment is able to be a route for people to help themselves mentally in multiple scenarios. An AI chatbot vs other means of therapy has been shown to be an effective form of therapy and a gateway to therapy and development of mental health progression. Studies have shown that accessibility and easy of use including the complexity of components adds to the realism of the application’s task and also keep’s users engaged throughout the use of the application. Overall, it is clear that the potential Mindcare offer’s to it’s users and the benefits that an user can achieve from utilising Mindcare to it’s full potential is there and ready to be achieved.

# 3. System Design

## 3.1. Introduction

The system design of Mindcare will showcase a structured approach, by integrating Front-end, Back-end and Database components to create a mental health platform. This chapter details the design decisions, chosen methodologies and implementation techniques and patterns to ensure a seamless integration between all layers and components of Mindcare.

## 3.2. Software Methodology

The Mindcare project is implemented using the Agile methodology therefore allowing iterative development and constant testing. This approach is very beneficial for a user-centered application just like Mindcare as it enables flexibility in responding to user feedback and testing completed. This allows for the overall improvement of vital components such as the AI, journal, UI and daily tasks.

## 3.3. Overview of System

An Overall overview of the system is as follows:

Frontend: Built with React and organized into different components and pages, such as Chatbot.js, MoodTracker.js and Journal Page. Each component will utilize React’s features such as Icons and sx prop styling, hence the reason for using react. The Frontend will be structured in accordance with best coding practice, with pages being in their own folder, reusable components stored together and then app.js being the heart of the application and routing. Use of imports such as Material UI will allow for dynamic and well styled pages.

Backend: Powered by Express.js, with a series of RESTful endpoints setup to support different functionalities. API routing to my own database can be seen throughout the entire application as all data inputted by the user is saved for processing and personalisation for future use of Mindcare. An external API is also setup when it comes to the Chatbot, this allows for time to be spent on Mindcare itself, instead of a language model taking up a lot of development time.

Database: PostgreSQL is used to manage user data, journal entires and chat conversations, daily tasks and all other components requiring persistence. Using PostgreSQL and creating my own database schema, I am allowing for scalability alongside data integrity as I develop a robust structure relational database for Mindcare. Queries will be handled with a pool connection to allow for asynchronous data access.

A diagram of a software development

Description automatically generated

Figure : Detailed System Architecture

## 3.4. Other Sections

User Authentication and Authorization: JWT tokens will be used for authentication and authorization with routes such as Chatbot and Journal being allowed to users with only access. This will work by letting a user registering and then creating a Token for them once they login using the JWT import for react. This token will have a key created for Mindcare and once users try access certain pages the protected route component will ensure a Token matching the key is seen in local storage before.

Error Handling: Implementing error handling will make the suer experience a lot smoother. Guiding the user to what they did wrong and not allowing for bugs means the user will be in full control of the application at all times.

UI: The frontend UI is designed to provide an interactive and calming environment that will keep users engaged throughout the use of the application. With pages such as the welcome page and login page having smooth animations to make users feel relaxed.

## 3.5. Conclusions

The Mindcare system design successfully integrates frontend and backend components with a focus on security, engagement, performance/responsiveness and usability. The chosen techonoligies and architectural patterns will allow Mindcare to be a scalable and maintainable application and will allow for development of the application to be iterated and changed to whatever means necessary. Facilitating user’s is Mindcare ultimate goal and these technologies allow for that.

# 4. Testing and Evaluation

## 4.1. Introduction

This chapter aims to address the testing and evaluation approaches that will be used throughout the development of Mindcare and also during the end of Mindcare’s development lifecycle in order to further refactor and improve Mindcare before it’s completion. Testing each component and gaining user reviews will ultimately lead to an application fit for it’s purpose.

## 4.2. Plan for Testing

Testing will include unit testing for fundamental components. This will be done to ensure that each element such as chatbot interactions, journal entires, AI generated daily taks all function as intended. Unit testing will be applied on these functions to isolate them and make sure under no circumstances will they fail, providing the user with functional components at all times. Integration testing will then follow to confirm that these functional components when integrated into the application will not not cause bugs or problems and that all components will work together seamlessly. Security testing such as SQL injection will then be used to make sure the database can not be breached and the unauthorised users can not access the private information about other users.

## 4.3. Plan for Evaluation

Evaluation for Mindcare will be focused on user feedback through user reviews and a survey to gather insights on Mindcare’s usability and effectiveness. This feedback will showcase areas of interest that either are good or may need refactoring and improving. Ease of use and engagement will be a big determinant of how great Mindcare can achieve it’s goal. So if users report back disengagement or confusion while using the application then refactoring of design or components will be vital to making sure Mindcare is a well rounded application.

A screenshot of a screenshot of a computer

Description automatically generated

Figure : User Feedback Forms

A screenshot of a computer

Description automatically generated

Figure : Example question's in Form

## 4.4. Conclusion

The testing and evaluation phases will provide insights into the strengths and weaknesses of Mindcare. By utilising unit testing and integration testing, I can ensure that Mindcare will fundamentally work, however by utilising user reviews and surveys, I can then ensure that the core functionality implemented is of use and effective for it’s purposes.

# 5. Prototype Development

## 5.1. Introduction

The prototype development phase will focus on building the core features of Mindcare, ensuring a functional and user-friendly interface for mental health support. This section will outline the progression from initial setup to the development of components, including Chatbot, journalling, mood tracker, authentication and overall development off the application’s flow(frontend, backend, database). The aim of this section is to showcase Mindcare’s scalable and modular design in action.

## 5.2. Prototype Development

At first I began by creating a simple front-end/back-end framework. This was to make sure my react project was created successfully and to get a basis off what to work on. My first challenge was to get my AI chatbot working as it was the only external piece of software I would need in my project. After attempting to work with my own chatbot hosted locally on a flask server I decided that an API would ultimately be the better option for my project. I use an API for AI/ML API’s.

A white background with black lines

Description automatically generated

Figure : API Services

Using an external service allows for my chatbot to respond a lot faster than being hosted locally so performance is gained, it also allows for better and more advanced conversational models to be used and configured to my liking.

A screen shot of a computer screen

Description automatically generated

Figure : Calling off API

This is where I call upon my API and configure it to my liking. My plan is to grab user information in the future and personalise the chatbot instead of the generic “AI assistant who knows everything”. This will allow for more complexity and personalisation in my application for users.

A computer screen with text

Description automatically generated

Figure : Conversations Loggin

I also setup up conversational logging during my development to see any errors and check for differences when I add in personalisation.

A computer screen with text and images

Description automatically generated

Figure : Error Checking/Validation

In this section I also make sure a message from the user does indeed exist when connecting to the chatbot, I also use verifytoken which calls upon my middleware to make sure that the user requests are coming from authorised users which I will explain later on.

A computer screen with text on it

Description automatically generated

Figure : External/Internal Error Checking

I also have error handling in case the external server goes down or if a user loses internet connection and can no longer send messages to the API. These are shown to the user on the frontend. I have added further complexity to the chatbot be allowing users to create multiple conversations and save messages inside each conversation for persistence. Ultimately my API only allows me to send and receive messages, everything else from parsing and authentication has to be done by myself including showcasing these messages to the user on the front end.

This was one of my biggest goals in this project, to allow users to interact with a bot that seems humane. I will now go through my frontend/backend to explain what else I have done so far.

Front-End:

Welcome.js:



Figure : Import for Vanta

A screen shot of a computer program

Description automatically generated

Figure : Vanta Configuration

In my welcome page I import a package named Vanta that is built for react and allows you to choose from 8 live wallpapers and configure them. I went with live clouds to provide users with a calming experience, I also have an animated welcome to Mindcare section to go with the calming experience of my welcome page as seen below:

A screen shot of a computer program

Description automatically generated

Figure : Styling for title

A cloud in the sky

Description automatically generated

Figure : Welcome Page

Authpage.js:

This is where the user login’s and registers their account to the application. I have a page and then three components, the loginform, registerform and particlewaveaffect.

LoginForm.js:

A screen shot of a computer program

Description automatically generated

Figure : State control in LoginForm

I use two states in my loginform, credentials and errors and manage them when a user is logging in. This requires careful synchronization and adds complexity.

A screen shot of a computer code

Description automatically generated

Figure : Asynchronous calls

I do an asynchronous API call to my backend API to check for authentication. This code handles both successful responses and saves the JWT token in local storage and also provides the user with errors(invalid sign in).

RegisterForm.js:

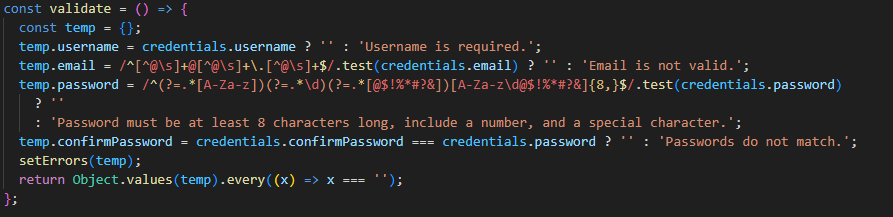


Figure : Validation Logic for Registering

Validation logic in my register form, it handles all types of validation from username being required to email’s being in correct format along with passwords being to a high enough standard. Also let’s users know exactly what is needed while registering.

A screen shot of a computer code

Description automatically generated

Figure : Dynamic Input Handling

Dynamic input handling allows for handling multiple form fields dynamically using computed property names (name attribute).

A computer screen with text on it

Description automatically generated

Figure : Asynchronous Calls

Just like in loginform I use asynchronous API calls to my backend for validation. The code differentiates between types of errors returned by the backend such as username’s existing and duplicate emails while also managing both successful registers to provide seamless user feedback.

ParticleWaveEffect.js:  
  
This is a piece of code I created in second year and that I changed and developed a bit further for my project as I thought it would add some complexity and calming factors to my application.

A computer screen shot of a code

Description automatically generated

Figure : Animation of ParticleWaveAffect

I used Three.js and just setup particles in a 3d space. These particles float and move around in a wave like motion to try calm users. I use sinusoidal waves to make the particles look like they’re moving the way they do using math.sin.

AuthPage.js:

This is where I integrate the three components together to provide the user with a professional and calming login/sign up experience.

A computer screen with text and numbers

Description automatically generated

Figure : State Handling

This is where I handle states to see if a user is trying to login or register. It decides whether the loginform or registerform should be shown.

A computer screen shot of text

Description automatically generated

Figure : Integration of Components

Integration of both components.

A computer screen with colorful text

Description automatically generated

Figure : Positioning of Components

Left side of screen with ParticlewaveEffect and the centered text on it.

With correct styling and format of components using states we end up with this result:

A screenshot of a computer screen

Description automatically generated

Figure : Login Page

A screenshot of a computer screen

Description automatically generated

Figure : Register Page

Homepage.js:

This is the heart of my application, although it is nowhere near complete I have implemented cards I will use for the final version and a carousel I plan to keep. I do want to add more to it.

A computer screen shot of a program code

Description automatically generated

Figure : Dynamic Feature Management in Homepage

Dynamic Feature’s Management, when a user clicks on a modal it then renders content based on the modal clicked. This requires a level of managing interactivity and requires proper state tracking of the application adding complexity to the front-end.

The carousel is implemented using an import:



Figure : Carousel Import

A screen shot of a computer program

Description automatically generated

Figure : Styling of Carousel

The carousel like everything in my project will change dynamically based on screen sizes, not just that it also dynamically renders content from the features array to showcase all the features of my application, it renders in images, text and routes to the feature that the user can click.

My homepage is more styling than complex functionality as it essentially routes the user to every aspect of the application. I plan to add complex functionality by showcasing previews of the journal and mood tracker in the homepage in the future.

A screenshot of a computer

Description automatically generated

Figure : Homepage

Chat.js:

A screen shot of a computer program

Description automatically generated

Figure : Integrating Components

Initiating both components for my chatpage, my chat.css aligns them properly.

A screen shot of a computer program

Description automatically generated

Figure : Components Positioning

In the chat.css I set the app container as a flexbox to allow for alignment and styling as appropriate. I also style the navbar to fix it’s position at the very top of the page.

Chatbot.js component:

This is one of my most complex front-end components currently as it is nearly completed. It manages these states:

Message Handling: message, conversation, conversations

UI States: isLoading, typing, anchorEl, isDialogOpen

Voice Control: voices, selectedVoice, autoReadEnabled, messageToRead

Error Handling: error

Dialog Management: isDeleteDialogOpen, conversationToDelete

A screen shot of a computer program

Description automatically generated

Figure : State Management

A screen shot of a computer program

Description automatically generated

Figure : Chat Pre-prompts

My chatbot contains preprompts user’s can select when speaking to the AI assistant.

A computer screen shot of a program code

Description automatically generated

Figure : Fetch User Data Hook

This is my useEffect hook to fetch user data, it allows me to set what conversations and messages the user has and will be useful for personalisation of the AI assistant for each user.

A screen shot of a computer program

Description automatically generated

Figure : Text to Speech Hook

This is my useEffect hook where I use a text to speech API call(SpeechSynthesis) to allow a user to chose if they want the returned AI messages to be read out aloud. Adds complexity and user engagement.

A screen shot of a computer program

Description automatically generated

Figure : Fetching Conversations Hook

This is my useEffect hook to fetch user conversations and then display them to the user.

A screen shot of a computer program

Description automatically generated

Figure : Fetching Messages Hook

This is my useEffect hook that fetches messages based on the conversation ID the user in on.

I have more useEffect hooks to:

Scrolltobottom of the chat when a new message from the chatbot is received.

Send the message inputted by the user to the backend to the ML API.

I then have multiple handles for all types of scenarios a user can go through on the chatbot page from adding in conversations to deleting them, selecting conversations, handling voice selection.

There is also a lot of styling when it comes to the UI from the chatbot and user having their own profile photos to typing animations being shown while a user waits for the AI assistants message to be returned. I have tried my best to make the page as engaging as possible for a user and this is the result:

A screenshot of a computer

Description automatically generated

Figure : Chatbot Page

JournalPage.js:

A black background with white text

Description automatically generated

Figure : Imports for Journal

In my journal page, I use dayjs and isbetween to find journal entries uploaded between a week interval as users will set a goal for how many journal entries they want to do per week. Every time a user enters a journal entry confetti will be shown on the screen to keep them engaged and motivated. It also has rotating motivational quotes to keep users going.

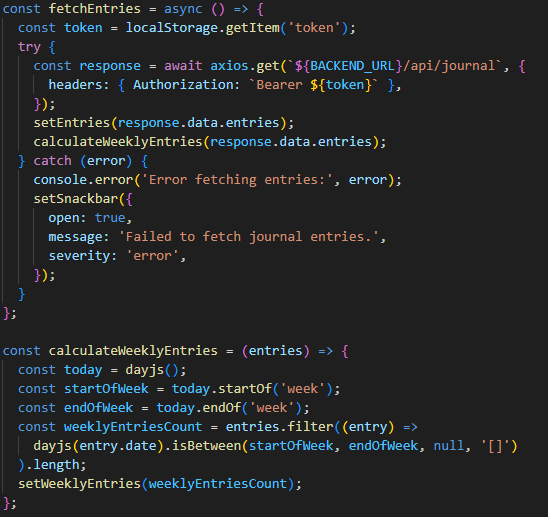


Figure : Fetching and calculation of Journals

The two functions shown fetches journal entries that the user has and also calculates how many entries a user has done in a week. As of right now the goal is hardcoded to 20 but I will allow users to chose their goal to allow for more personalisation.

A screen shot of a computer screen

Description automatically generated

Figure : CRUD operations for Journal

Entries can be submitted, edited and deleted alongside also being previewed before submission. I have handlers for all these situation and above is an example of the submit handler.

A black screen with blue text

Description automatically generated

I also have a progress bar for the weekly journal entry goal based on weekly entries divided by the weekly goal.

The rest of my code is responsive styling to card hovers, clicks and the overall look of the journal entries and styling of the page. It is all done with sx prop styling just like my other pages.

A screenshot of a computer

Description automatically generated

Figure : Journal Page

MoodTracker.js:

In my moodtracker I get users to input the mood they are feeling and then display it to them with an interactive graph. Persistance has been setup and all moods are saved for each user. As of right now there is not editing or deletion of moods.

A screen shot of a computer

Description automatically generated

Figure : Imports for Moodtracker

Imports needed from the recharts package to create my chart.

A computer screen with text

Description automatically generated

Figure : Data Preparation

The data preparation for my chart.

A screenshot of a computer program

Description automatically generated

Figure : Mood Mapping

Mapping moods that the user see’s to numbers to help with chart display.

A computer screen shot of a program

Description automatically generated

Figure : Rendering of Chart

The rendering of the chart. ResponsiveContainer ensures that the chart adjusts to different screen sizes.

A computer screen with text

Description automatically generated

Figure : Modal for adding new moods

I also have created a modal that allows users to add in their moods and submit it to the database. Once the mood is added it is fetched from the database and added to the chart. All this happens in realtime for a seamless user experience.

Just like my previous components all API calls are done through authentication first and then the API call being made.

A screen shot of a computer code

Description automatically generated

Figure : "Did you know" animation

Keyframes and slider for an animated “did you know” section where text is brought in and out at 10 second intervals.

With these components added I have a fully functional Moodtracker:

A screenshot of a computer

Description automatically generated

Figure : MoodTracker

Backend:

Server.js has already been covered.

My backend runs fairly the same throughout my components. The routing and setup is the same however the queries to my database are different:

Conversations.js:  
  
A screenshot of a computer screen

Description automatically generated

Figure : GET and POST for conversations

GET and POST requests to my database to send and receive conversations based on a user.

Journal.js:  
  
A screen shot of a computer program

Description automatically generated

Figure : GET and POST for journals

GET and POST request for my journal entries to my database.

Mood.js:  
  
A screenshot of a computer program

Description automatically generated

Figure : GET and POST for moods

GET and POST requests for my moods, only big differences here are the fact in my Mood.js I wanted a user to be able to view the graph based on different timeframes, that functionality still isn’t fully completed.

DB.js:  
  
A computer screen shot of code

Description automatically generated

Figure : Connection Pool for DB

This my connection pool for my database that is used by all my database components.

## 5.3. Other Sections

Authorisation:

Auth.js:



Figure : Import for Authentication

A screen shot of a computer code

Description automatically generated

Figure : Authentication for User Login

My auth.js contains my authentication where secure password handling occurs using Bcrypt.

The bcrypt.hash() function hashes the user's password using a salt.

The second argument 10 specifies the salt rounds, determining how many times the data is processed. Higher numbers increase security but also processing time.

I also use JWT tokens for authentication throughout the use of the application.

A screen shot of a computer screen

Description automatically generated

Figure : JWT token creation on Login

Once a user logins and Bcrypt is used to make sure the hashed passwords are matching we then generate the user a JWT token based on the key I have created and sets it to expire in 1 hour for increased user safety.

/middleware/auth.js:

A computer screen shot of code

Description automatically generated

Figure : Middleware for JWT tokens

This code implements a middleware function verifyToken for express.js that authenticates users requests using JWT’s. The complexity arises from extracting the token from the Authorization header, verifying it with jwt.verify using the secret key provided in my .env variables file.

Database:

A screenshot of a computer program

Description automatically generated

Figure : DB schema

The SQL code defines a simple yet robust database schema for managing users and their data, including conversations, messages, moods, journals and more to come. It uses foreign keys with cascading deletes to maintain integrity.

## 5.4. Conclusions

In the Prototype Development phase, I successfully created a Mental Health application that integrates both frontend and backend components. The frontend, built with React and Material-UI, features interactive charts, modals, and responsive design for an engaging user experience. The backend utilizes Express.js and PostgreSQL, implementing secure authentication, robust API endpoints, and efficient database schemas with proper error handling. This prototype effectively combines user-friendly interface design with secure data management, providing a solid foundation for future enhancements and testing.

# 6. Issues and Future Work

## 6.1. Introduction

In this chapter, we examine the challenges encountered during the development of Mindcare and we’ll outline the future work that can be implemented to improve Mindcare even further. By diving into issues, risks and proposed future works we can adapt and change the project to work with whatever sort of needs are accessed by the user and have a easy and effective development structure.

## 6.2. Issues and Risks

The main issues that will be faced for the project’s development are ensuring data security and no database breaches can occur by means such as SQL injection, optimising the AI chatbot and interface to make it seem as humane as possible, managing the application performance and UI design to make sure user’s stay engaged and lastly, implementing components that meet the needs of most users to keep Mindcare’s user base happy. Risks for Mindcare would be user engagement and easy of use. Without understanding these two risks Mindcare even with proper and robust development could still fail to meet it’s purpose.

## 6.3. Plans and Future Work

Plans are to get basic functionality of all components done. Front-end to back-end to database and external services is all setup at this moment. Currently the task at hand is to build all components discussed throughout this report atleast with basic functionality and then expand upon all components to provide final functionality. An example being would be the Chatbot, simple API is fundamental, making the chatbot personalised for each user based on their registering details and also their moods inputted throughout the application would be a lot better. Persistance and memory of each specific conversation per each user would also be further functionality to make Mindcare complex yet effective at what it does. The plan is ultimately to develop a base line where then development will only move up in complexity from there.

### 6.3.1. GANTT Chart

A screenshot of a computer

Description automatically generated

Figure : Gantt Chart

A screenshot of a project

Description automatically generated

Figure : Gantt Chart

# Bibliography

1. <https://www.publichealth.ie/mental-health#:~:text=Mental%20health%20refers%20to%20our,to%20the%20World%20Health%20Organization> – One in four people will have mental health conditions in their life
2. <https://pmc.ncbi.nlm.nih.gov/articles/PMC5553319/> - Stigma towards getting help
3. <https://help.snapchat.com/hc/en-us/articles/13266788358932-What-is-My-AI-on-Snapchat-and-how-do-I-use-it> - Snapchat AI
4. <https://pmc.ncbi.nlm.nih.gov/articles/PMC9505389/> - Pitfalls in mental health apps
5. <https://www.michiganmedicine.org/health-lab/mental-health-apps-may-help-those-waiting-care-study-finds> - Positive effect on people using mental health apps
6. <https://elomia.com/> - Mental Health chatbot application
7. <https://earkick.com/> - Mental Health chatbot application
8. <https://www.youper.ai/> - Mental Health Application
9. <https://www.mdpi.com/2076-3417/14/13/5889> - AI chatbots effectiveness on Mental Health
10. <https://www.jotpsych.com/post/ai-chatbots-psychotherapy?gad_source=1&gclid=Cj0KCQjwu-63BhC9ARIsAMMTLXRXnWW1VAeuM_H-uqDcZ_TcnkvOL3qugIJxz4bzDhOGho0QhePdTjEaAqJ2EALw_wcB> – Chatbots for university students
11. <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10242473/>] – User reviews on chat-based mental health applications
12. <https://www.psychreg.org/mood-tracking-offers-deeper-insight-emotional-well-being> - Mood-tracking leading to deeper insights on our emotions and mental well-being