

MINDSIGHT: EMPOWERING MENTAL WELLNESS

A Mini Project Report

Submitted in partial fulfillment of the requirements for the award of the degree of

Bachelor of Engineering

in

Computer Science and Engineering

By

ALAPATI NEHA SRAVANI	2456-21-733-131
MANYAPU SNEHA VARSHITA	2456-21-733-166
PEDDAPALLY MAITREYIE	2456-21-733-176

Under the guidance of

Ms.Anitha V

Assistant Professor, CSE



Department of Computer Science and Engineering

Gokaraju Lailavathi Women's Engineering College

(Affiliated to Osmania University, Approved by AICTE)

Nizampet, Hyderabad-500090

(2023-2024)

Department of Computer Science and Engineering
Gokaraju Lailavathi Womens Engineering College

Accredited by NBA & NAAC

(Affiliated to Osmania University, Approved by AICTE)

Nizampet, Hyderabad-500090

(2023-2024)



CERTIFICATE

This is to Certify that A Mini Project report entitled “**MINDSIGHT: EMPOWERING MENTAL WELLNESS**” is being submitted by Alapati Neha Sravani (2456-21-733-131), Manyapu Sneha Varshita (2456-21-733-166), Peddapally Maitreyie (2456-21-733-176) in partial fulfillment of the requirement of the award for the degree of Bachelor of Engineering in “**Computer Science and Engineering**” O.U., Hyderabad during the year 2024 is a record of bonafide work carried out by them under my guidance. The results presented in this project have been verified and are satisfactory.

Project Guide

Ms.Anitha V

Assistant Professor

Dept of CSE

H.O.D

Dr. Padmalaya Nayak

Professor & Head

Dept. of CS

External Examiner(s)



Department of Computer Science and Engineering
Gokaraju Lailavathi Womens Engineering College

Accredited by NBA & NAAC

(Affiliated to Osmania University, Approved by AICTE)

Nizampet, Hyderabad-500090

(2023-2024)

DECLARATION

We, Alapati Neha Sravani (2456-21-733-131), Manyapu Sneha Varshita (2456-21-733-166), and Peddapally Maitreyie (2456-21-733-176), hereby certify that the minor project entitled “**MINDSIGHT: EMPOWERING MENTAL WELLNESS**” has been submitted in partial fulfillment of the requirements for the award of the degree of Bachelor of Engineering in Computer Science and Engineering. This work has been carried out under the guidance of Ms. Anitha V, Assistant Professor, Department of Computer Science and Engineering, Gokaraju Lailavathi Women’s Engineering College, Hyderabad. We further declare that the results presented in this report are the original work we have conducted and have not been reproduced or copied from any source. Additionally, these results have not been submitted to any other university or institute for the award of any other degree or diploma.

ALAPATI NEHA SRAVANI	2456-21-733-131
MANYAPU SNEHA VARSHITA	2456-21-733-166
PEDDAPALLY MAITREYIE	2456-21-733-176

ACKNOWLEDGEMENT

It is our privilege and pleasure to express my profound sense of respect, gratitude, and indebtedness to our guide Ms.Anitha V, Assistant Professor, Department of Computer Science and Engineering, Gokaraju Lailavathi Women's Engineering College, for her inspiration, guidance, cogent discussion, constructive criticisms and encouragement throughout this dissertation work. We express our sincere thanks to Project Coordinator Mrs. Siva Naga Jyothi, Department of Computer Science and Engineering, Gokaraju Lailavathi Women's Engineering College, for his valuable suggestions and constant help in completing the work. We express our sincere gratitude to Dr. Padmalaya Nayak, Professor & Head of the Department of Computer Science and Engineering, at Gokaraju Lailavathi Women's Engineering College, for her precious suggestions, motivation, and cooperation. We express our sincere thanks to Dr. Akund Sai Hanuman, Principal, Gokaraju Lailavathi Women's Engineering College, Kukatpally, Hyderabad, for his encouragement and constant help. We extend our sincere thanks to all the teaching and non-teaching staff of the Information Technology Department for their support and encouragement. Last but not least, we wish to acknowledge my friends and family members for giving moral strength moral strength and helping us to complete this dissertation.

ABSTRACT

In recent years, owing to the fact that mental health is a relatively new field of concern, people have used digital technologies in different ways. This project aims to address the challenge of mental health management by proposing a holistic solution in the form of a web application called MINDSIGHT: EMPOWERING MENTAL WELLNESS. All in all, one can undoubtedly mention several useful aspects of this software that can be helpful for ordinary users as well as for certain clients with particular needs concerning their mental health. These include the questionnaire, activity ideas, affirmations, and meditation countdown services for the enhancement of the users mental health. On the frontend side of the application, a well-organized and well-coded supporting platform was designed through the HTML setup, CSS, and JavaScript, enabling the independent operation of the application. The backend, powered by Node.js and Express.js, provides a dependable and readily expandable method for handling user data, which is actually kept in MongoDB. Generally speaking, this web application has been designed to be helpful for people who wish to keep a close eye on their mental health.

Key Words: MongoDB, user-friendly interface, robust, self-awareness, innovative tools

CONTENTS

Chapter-1 INTRODUCTION	10
1.1 MOTIVATION	10
1.2 PROBLEM STATEMENT	10
1.3 PROJECT OBJECTIVE	11
Chapter-2 LITERATURE SURVEY	12
2.1 EXISTING SYSTEM	14
2.2 LIMITATIONS OF EXISTING SYSTEM	14
Chapter-3 SOFTWARE REQUIREMENT SPECIFICATION	15
3.1 SOFTWARE REQUIREMENTS	15
3.2 HARDWARE REQUIREMENTS	15
3.3 FUNCTIONAL REQUIREMENTS	15
3.4 NON-FUNCTIONAL REQUIREMENTS	16
Chapter-4 SYSTEM DESIGN	17
4.1 SYSTEM DESIGN	17
4.2 SYSTEM ARCHITECTURE	18
4.3 UML DESIGN	19
4.3.1 CLASS DIAGRAM	21
4.3.2 USE CASE DIAGRAM	22
4.3.3 COMPONENT DIAGRAM	23
4.3.4 SEQUENCE DIAGRAM	24
4.3.4 ACTIVITY DIAGRAM	25
4.4 TECHNOLOGY DESCRIPTION	26

Chapter-5 IMPLEMENTATION	28
5.1 IMPLEMENTATION	28
5.2 MODULES	30
5.3 EXECUTABLE CODE	32
Chapter-6 TESTING	40
6.1 TESTING DEFINITION	40
6.2 UNIT TESTING	40
Chapter-7 RESULTS	44
Chapter-8 CONCLUSION	50
7.1 CONCLUSION	50
7.2 FUTURE SCOPE	50
REFERENCES	50

LIST OF FIGURES

SNO.	LIST OF FIGURES	PAGE NO.
1.	System Architecture	18
2.	UML Hierarchy Diagrams	20
3.	Class Diagram	21
4.	Use Case Diagram	22
5.	Component Diagram	23
6.	Sequence Diagram	24
7.	Activity Diagram	25

LIST OF TABLES

Table No.	List of Tables	Page No.
2.1	Literature Survey	13
6.3.1	Login	43
6.3.2	Questionnaire	43
6.3.3	Activity Suggestions	44
6.3.4	SignUP	44

CHAPTER-1

INTRODUCTION

1.1 MOTIVATION

Mental health and wellness challenges have become one of the most significant public health concerns in today's world, marked by high-stress environments and a fast-paced lifestyle. There is growing concern worldwide about the mental well-being of individuals across various age groups. Despite this awareness, many people struggle to find the necessary resources or lack access to tools that can help them monitor and enhance their mental health. Our proposed mental health tracker aims to address these gaps by offering comprehensive solutions that empower users to take charge of their mental well-being. This platform will provide an interface that enables individuals to monitor their mental health progress through the use of questionnaires and quizzes. By assessing wellness scores, the application can suggest personalized activities, while additional features like affirmation buttons and meditation timers are designed to support users in their journey toward improved mental health.

1.2 PROBLEM STATEMENT

In today's hectic and stressful environment, maintaining mental health is crucial yet often overlooked. For many, self-reporting or routine visits to mental health professionals can be burdensome and insufficient for consistent mental health tracking. Additionally, the stigma surrounding mental health issues discourages many from seeking the necessary care. Although the significance of mental health is increasingly acknowledged, existing solutions often lack the comprehensiveness and personalization needed to address the diverse needs of individuals. Current mental health applications may offer limited features and fail to provide a holistic approach that includes daily support, personalized guidance, and self-assessment. Therefore, there is a need for an effective, user-friendly, and all-inclusive tool that allows users to monitor their mental health regularly, offering features like daily affirmations, meditation tools, and self-assessment questionnaires, along with personalized activity suggestions.

1.3 PROJECT OBJECTIVE

The goal of the MindSight: Mental Health Tracker project is to create a comprehensive and user-friendly application that transforms the way mental health is managed. This software is designed with a range of innovative tools and features that cater to the diverse needs of users who wish to monitor and improve their mental well-being. The main objective is to empower users by providing a holistic approach to mental health management. Among its key features are meditation timers, activity scheduling, symptom tracking, and mood monitoring, which allow users to record their thoughts, emotions, and daily activities. MindSight aids users in becoming more self-aware, helping them to identify patterns and triggers that influence their mental health.

Additionally, the project seeks to develop a strong and user-friendly front end using HTML, CSS, and JavaScript, ensuring an optimal user experience. The backend, built on Express.js and Node.js, guarantees scalable and reliable handling of user data, which is securely stored in MongoDB. MindSight is intended to help users tackle their daily mental health challenges and support them with features like daily affirmations, fostering a positive mindset. The MindSight mental health tracker aims to become a vital tool for those actively managing their mental well-being. By offering users the resources they need to build healthy habits, manage stress, and achieve both psychological and emotional balance, the application provides easy access to and management of mental health information across multiple platforms, supporting a dynamic and rewarding journey toward better mental health.

CHAPTER-2

LITERATURE SURVEY

1. Mental Health Tracker :

The work under analysis is the research paper of four students of Vidyodaya College of Arts and Science, India: Shreeya Mayekar, Dakshali Merya, Kamini Patil, and Akshata Sonawane, entitled “Mental Health Tracker” published in the International Research Journal of Engineering and Technology (IRJET 2019) and dedicated to the description of the application for improving and maintaining the Mental Stressing own's scenario of the present world, this paper notes that mental health disorders have been amplified by such circumstances as COVID-19. Below is the list of self-assessments as well as activities that are available in the app such as breathing exercises, joke activities, and sound therapy. Mood diary, quotes, a list of tasks – these and other functions transform the application into comprehensive therapy without having a disease.

The study therefore advocates for the improvement of the use of mobile technology in mental health management with ease of use being the main factor. Users can measure their state of mind with time, receive news and be provided with recommendations that have been made especially for them in order to improve the state of mind. The research also describes the future work, containing the further improvement of the GUI, the improvement of security level, and the extension of the number of available kinds of operations and choices for the user. All in all, the paper provides extensive guidelines to create an efficient and individually aimed application that could facilitate in defining specific mental health problems.

2. Companion App: A Mental Health Tracker :

The research paper titled "COMPANION APP: Therefore, one of the themes that can be discussed is “A MENTAL HEALTH TRACKER: Development of an Android mental health application and the possible benefits. ” The paper also reveals the statistics given by the WHO as the attitude to mental or neurological disorders as the global challenges: at least one in four people in the world will experience mental health issues at some point of their life. On the same note, traditional services tend to be costly, difficult to access, thereby entail social cultural as well as self-identity stigmatizations, which the proposed app must address. Here the features of self-assessment, resources, personally customized approaches and options to reach out to a mental health worker are designed to increase the use and uptake of mental health products. Implemented features of the application include the following aspects that assist to enhance the engagement and

efficiency to the users. These are the outcome based reward system, the missions that have to be accomplished in order to level up, the breathing exercises and the discussion boards.

Its attributes such as Artificial Intelligence and natural language processing guarantee that a user has personal and evidence-based support for mental health challenges. In addition, the app safeguards user data so as to ensure that records of the users' mental health are not compromised. For the technical aspect concerning the concept of development, Android Studio as the IDE, Java as the language and Firebase for the backend is used so as to maintain the technical realism. Altogether, the discussed app suggests the main objective of improving availability of mental health services as for affordability, personalized approaches, and global availability.

Year	Authors	Title	Methodology	Features	Disadvantages
2023	Shreeya Mayekar, Dakshali Merya, Kamini Patil, Akshata Sonawane	“Mental Health Tracker “	“Mental Health Tracker “	Self-Assessment, Sound Activity, Breathing Exercise, To-Do List, Mood Tracker	Use of user self-completion which may at times be misleading.
2022	Apoorva Bagul, Pooja Sinkar, Priyanka Jadhav, Deepali Ahire, Prof.D.D.Sharma	“A Mental Health Tracker built using Flutter and Firebase”	Daily questions using natural language.	User-friendly system, Hacking secure, Centralized system, Security for user data	Lack of Accuracy, Engagement, Dependence
2023	Mayureshwar Jadhav, Rohit Kulkarni, Bhagyashri Nage, Amit Dixit, Vishal Walnuj	“Companion App: A Mental Health Tracker”	Development of Android-based mental health app designed to help users manage their mental health issues.	Personalized treatment plans, mental health assessments, self-help tools, access to licensed professionals	Limited human interaction, potential privacy concerns

2.1 Existing System

Today numerous individuals turn to smartphone programs and other technologies to approach mental wellbeing. Many of these systems offer additional amenities like mood logging, listening to guided meditations, take self-diagnostic questionnaires, and have access to articles concerning mental health. Even though these tools have greatly enhanced access to mental health services, the options frequently encounter several challenges. Currently, users may not receive the required number of features in an app, the apps may not be contextually relevant and personalized and often, the support is not as profound as required for managing mental health. Most current systems offer generic mental health information and services to clients with no customizability to clients' conditions.

2.2 Limitations of Existing System

1. Lack of Personalization
2. Insufficient Data Security and Privacy
3. Lack of Real-Time Monitoring and Feedback
4. Reliance on Internet Connection

CHAPTER-3

SOFTWARE REQUIREMENT SPECIFICATION

This chapter focuses on the features in both software and hardware that should be incorporated while developing the Mental Health Tracker application.

3.1 SOFTWARE REQUIREMENTS

Operating System: Windows 10 or later versions

Backend Framework: Node.js with Express.js

Database: MongoDB

Templating Engine: Board – brd

Frontend Languages: HTML, CSS, JavaScript

3.2 HARDWARE REQUIREMENTS

System: Intel i5 processor, or any other higher processor.

Storage: Sufficient storage

3.3 FUNCTIONAL REQUIREMENTS

These requirements should illustrate the basic characteristics of what the system shall provide. That brings what could be described as integral performance which the end-user would have expected to be provided in the final product.

- User Registration and Authentication
- Activity Scheduling
- Meditation Timers
- Affirmations module
- Feedback Submission
- Questionnaire Feature
- Progress Tracking and Reporting
- User Interface for Real-Time Interaction

3.4 NON-FUNCTIONAL REQUIREMENTS

Non-functional requirements are those that describe and constrain qualities that are supposed to be provided in the system other than the functionality. These requirements offer the confidence that the system is going to perform well and with little effort in different scenarios.

- Portability
- Security
- Maintainability
- Reliability
- Scalability
- Performance
- Reusability
- Flexibility

CHAPTER-4

SYSTEM DESIGN

4.1 System Design

In this phase, the system and software design documents are prepared as per the requirement specification document. This helps define overall system architecture. There are two kinds of design documents developed in this phase:

High-Level Design (HLD)

- Brief description and name of each module
- An outline of the functionality of every module
- Interface relationship and dependencies between modules
- Database tables identified along with their key elements
- Complete architecture diagrams along with technology details

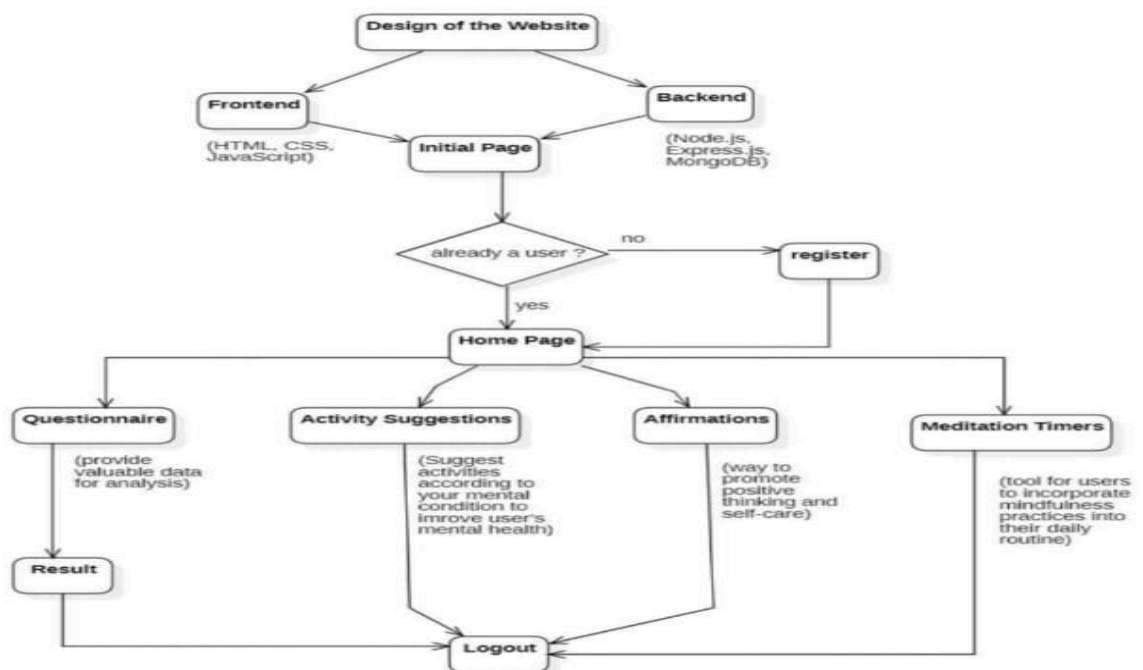
Low-Level Design(LLD)

- Functional logic of the modules
- Database tables, which include type and size
- Complete detail of the interface
- Addresses all types of dependency issues
- Listing of error messages

4.2 SYSTEM ARCHITECTURE:

MINDSIGHT: EMPOWERING MENTAL WELLNESS

PROCEDURE/ PROPOSED METHODOLOGY



4.3 UML Design:

Unified Modeling Language (UML) is a visual tool that plays a crucial role in designing software systems. It was officially standardized in 1997, with the Object Management Group (OMG) recognizing its importance. Later, the International Organization for Standardization (ISO) also acknowledged UML, further establishing its significance. UML is widely used because it provides a clear way to visualize both the structure and behavior of software systems, making complex designs more understandable and easier to communicate among developers and stakeholders.

Do we need UML?

- For complex applications, multiple teams need a clear way to communicate. UML provides a shared visual language that keeps everyone aligned.
- Business stakeholders often don't understand code.
- UML helps teams visualize processes and system structure, reducing misunderstandings and saving time taken during development.
- UML supports object-oriented design and analysis by using diagrams to show relationships and interactions within the system.

The Primary goals in the design of the UML are as follows:

- Provide users with a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
- Extendibility and specialization mechanisms are provided to extend the core concepts.
- Be independent of particular programming languages and development processes.
- A formal basis for understanding the modeling language is provided.
- Encourage the growth of the OO tools market.
- Support higher-level development concepts such as collaborations, frameworks, patterns, and components.
- Integrate best practices.

Types of UML Diagrams

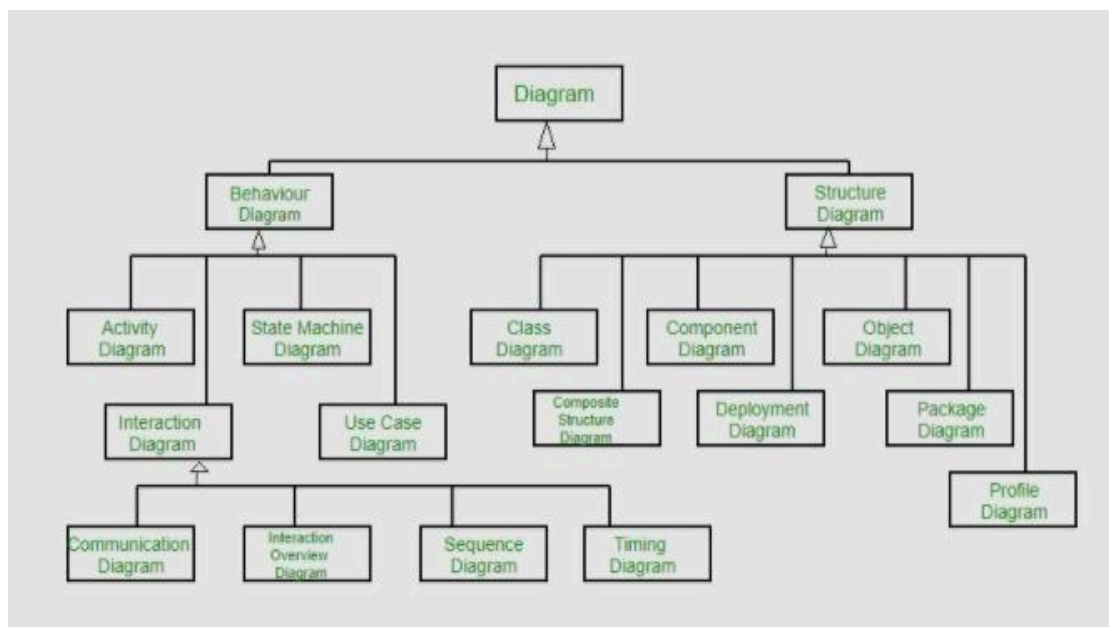
Structural Diagrams:

Capture static aspects or structure of a system. Structural Diagrams include: Component Diagrams, Object Diagrams, Class Diagrams and Deployment Diagrams.

Behavior Diagrams:

Capture dynamic aspects or behavior of the system. Behavior diagrams include: Use Case Diagrams, State Diagrams, Activity Diagrams and Interaction Diagrams.

The image below shows the hierarchy of diagrams according to UML



UML Hierarchy diagrams

4.3.1 CLASS DIAGRAM:

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.

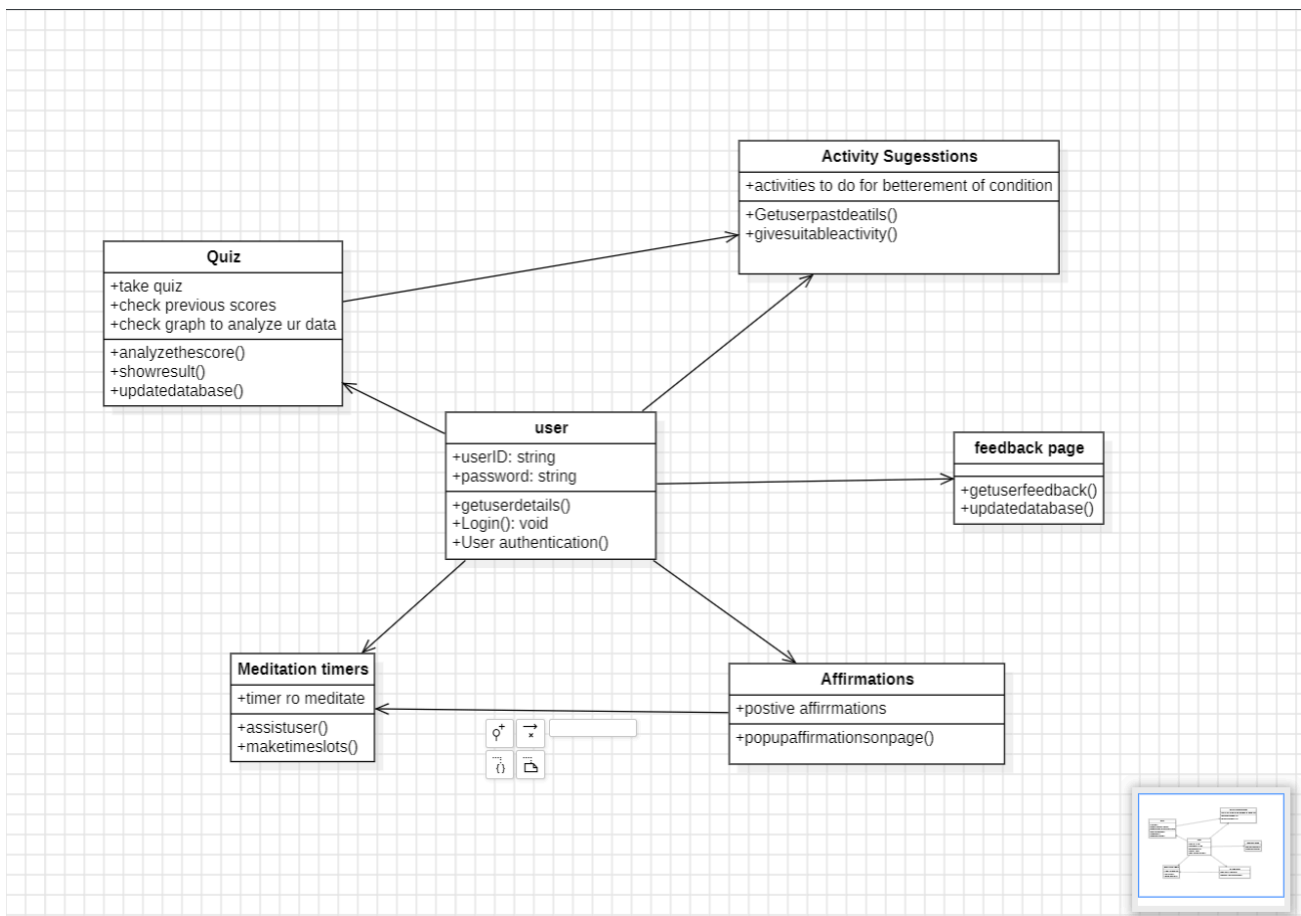


Figure-4.3.1.1 Use Case Diagram

4.3.2 USE CASE DIAGRAM:

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis.

Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases.

The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted

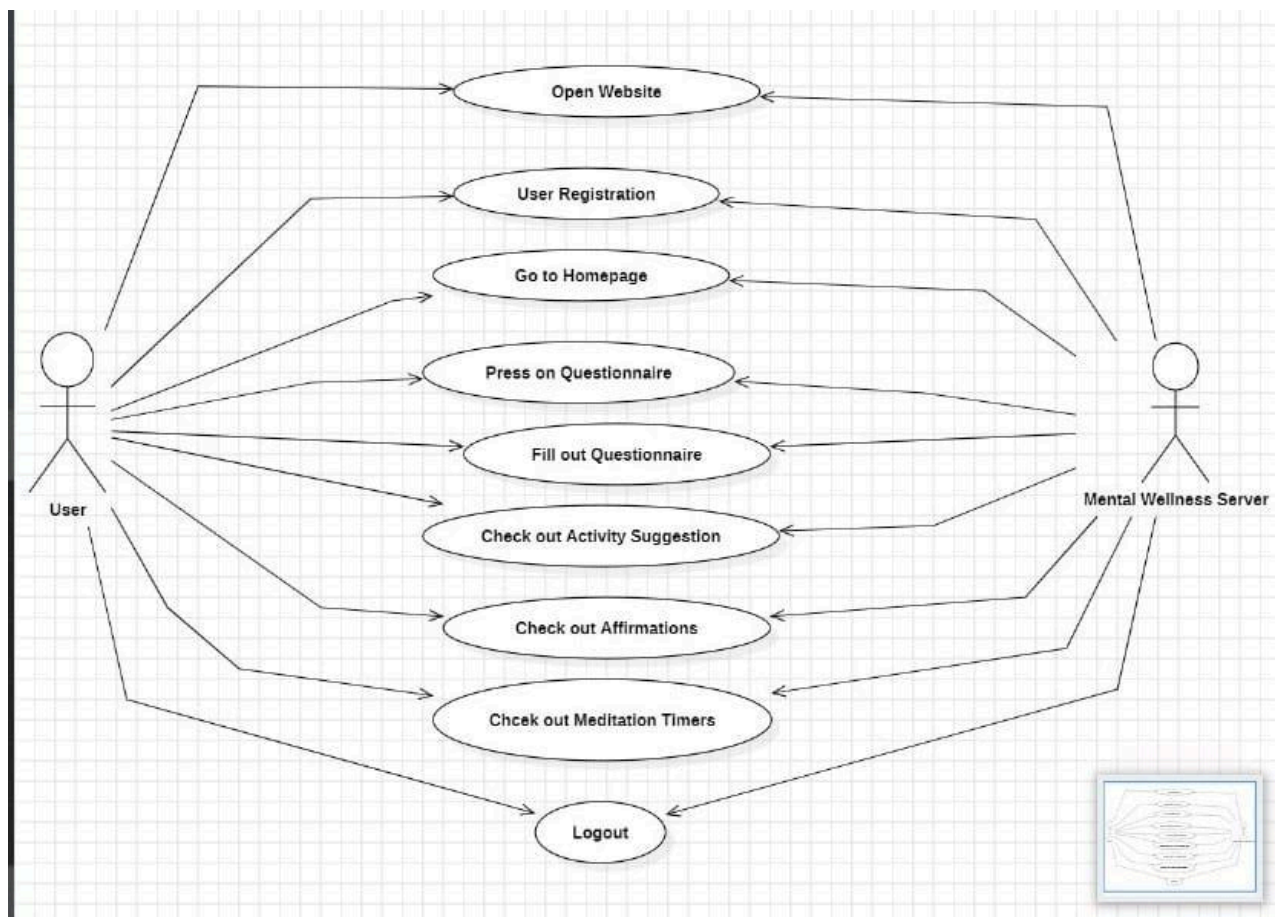


Figure-4.3.2.1 Use Case Diagram

4.3.3 COMPONENT DIAGRAM:

A component diagram, also known as a UML component diagram, describes the organization and wiring of the physical components in a system. Component diagrams are often drawn to help model implementation details and double-check that every aspect of the system's required functions is covered by planned development.

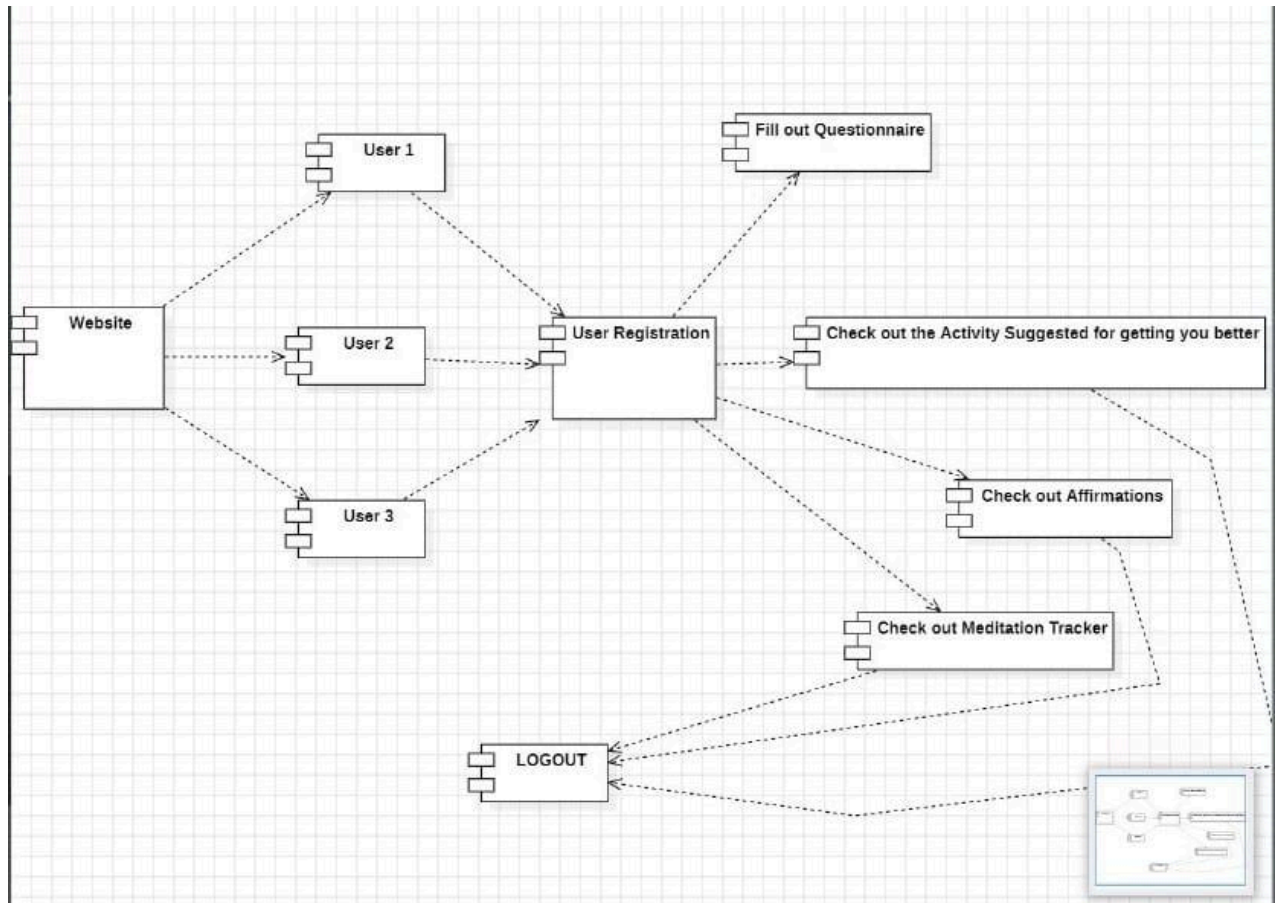


Figure-4.3.3.1 Component Diagram

4.3.4 SEQUENCE DIAGRAM:

A sequence diagram in Unified Modeling Language (UML) is an interaction diagram showing how processes operate and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.

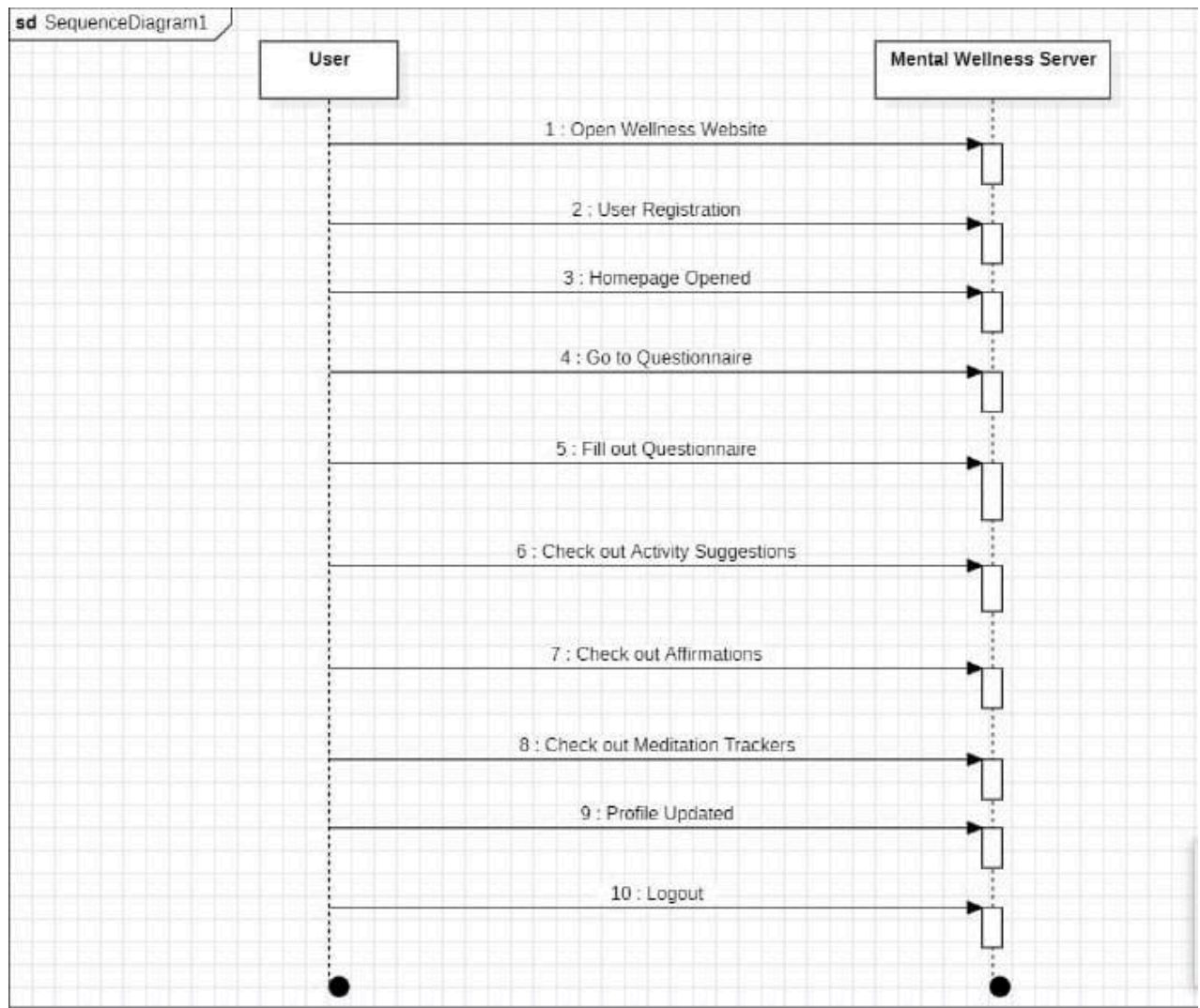


Figure-4.3.4.1 Sequence Diagram

4.3.5 ACTIVITY DIAGRAM:

In UML, an activity diagram is used to display the sequence of activities. Activity diagrams show the workflow from a start point to the finish point detailing the many decision paths that exist in the progression of events contained in the activity.

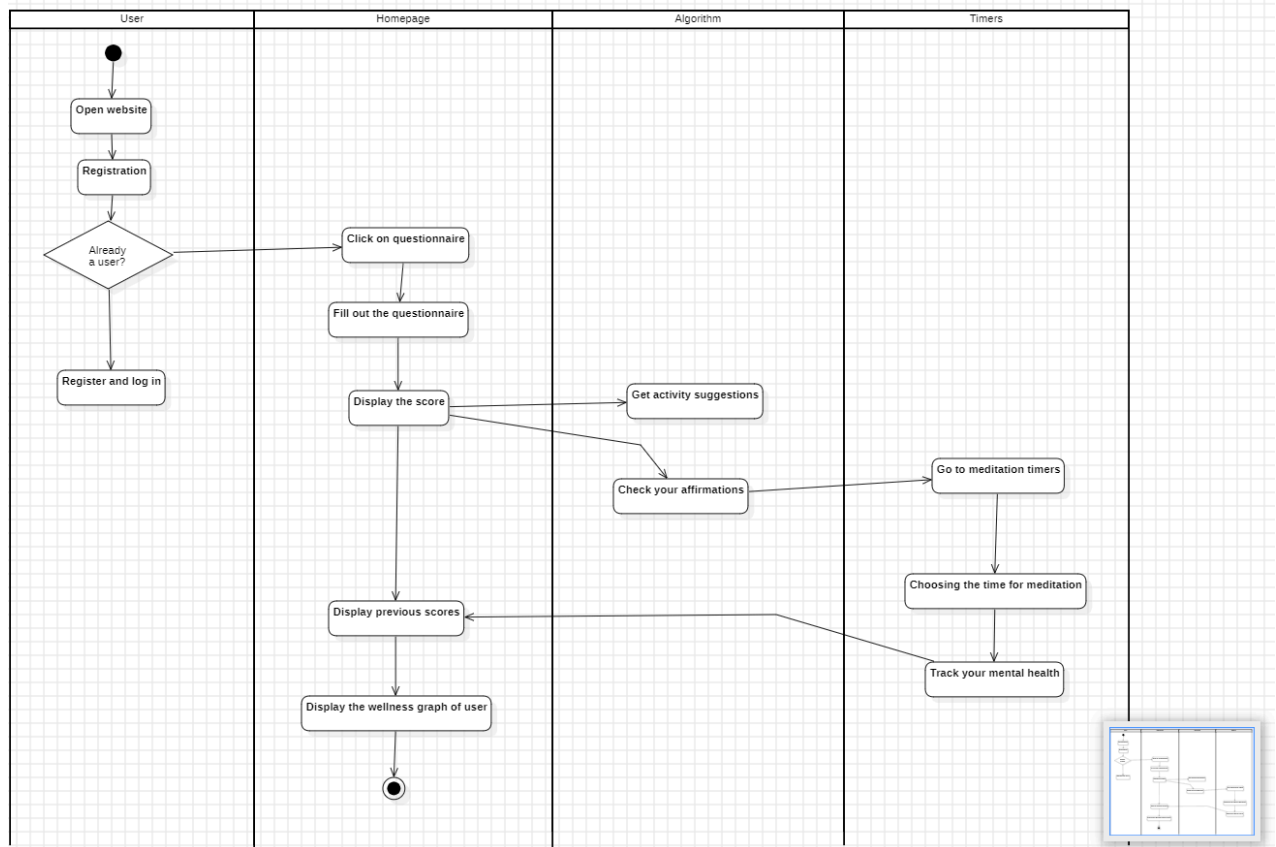


Figure-4.3.5.1 Activity Diagram

4.4 TECHNOLOGY DESCRIPTION

HTML

HTML is used in the mental health tracker application to create and format the web pages a user will be exposed to. It provides the layout and structural organization for content from the app, for example, forms capturing information on mood, buttons for entry submission, space to indicate progress over time. Tags in HTML would do the formatting of these elements in front of the user in a clear and user-friendly way, making them accessible on different devices.

CSS

The CSS (Cascading Style Sheets) is applied in the mental health tracker app to manage the look and the position of the web pages of the application. It improves the usability of the app by providing a proper layout and design such as colors, font size and types, spaces, etc. CSS makes sure that the app is good to look at, looks professional and balanced the layout so it looks good on different devices. This goes a long way in developing a fascinating and friendly platform for monitoring and managing mental disorders.

JAVASCRIPT

JavaScript in a mental health tracker application is a part of interactivity and dynamic functionalities on the web page. This real-time updating includes: doing personal mood trends' automatic calculations, validation of user inputs, and providing instant feedback without the page reloading. Furthermore, it can do work related to local data storage, chart creation for progress visualization, and interaction with external APIs to give more functionality. This makes the app more interactive, responsive, and personalized towards the user.

MONGODB

It serves as the database in a mental health tracker application, storing all the user data. It manages vast amounts of information, which will include user profiles, mood entries, journal notes, and progress reports, efficiently. This is possible because MongoDB, with

its document-based structure, can easily store complicated data types, which are apt for storing varied kinds of user input. It also supports fast querying and data retrieval, enabling a number of other features that include tracing mood patterns over time and generating personalized insights.

REACT JS

It builds a responsive and interactive user interface for the mental health tracker app. Using React.js, it can easily create reusable UI elements like mood tracking forms, progress charts, and dashboards able to update dynamically while users continue to use the app. Due to its component-based architecture, React makes it easier to handle intricate user interaction and ensures an application stays fast and responsive. This will ensure a smoother user experience with real-time updates and provide an engaging interface for mental health monitoring.

CHAPTER-5

5.1 IMPLEMENTATION:

Environment Setup:

To initiate the Mental Health Tracker application, we first created a strong Node environment that can run the application smoothly. JS and npm. This is why we targeted Express. JS for the backend is that using it as a framework gives reliability and convenience to construct a potent server. MongoDB was used for the database because it is malleable and will be used as the company expands. A4 handlebars, known as hbs, were used when executing HTML templates to enhance the consumer experience.

Frontend Development:

As for the front end of the Mental Health Tracker, the design was very UX-friendly. This was to enable us to develop a clean, standard-based user interface using HTML, CSS, and JavaScript. It is live and comprises custom backgrounds and/or fonts. They also contain some straightforward associations with such elements as quizzes, suggestions on activities, affirmations, and meditation timers. Special attention is paid to the login page to make a guest feel warm and to provide them with an up-to-date interface right from the start.

Backend Development:

At the core of this application, the backend is established, accomplished by Node.js and Express.js, which provide something solid to support the application. Authentication of users and responses, management of feedback, handling of data, the routes to present various pages, and management of feedback submissions are some of the prominent tasks of this module.

User Authentication:

This involved user verification, and the credentials provided matched the ones saved in MongoDB. After a proper login, the interface of the system will be moved to the home screen, where most of the mental health tracking features of the app are disclosed.

Feedback Management:

A feature for feedback was made available that collects user input and keeps it safe in the

database. It is also possible to see all the feedback if one wants to gain some understanding of the users' experiences.

Template Engine Integration:

This entailed the construction of HTML output on the fly with the possibility of using handlebars, hbs partials, and templates. The setting provides rendering for several viewpoints, depending on user actions, while simultaneously offering the consistency of pages and easier management.

Management of User Data:

Feedback given and login details typed to the site by users are protected securely in MongoDB. Optimization of the database is done w.r.t. storing and retrieving the user data. Sensitive information, which is to be stored in the database, is encrypted to make sure that using the app is as fluent and safe as possible.

Implementation of Features:

- **Activity Scheduling:** Patients can plan and set activities that are healthy for their mental health with options like reminders and alarms.
- **Meditation Timers:** This application contains modes that have timers for meditating, thus helping people to be mindful, which improves mental health.
- **Affirmations Feature:** This feature helps users develop a positive mindset.
- **Questionnaire Feature:** Users can complete a questionnaire to assess their mental health. The application calculates and displays scores based on their responses, allowing users to track their progress over time.
- **Progress Graph:** A graphical representation of the user's scores over time is available, providing visual insights into their mental health progress.

Safety Features:

Given the privacy risks associated with giving out such information, we took all possible security measures. Access procedures are guarded, and the information to be written in the database is encrypted. We also use HTTPS to keep communication between the frontend and backend processes safe.

Deployment:

For the Mental Health Tracker, a cloud-based structure was designed, which permitted its easy use and escalation among the user base. To deploy the software, setting up the environment on the

server, connecting the database, and fine-tuning the app to provide smooth sequential use by many users were necessary.

5.2 MODULES:

User Authentication Module:

Of the entire security model of the Mental Health Tracker application, the User Authentication Module is nothing but the foundation. This module contains the user registration information that will enable new users to register with information that is relevant to their well-being. Upon registration, the credentials of the user will be saved in MongoDB, but passwords will be hashed so that information security is not compromised.

It is also liable for the login process of the user, where it checks the credentials with the data available with the system before allowing the user to access all the facilities of the application. After successful user authentication, the users are redirected to the main page, where they can access different tracking interfaces dealing with mental health. The module is also intended to regulate the user sessions safely; of course, to prevent unauthorized access some measures would have been implemented.

Feedback Management Module:

This module is intended to gather and handle the app feedback of the users. From the graphic presentation of processes, it is clearly seen that the processes provide a friendly environment for the users to key in a description based on an idea, suggestion, or note of problems. All the mentioned or developed feedback is saved by secure methods in the database, which means that it is available for the developers to read and ‘work on’.

It has options that allow one to view all feedback collected with the help of this module and thus get an idea of users’ perceptions. The application can therefore be refined similarly to become user-friendly as well as effective for the intended purpose, as supported by the overall analysis of feedback.

Activity Scheduling Module:

The Activity Scheduling Module aims to help schedule and execute activities that help enhance mental health. It allows a user to plan an exercise, relaxation, or anything to do with hobbies, including alarms for reminding the user to get back to the activity. The principle behind the use of this module lies in the regularity of such activities or behaviors that would foster good mental health, and therefore it works to foster a balanced lifestyle among users. In this way, a user,

for example, can successfully set mental health goals and, at the same time, build a mental health-promoting pattern of living that will help them to be mentally healthy in the future.

Questionnaire Module:

This module enables users to complete mental health questionnaires, calculate scores based on their responses, and review their answers. The system displays current scores and historical data, allowing users to monitor their mental health progress over time.

Affirmations Module:

The Affirmations feature in the MINDSIGHT application is designed to help users foster a positive mindset by offering daily motivational statements. This feature encourages consistent positive thinking, which can contribute to improved mental well-being over time. By regularly engaging with affirmations, users are empowered to build resilience and maintain a constructive outlook on life.

Progress Graph Module:

The Progress Graph Module generates visual representations of user scores over time. It provides a graphical overview of mental health trends, helping users track their progress and gain insights into their emotional and mental well-being.

Meditation Timers Module:

The Meditation Timers Module gives the users the tools they need to conduct and time the meditation themselves. This module provided different timer preferences, hence giving the user an option to choose from to meet the required set time while meditating.

The timers are also to be rather straightforward and minimalistic in their design to enhance the relaxation process. Just by integrating this module into the user's daily schedule, their cognitive health will be improved, and they will reduce stress, thereby improving their mental well-being.

User Data Management Module:

The user data management module has been established to be responsible for handling the totality of users' data in the system. Of the collected data, it keeps login details, mood records, symptom reports, and feedback in the safe MongoDB database. The module equally ensures that all the private information is protected by encryption and confined to memory before being stored, thus making certain that the user's privacy is protected.

It also caters to data acquisition with the ability to deliver the desired information to an application in a timely and efficient manner. This will still ensure that whatever applications are being used will be effective in their delivery, and most importantly, the user end will be satisfied.

5.3 EXECUTABLE CODE

index.js:

```
const express = require("express");
const app = express();
const path = require("path");
const hbs = require("hbs");
const { User, Feedback } = require("../mongodb");

const templatePath = path.join(__dirname, '../templates');
const publicDirectoryPath = path.join(__dirname, '../public');
const partialsPath = path.join(__dirname, 'partials');

// Setup handlebars engine and views location
app.set("view engine", "hbs");
app.set("views", templatePath);
hbs.registerPartials(partialsPath);

// Setup static directory to serve
app.use(express.static(publicDirectoryPath));
app.use(express.json());
app.use(express.urlencoded({ extended: false }));

// Routes
app.get("/", (req, res) => {
  res.render("login");
});

app.get("/signup", (req, res) => {
  res.render("signup");
});

app.post("/signup", async (req, res) => {
  const data = {
    name: req.body.name,
    password: req.body.password,
  };

  await User.insertMany([data]);
  res.redirect("/");
});
```



```

app.post("/login", async (req, res) => {
  try {
    const user = await User.findOne({ name: req.body.name });
    if (!user) {
      res.send("User not found. Please sign up.");
      return;
    }

    if (user.password === req.body.password) {
      res.sendFile(path.join(publicDirectoryPath, 'page.html'));
    } else {
      res.send("Wrong password");
    }
  } catch (error) {
    console.error(error);
    res.send("An error occurred. Please try again.");
  }
});

// Route to handle feedback form submission
app.post("/contact", async (req, res) => {
  const feedback = new Feedback({
    name: req.body.name,
    message: req.body.message,
  });

  try {
    await feedback.save();
    res.render("feedback", { message: "Feedback received. Thank you!" });
  } catch (error) {
    console.error(error);
    res.render("feedback", { message: "An error occurred. Please try again." });
  }
});

// Route to view all feedbacks
app.get("/feedbacks", async (req, res) => {
  try {
    const feedbacks = await Feedback.find();
    res.json(feedbacks);
  } catch (error) {
    console.error(error);
    res.send("An error occurred. Please try again.");
  }
});

// Start the server
app.listen(3000, () => {

```

```
    console.log("Server is running on http://localhost:3000");
  });
```

mongodb.js:

```
const mongoose = require("mongoose");

mongoose.connect("mongodb://localhost:27017/MindSightUserDetails")
  .then(() => {
    console.log("MongoDB connected");
  })
  .catch(() => {
    console.log("Failed to connect to MongoDB");
  });

// User schema for login information
const LogInSchema = new mongoose.Schema({
  name: {
    type: String,
    required: true
  },
  password: {
    type: String,
    required: true
  }
});

// Feedback schema for storing feedback messages
const FeedbackSchema = new mongoose.Schema({
  name: {
    type: String,
    required: true
  },
  message: {
    type: String,
    required: true
  },
  createdAt: {
    type: Date,
    default: Date.now
  }
});

// Models
const User = mongoose.model("LogInCollections", LogInSchema);
const Feedback = mongoose.model("Feedback", FeedbackSchema);

module.exports = { User, Feedback };
```

page.html:

```

<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>MindSight</title>
  <style>
    body {
      height: 100vh;
      width: 100%;
      background: url("Screenshot%20(196).png") center/cover no-repeat;
      margin: 0;
      padding: 0;
      font-family: Arial, sans-serif;
    }
  </style>
  <link rel="stylesheet"
href="https://fonts.googleapis.com/css2?family=Material+Symbols+Rounded:opsz,wght,FILL,GRAD@48,
400,0,0">
  <link rel="stylesheet" href="page.css">
  <script src="page.js" defer></script>
</head>
<body>
  <header>
    <nav class="navbar">
      <span class="hamburger-btn material-symbols-rounded">menu</span>
      <a href="page.html" class="logo">
        
        <h2>MindSight</h2>
      </a>
      <ul class="links">
        <span class="close-btn material-symbols-rounded">close</span>
        <li><a href="q.html">Quiz</a></li>
        <li><a href="ac.html">Activity Suggestions</a></li>
        <li><a href="a.html">Affirmations</a></li>
        <li><a href="med.html">Meditation Timers</a></li>
        <li><a href="contact.html">Contact Us</a></li>
        <li><button id="logout-btn" style="font-size: 20px; cursor: pointer; border: none;
background-color: #d98389; color: white; border-radius: 5px;">Log Out</button></li>
      </ul>
    </nav>
  </header>
</body>
</html>

```

q.js:

```

document.addEventListener('DOMContentLoaded', () => {

```

```

const contentContainer = document.getElementById('content-container');
const menuContainer = document.getElementById('menu-container');
let userScore = null; // Variable to store the user's score

// Load the logged-in user's username from localStorage
const username = localStorage.getItem('username');

// Function to get previous scores for the logged-in user
function getPreviousScores() {
  return JSON.parse(localStorage.getItem(`${username}_previousScores`)) || [];
}

// Function to set previous scores for the logged-in user
function setPreviousScores(scores) {
  localStorage.setItem(`${username}_previousScores`, JSON.stringify(scores));
}

// Load previous score from localStorage if available
const previousScores = getPreviousScores();
if (previousScores.length > 0) {
  userScore = previousScores[previousScores.length - 1].score;
}

// Function to redirect to activity suggestions if the test is completed
function showActivitySuggestions() {
  if (userScore !== null) {
    window.location.href = 'ac.html'; // Redirect to activity suggestions page
  } else {
    alert('Please complete the test first.');
```

```

  }
}

// Function to randomly select a specified number of questions from the array
function getRandomQuestions(questions, num) {
  const shuffled = questions.sort(() => 0.5 - Math.random());
  return shuffled.slice(0, num);
}

```

```

// Function to create a question block
function createQuestionBlock(question, index) {
  const questionBlock = document.createElement('div');
  questionBlock.classList.add('question-block');

```

```

  const questionText = document.createElement('p');
  questionText.textContent = question;
  questionBlock.appendChild(questionText);

```

```

  const options = document.createElement('div');

```

```

options.classList.add('options');

const optionValues = {
  'Never': 0,
  'Rarely': 1,
  'Sometimes': 2,
  'Often': 3,
  'Always': 4
};

Object.keys(optionValues).forEach(option => {
  const label = document.createElement('label');
  label.classList.add('option');

  const input = document.createElement('input');
  input.type = 'radio';
  input.name = `q${index}`;
  input.value = optionValues[option];

  const circle = document.createElement('span');
  circle.classList.add('circle');

  const labelText = document.createElement('span');
  labelText.classList.add('label');
  labelText.textContent = option;

  label.appendChild(input);
  label.appendChild(circle);
  label.appendChild(labelText);
  options.appendChild(label);
});

questionBlock.appendChild(options);
return questionBlock;
}

// Function to load and display the wellness check-in test
function loadTest() {
  contentContainer.innerHTML = '<h1>Wellness Check-in</h1><form
id="survey-form"></form><button type="submit" form="survey-form">Submit</button>';
  const form = document.getElementById('survey-form');

  const selectedQuestions = getRandomQuestions(questions, 15);
  selectedQuestions.forEach((question, index) => {
    const questionBlock = createQuestionBlock(question, index + 1);
    form.insertBefore(questionBlock, form.lastElementChild);
  });
}

```

```

form.addEventListener('submit', function(event) {
  event.preventDefault();
  const formData = new FormData(this);
  let totalScore = 0;
  formData.forEach((value) => {
    totalScore += parseInt(value);
  });

  // Store the score in localStorage
  const timestamp = new Date().toLocaleString();
  const previousScores = getPreviousScores();
  previousScores.push({score: totalScore, timestamp: timestamp});
  setPreviousScores(previousScores);

  alert('Survey submitted! Your score is ' + totalScore);
  loadPreviousScores();
});

menuContainer.style.display = 'none'; // Hide the menu
}

// Function to load and display the previous scores
function loadPreviousScores() {
  const previousScores = getPreviousScores();
  contentContainer.innerHTML = '<h1>Previous Scores</h1><ul>' + previousScores.map(scoreEntry
=> `<li>Your previous score: ${scoreEntry.score} (on ${scoreEntry.timestamp})</li>`).join("") + '</ul>';
  contentContainer.innerHTML += '<button onclick="goBack()">Go Back</button>';
}

// Function to load and display the score graph
function loadScoreGraph() {
  const previousScores = getPreviousScores();
  contentContainer.innerHTML = '<h1>Score Graph</h1><canvas id="scoreChart" width="400"
height="400"></canvas>';
  contentContainer.innerHTML += '<button onclick="goBack()">Go Back</button>';

  const ctx = document.getElementById('scoreChart').getContext('2d');
  new Chart(ctx, {
    type: 'line',
    data: {
      labels: previousScores.map((entry, index) => index + 1),
      datasets: [{
        label: 'Scores',
        data: previousScores.map(entry => entry.score),
        borderColor: 'rgba(75, 192, 192, 1)',
        borderWidth: 2,
        fill: false
      }]
  })
}

```

```

    },
    options: {
      scales: {
        y: {
          beginAtZero: true
        }
      }
    }
  });
}

// Function to go back to the main menu
function goBack() {
  contentContainer.innerHTML = "";
  menuContainer.style.display = 'flex';
}

// Event listeners for menu buttons
document.getElementById('complete-test').addEventListener('click', loadTest);
document.getElementById('previous-scores').addEventListener('click', loadPreviousScores);
document.getElementById('score-graph').addEventListener('click', loadScoreGraph);
document.getElementById('activity-suggestions').addEventListener('click', showActivitySuggestions);
});

```

CHAPTER-6

TESTING

6.1 TESTING DEFINITION:

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub assemblies, assemblies and/or a finished product. It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of tests. Each test type addresses a specific testing requirement.

6.2 Unit Testing

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

Test strategy and approach

Field testing will be performed manually and functional tests will be written in detail.

Test objectives

- All field entries must work properly.
- Pages must be activated from the identified link.
- The entry screen, messages and responses must not be delayed.

Features to be tested

- Verify that the entries are of the correct format
- No duplicate entries should be allowed
- All links should take the user to the correct page.

Integration Testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects. The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

Acceptance Testing

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

Test Results: All the test cases mentioned above passed successfully. No defects encountered. Since, the gray box testing includes access to internal coding for designing test cases. Gray box testing is performed by a person who knows coding as well as testing.

Outcomes Possible:

Pass: The test case successfully validates the expected behavior of the application, indicating that specific functionality works as intended.

Fail: The test case fails to validate the expected behavior, indicating a defect or issue in the application. This outcome requires further investigation and fixing the identified problem.

Error: An error occurs during the test execution due to unexpected system behavior or exceptions. This could indicate a bug or potential issue that needs to be addressed.

Blocked: The test case cannot be executed due to external dependencies or environmental constraints. This outcome indicates that the test case is blocked and cannot be validated at the moment.

Skipped: The test case is intentionally skipped from execution, typically due to low priority or specific conditions that prevent it from being applicable at the current stage of testing.

Test case

User requirements:

Login:

User Case Id	Mental Health Tracker
Use Case Name	Login/SignUp Button
Description	Input correct credentials to access the website
Primary Factor	User
Pre Condition	User must open the application
Post Condition	Displaying the homepage of the application
Frequency of Use Case	Many Times
Alternative Use Case	N/A
Use Case Diagrams	
Attachments	N/A

Table-6.3.1

Questionnaire:

User Case Id	Mental Health Tracker
Use Case Name	Filling Out Mental Health Questionnaire
Description	The user answers a set of questions
Primary Factor	User
Pre Condition	User has navigated the questionnaire section
Post Condition	User's mental health score is calculated based on their responses.
Frequency of Use Case	Many Times
Alternative Use Case	Can directly check their previous scores
Use Case Diagrams	
Attachments	N/A

Table-6.3.2

Activity Suggestions:

User Case Id	Mental Health Tracker
Use Case Name	Providing Activity Suggestions
Description	Process of providing activity suggestions to the user based on their mental health scores
Primary Factor	User
Pre Condition	The user has completed the mental health questionnaire
Post Condition	Users receive personalized activity suggestions based on their mental health score
Frequency of Use Case	Many Times
Alternative Use Case	N/A
Use Case Diagrams	
Attachments	N/A

Table-6.3.3

Sign Up:

User Case Id	Mental Health Tracker
Use Case Name	Login/SignUp Button
Description	Enter Username and Password
Primary Factor	User
Pre Condition	User must open application
Post Condition	User can login in case of correct credentials
Frequency of Use Case	Many Times
Alternative Use Case	N/A
Use Case Diagrams	
Attachments	N/A

Table-6.3

CHAPTER-7

RESULTS

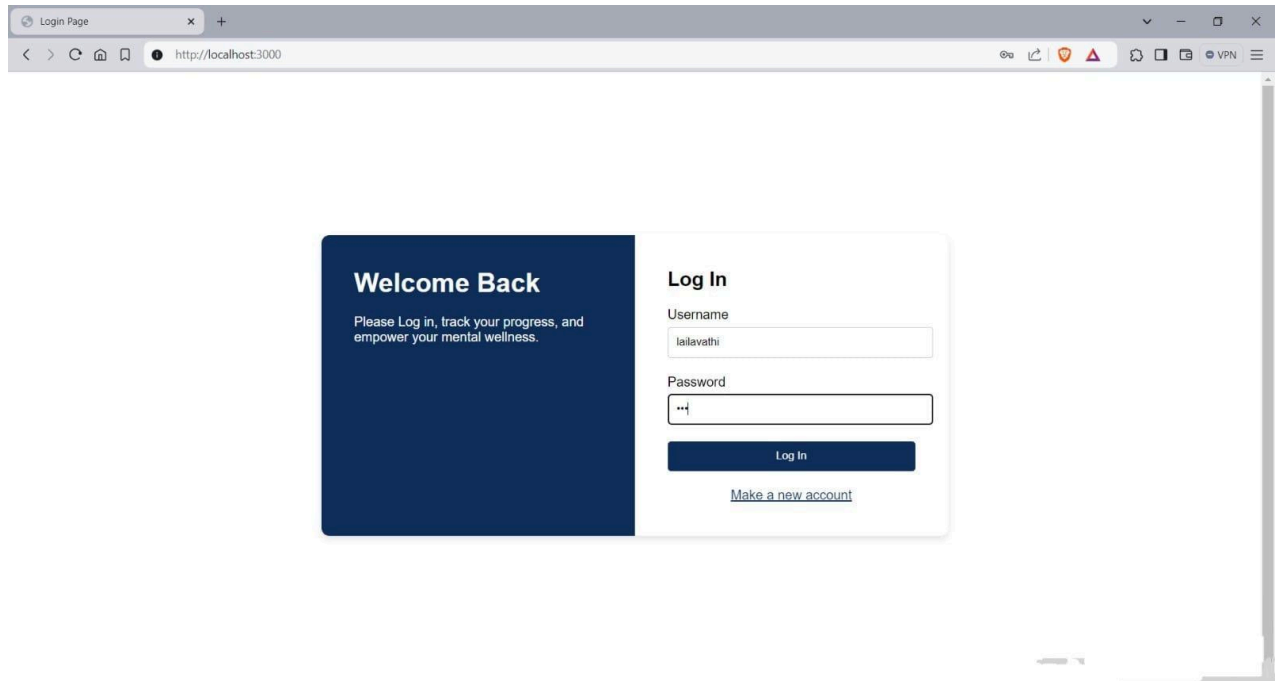


Figure-7.1 Log In page

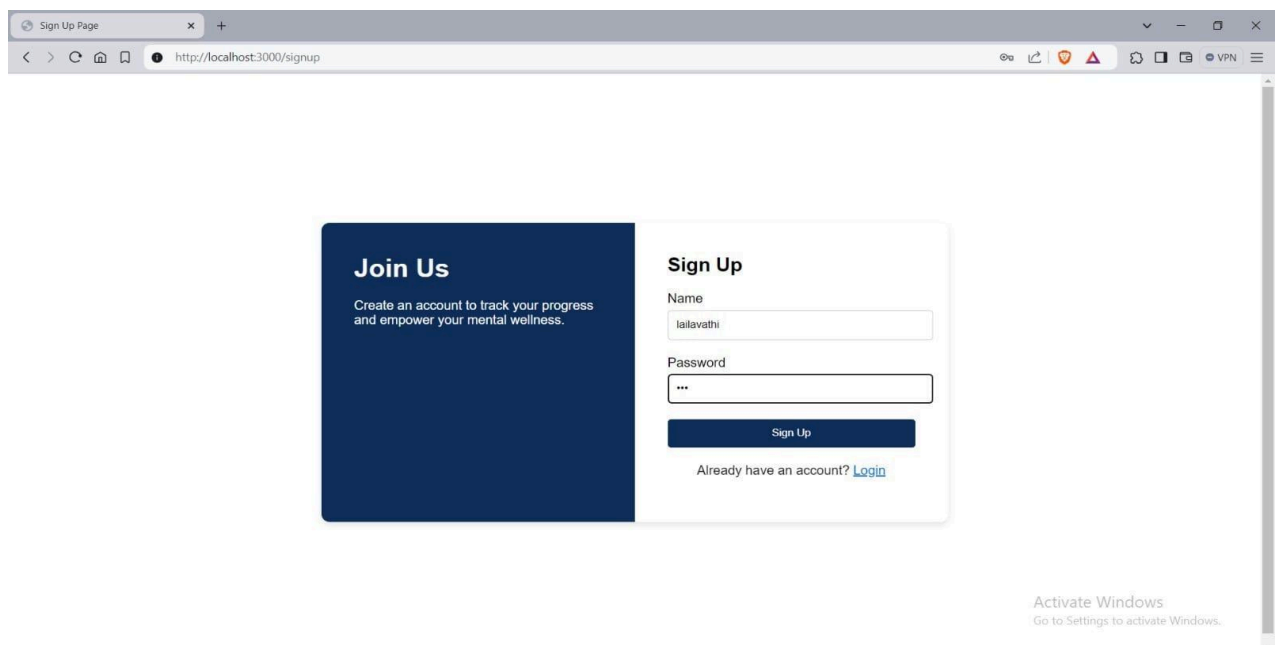


Figure-7.2 Sign up page if not a user

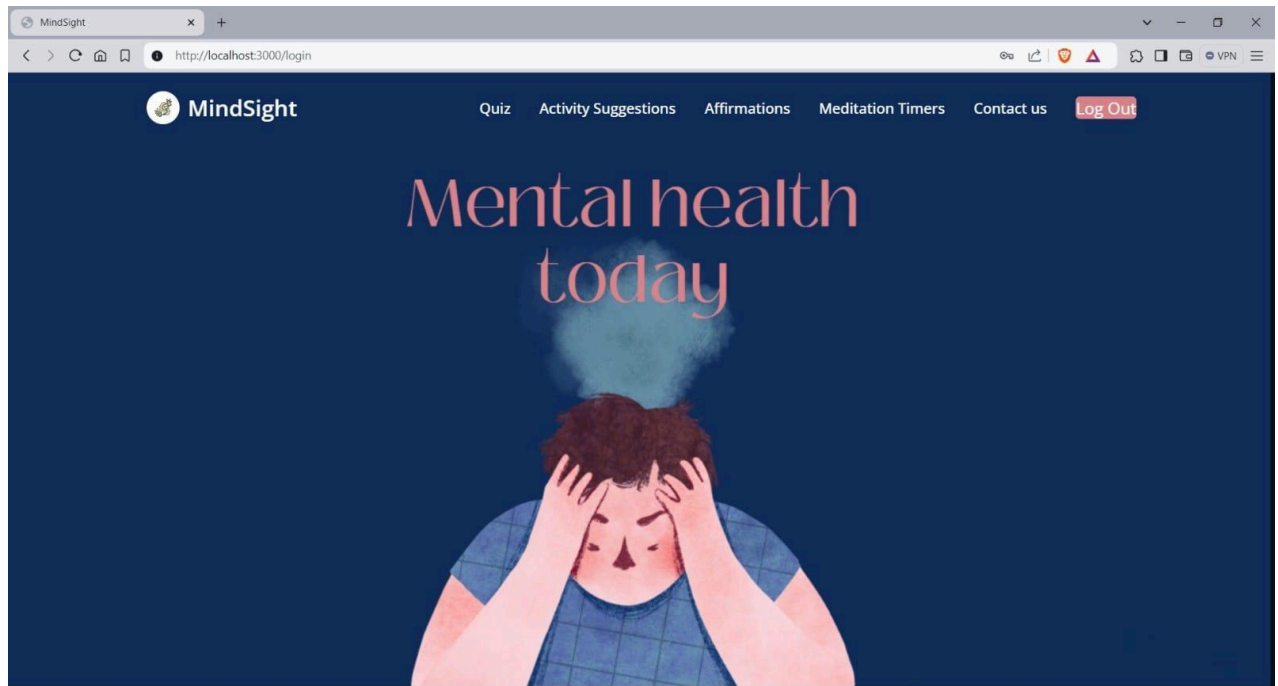


Figure-7.3 Homepage

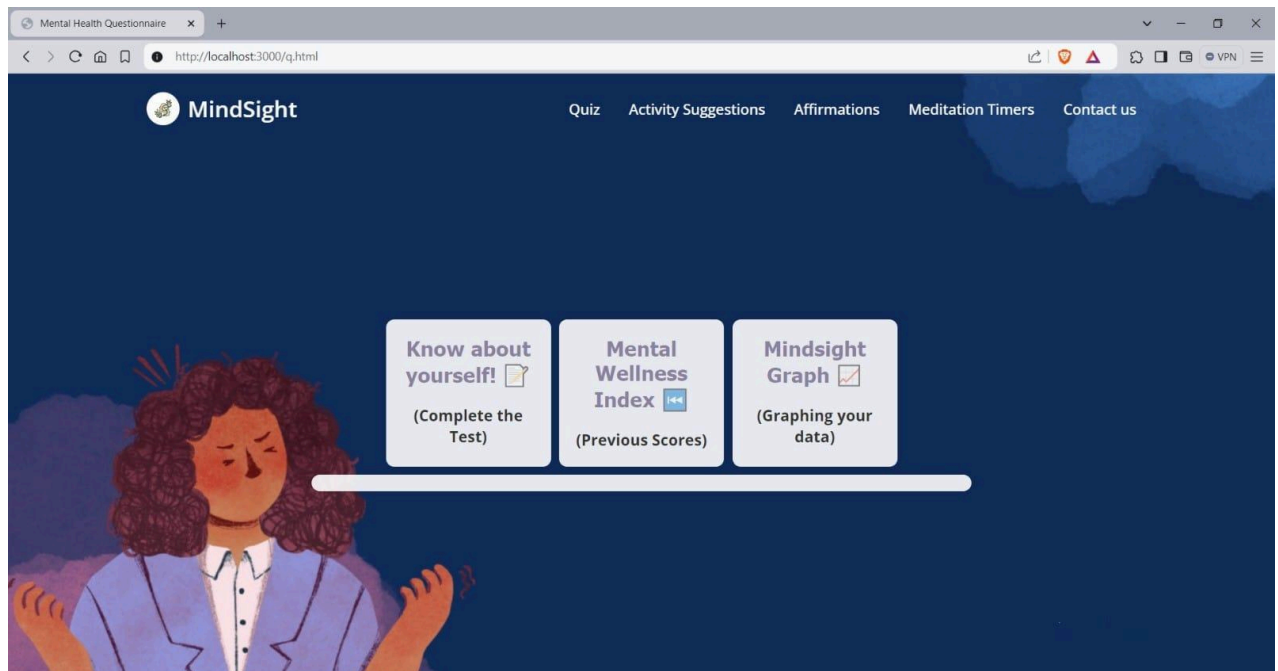


Figure 7.4 Explore the Quiz Section

MindSight Quiz Activity Suggestions Affirmations Meditation Timers Contact us

Know about yourself! (Complete the Test)

Mental Wellness Index (Previous Scores)

Mindsight Graph (Graphing your data)

Wellness Check-in

How often do you have trouble concentrating on tasks?

☐ Never
 ☐ Rarely
 ☐ Sometimes
 ☒ Often
 ☐ Always

How often do you feel like you are worried about the future?

☐ Never
 ☐ Rarely
 ☒ Sometimes
 ☐ Often
 ☐ Always

How often do you feel like you are lacking motivation?

☐ Never
 ☐ Rarely
 ☐ Sometimes
 ☐ Often
 ☒ Always

How often do you feel down, depressed, or hopeless?

☐ Never
 ☒ Rarely
 ☐ Sometimes
 ☐ Often
 ☐ Always

Figure-7.5 Take the quiz

MindSight Quiz Activity Suggestions Affirmations Meditation Timers Contact us

Know about yourself! (Complete the Test)

Mental Wellness Index (Previous Scores)

Mindsight Graph (Graphing your data)

Previous Scores

Your previous score: 38 (on 8/14/2024, 1:17:52 PM)

Your previous score: 15 (on 8/14/2024, 1:18:24 PM)

[Go Back](#)

Figure-7.6 Check previous scores

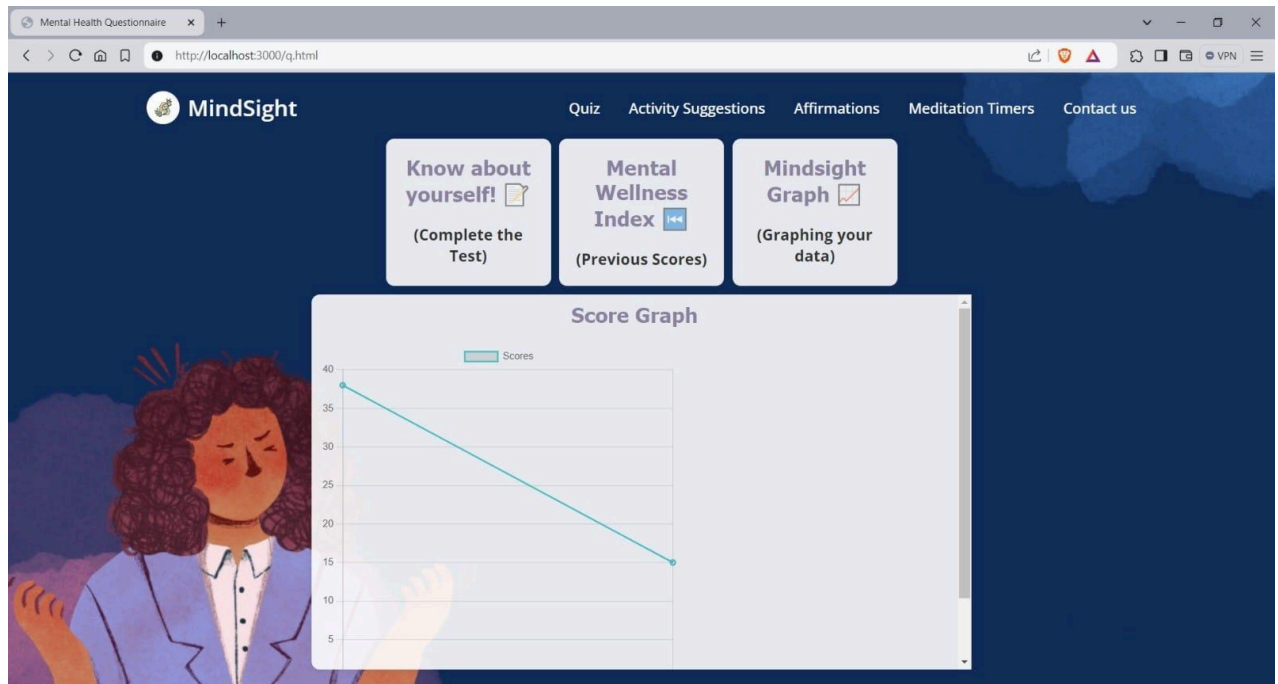


Figure-7.7 Graphing the data

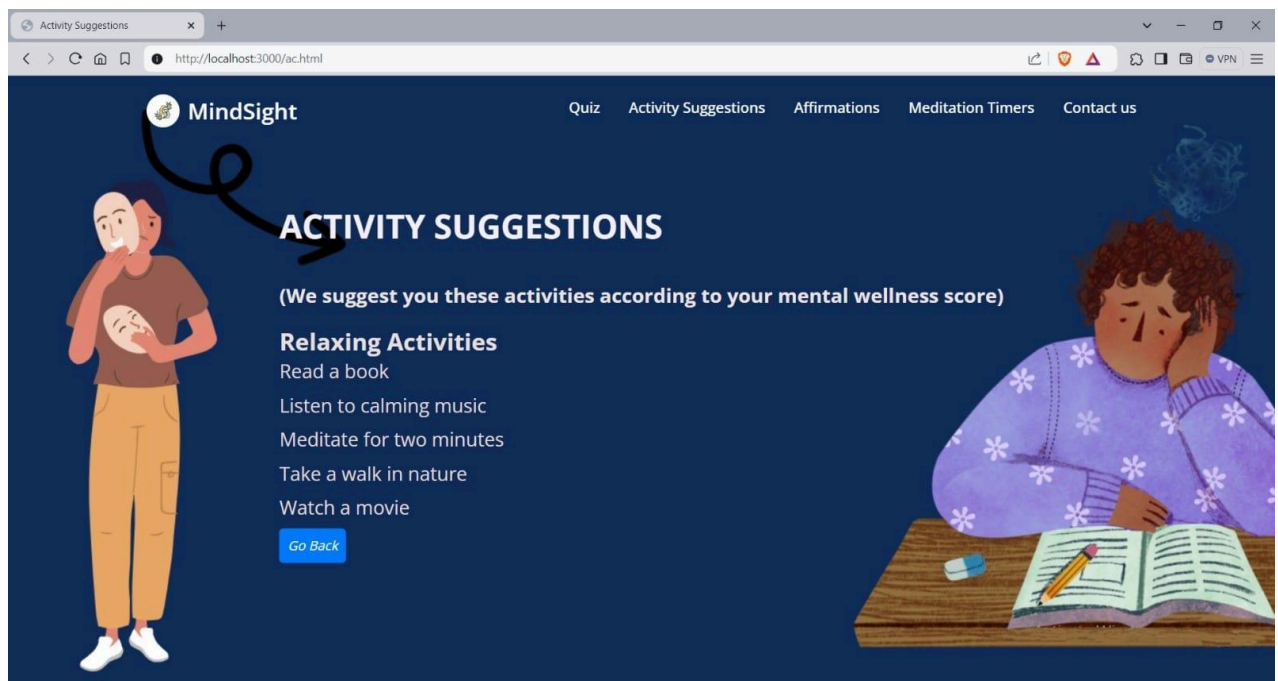


Figure-7.8 Activity Suggestions Page

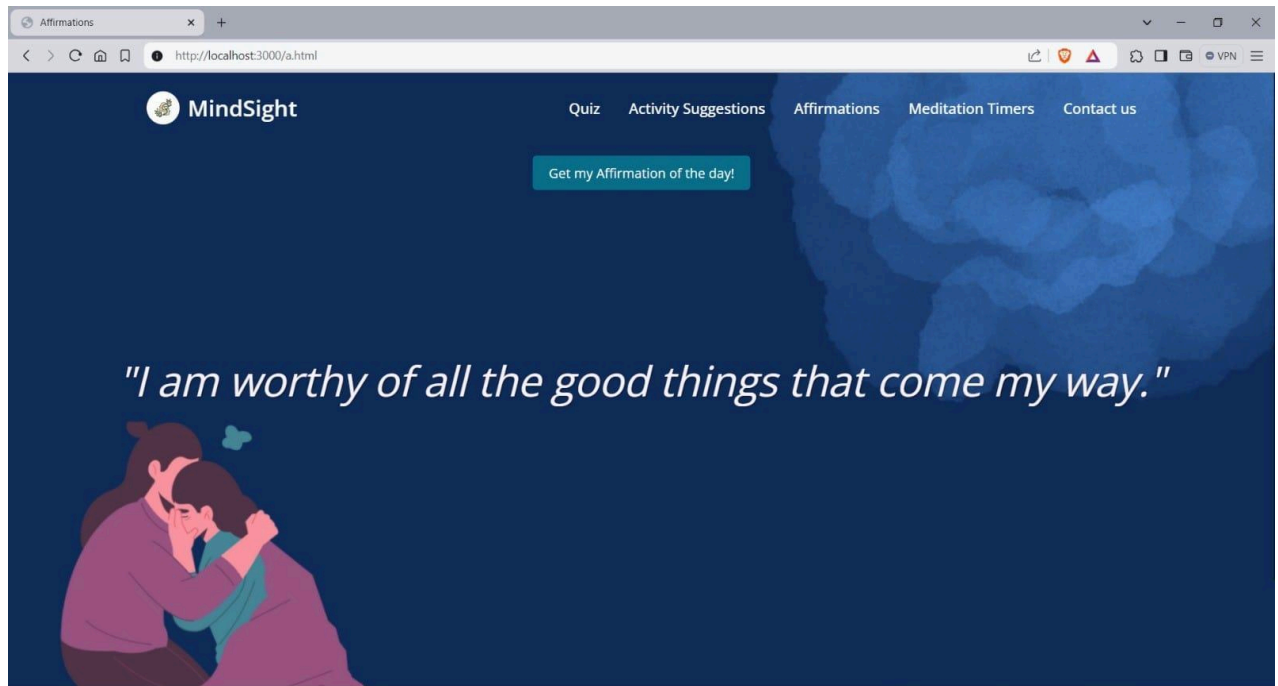


Figure-7.9 Affirmations Page

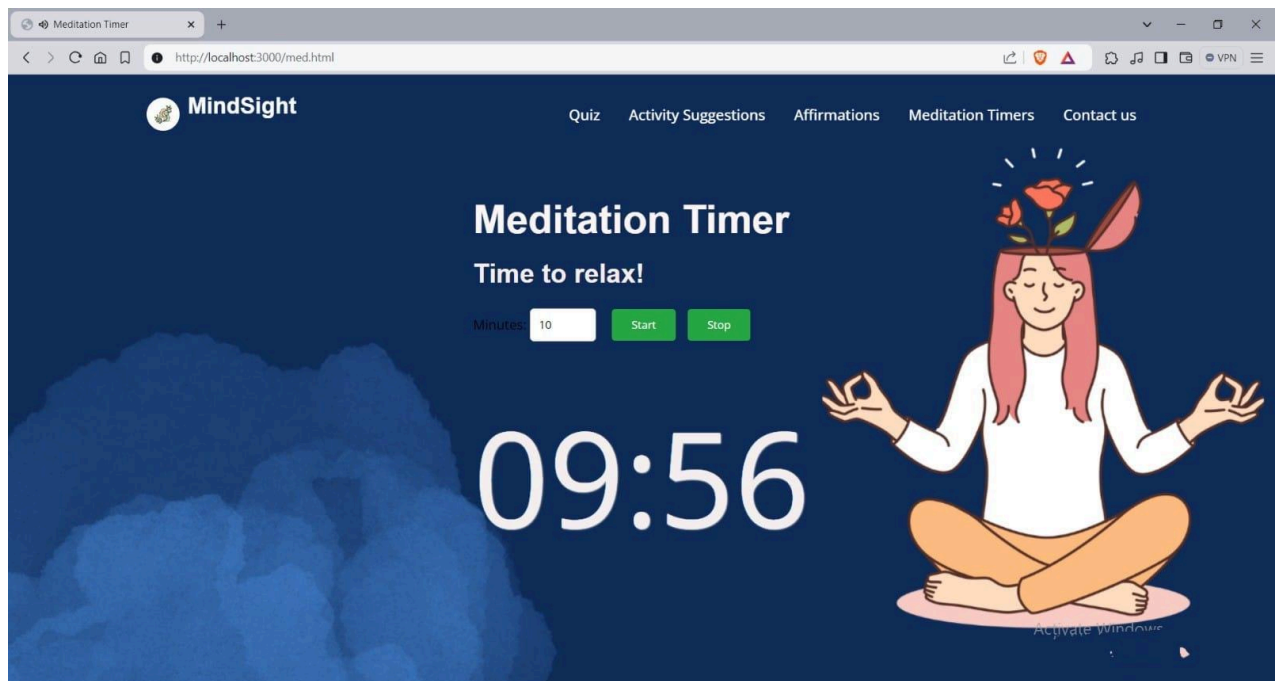


Figure-7.10 Meditation Timer

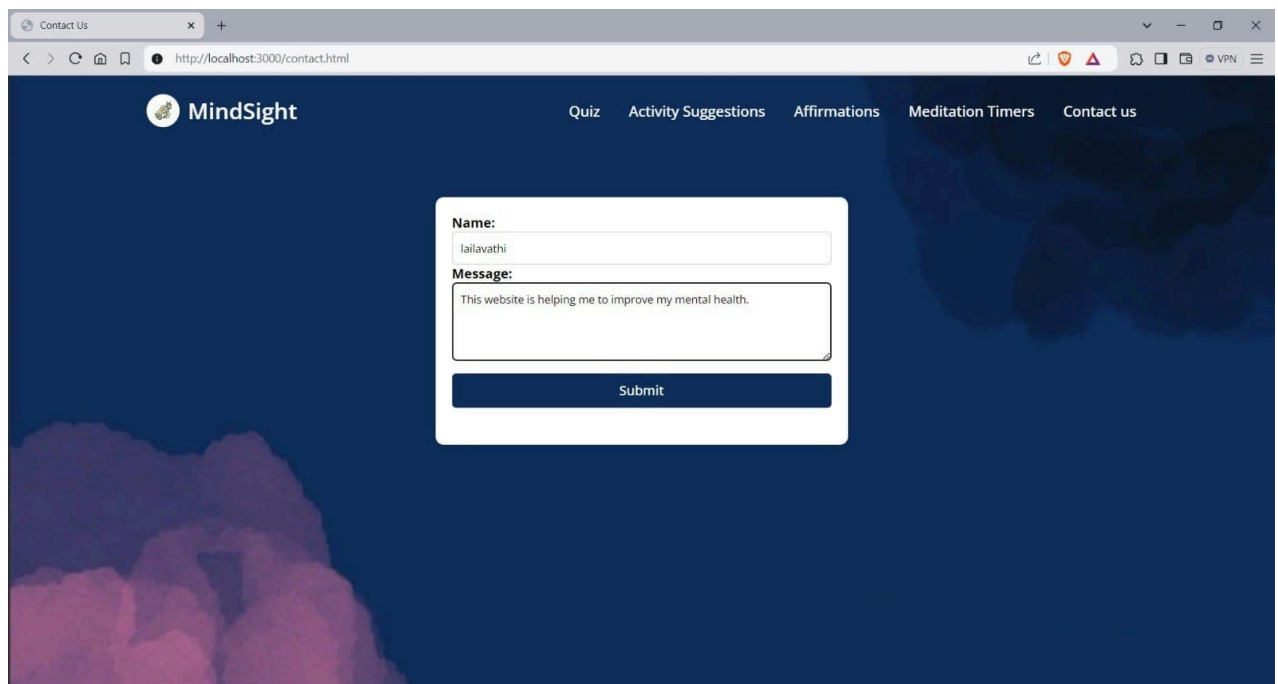


Figure-7.11 Contact us page

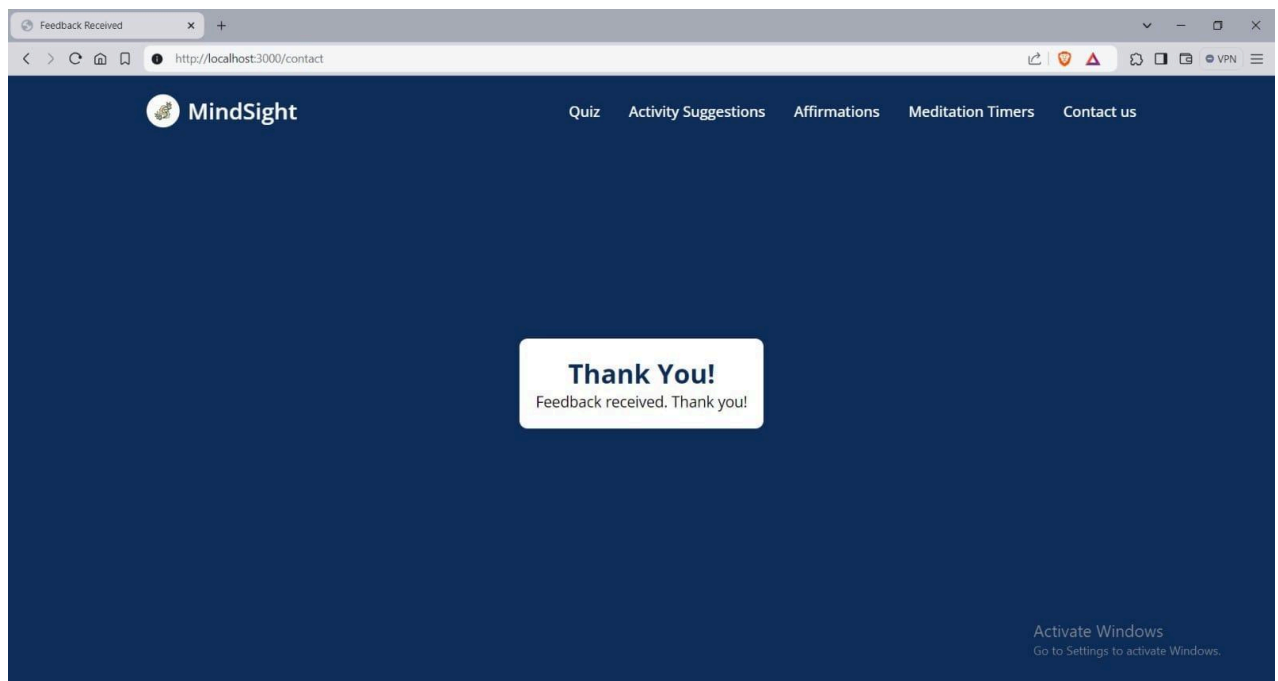


Figure-7.12 Feedback received

CHAPTER-8

8.1 CONCLUSION

The website of mental health trackers should allow forenabled users to obtain valuable information and create an all-in-one tool to be used to track and enhance their mental health and well being. These modes allow users to log in to the website, take surveys, quizzes, and fill out questionnaires to obtain information concerning their mental health status. concerning the test or survey results. Individualized activity recommendations are provided to achieve better customization of the services based on the needs of every user. Affirmation button provides users with positive statements every day. In other words, the affirmations, and the fThe affirmation exercise the intention of meditation timer assist. Users apply mindfulness and relaxation techniques. The aim is to equip users with the tools they require to monitor their psychological state development and participate in activities that they can use to further augment their mental health journey.


8.2 FUTURE SCOPE


- Personalized Recommendations: Integrate smart recommendations to recommend mental health care according to user's patterns and previous data.
- Regular Check-ins and Surveys: Employ a regime that periodically checks users and their input or alternatively conducts polls with the purpose of referencing the application's features as per the needs and expectation of the users.
- Crisis Management Tools: Integrate features, such as the buttons for emergency contacts or the direct access to the hotlines of the crisis centers.

REFERENCES

- Shreeya Mayekar, Dakshali Merya, Kamini Patil , Akshata Sonawane, "Mental Health Tracker", International Research Journal of Engineering and Technology (IRJET), Volume 10 , Issue 03, March 2023.
- Apoorva Bagul, Pooja Sinkar, Priyanka Jadhav, Deepali Ahire, Prof. D. D. Sharma, "A Mental Health Tracker Built Using Flutter and Firebase", Department of Computer Engineering ,Late G. N. Sapkal College of Engineering, Anjaneri, Nashik, Volume 7, Issue 1, January 2023.
- Mayureshwar Jadhav, Rohit Kulkarni, Bhagyashri Nage, Amit Dixit, Vishal Walunj, "Companion App: A Mental Health Tracker", International Research Journal of Modernization in Engineering Technology and Science , Volume 05, Issue 05, May 2023.

PLAGIARISM REPORT OF MINDSIGHT: Empowering Mental Wellness:

PapersOwl

[Log out](#)  [My orders](#) [My balance \\$0.00](#) [Earn 35](#) [ADD FUNDS](#) [PLACE ORDER](#) [Menu](#)

Free Online Plagiarism Checker

in recent years owing to the fact that mental health is a relatively new field of concern people have used digital technologies in different ways this project aims to address the challenge of mental health management by proposing a holistic solution in the form of a web application called mindsight empowering mental wellness all in all one can undoubtedly mention several useful aspects of this software that can be helpful for ordinary users as well as for certain clients with

182 words (1158 characters)

[Recheck this text after changes](#) [Check another text](#)

SIMILAR0.0%

ORIGINAL100.0%

Well done, your text is unique!

Need an essay written but don't have the time?
With PapersOwl you'll get it professionally researched,
written and received right on time!

GET MY ESSAY DONE

How to avoid plagiarism?

Proper citation style

Avoid plagiarism by always listing the source and formatting it correctly when you are note-taking. Take care of the proper formatting and citation style when using content from outside sources.

Write on your own


Avoid borrowing and overusing large pieces of the content from outside sources, especially from Wikipedia. Write your own thoughts and use sources only to support your opinion (remember to cite it though!).


Rewriting Service


PapersOwl expert can rewrite up to 75% of your content, edit and proofread your paper to make it plagiarism free and ready to use.

FROM \$8⁰⁰ page

PLACE ORDER

Let's Chat

PapersOwl

[Log out](#)  [My orders](#) [My balance \\$0.00](#) [Earn 35](#) [ADD FUNDS](#) [PLACE ORDER](#) [Menu](#)

Free Online Plagiarism Checker

chapter-8 81 conclusion the website of mental health trackers should allow forenable users to obtain valuable information and create an all-in-one tool to be used to track and enhance their mental health and well being these modes allow users to log in to the website take surveys quizzes and fail out questionnaires to obtain information concerning their mental health status concerning the test or survey results individualized activity recommendations are provided to achieve better

351 words (2334 characters)

[Recheck this text after changes](#) [Check another text](#)

SIMILAR0.0%

ORIGINAL100.0%

Well done, your text is unique!

Need an essay written but don't have the time?
With PapersOwl you'll get it professionally researched,
written and received right on time!

GET MY ESSAY DONE

How to avoid plagiarism?

Proper citation style

Avoid plagiarism by always listing the source and formatting it correctly when you are note-taking. Take care of the proper formatting and citation style when using content from outside sources.

Write on your own


Avoid borrowing and overusing large pieces of the content from outside sources, especially from Wikipedia. Write your own thoughts and use sources only to support your opinion (remember to cite it though!).

Editing Service

PapersOwl expert can edit up to 50% of your content, proofread and polish your paper to make it plagiarism free and ready to use.

FROM \$8⁰⁰ page

PLACE ORDER

Let's Chat

51