Project Report

on

MIND ASSESSMENT & INSIGHTS

Submitted In Partial Fulfillment of

MASTER OF COMPUTER APPLICATIONS (MCA)

Submitted by:

Abhinav Mishra
24/SCA/BCA(AI&ML)/002
Astitva Sharma
24/SCA/BCA(AI&ML)/011
Harsh Pradhan
24/SCA/BCA(AI&ML)/024

Under the Supervision of:

(Ms.Stuti)
(Ms.Priya)

School of Computer Applications
Manav Rachna International Institute of Research and Studies
(DEEMED TO BE UNIVERSITY)
Sector-43, Aravalli Hills Faridabad –
121001

Dec 2025

Declaration

I do hereby declare that this project work entitled “Mind Assessment & Insights” submitted by Our Team for the partial fulfillment of the requirement for the award of **BACHELOR OF COMPUTER APPLICATIONS** is a record of my own work. The report embodies the finding based on my study and observation and has not been submitted earlier for the award of any degree or diploma to any Institute or University.

SIGNATURE

Names :Abhinav Mishra
Harsh Pradhan
Astitva Sharma

Roll No: 24/SCA/BCA(AI&ML)/002
24/SCA/BCA(AI&ML)/024
24/SCA/BCA(AI&ML)/011

Date: 5 DEC 2025

Certificate from the Guide

This is to certify that the project report entitled "MIND ASSESSMENT & INSIGHTS" submitted in partial fulfillment of the degree of **BACHELOR OF COMPUTER APPLICATIONS** to Manav Rachna International Institute of Research and Studies, Faridabad is carried out by Abhinav Mishra, Astitva Sharma and Harsh Pradhan (Roll No), 24/SCA/BCA(AI&ML)/002,011 and 024 under my guidance.

Signature of the Guide

Name: DR. Stuti & Ms. Priya

Date:5 Dec 2025

Head of Department

Name: DR. Sohail Javed

Date: 5 DEC 2025

ACKNOWLEDGEMENT

I gratefully acknowledge for the assistance, cooperation, guidance and clarification provided by Ms./Mr. Stuti & Priya during the development of Mind Assessment. My extreme gratitude to **Our Professors** who guided us throughout the project. Without his willing disposition, spirit accommodation, frankness, timely clarification and above all faith in us, this project could not have been completed in due time. His readiness to discuss all important matters at work deserves special attention of.

I would like to extend my sincere gratitude to **Prof. (Dr.) Suhail Javed Quraishi (HOD), Prof. (Dr.) Rashmi Agrawal , (Associate Dean) and Prof. (Dr.) Brijesh Kumar (Dean)** for their valuable teachings and advice. I want to thank all the department faculty members for their cooperation and support. I want to thank non-teaching staff of the department for their cooperation and support.

This opportunity is a big milestone in my career development. I will strive to use gained skills and knowledge in the best possible way, and I will continue to work on their improvement, to attain desired career objectives. I hope to continue cooperation with all of you in the future.

INDEX

Introduction

a) About Project

The Mental Health Assessment Portal is a simple and practical web-based system designed to help students evaluate their mental well-being through a structured questionnaire. The idea behind this project is to provide an accessible and private tool that works directly on a local machine without needing advanced resources or complicated installations. The project focuses on making mental health screening easy to understand by converting user responses into a score and displaying it in the form of color-coded zones that indicate the user's general mental state.

The portal is built using widely used technologies such as Node.js, Express, SQLite, HTML, CSS, and JavaScript. It demonstrates full-stack development concepts like backend routing, database connectivity, REST APIs, frontend-backend communication, and secure password handling. This setup also helps learners understand how simple data collection and analysis systems are created and maintained.

The project environment is suitable for students because it allows them to explore backend logic, database management, and basic UI design in a clean and controlled setup. The system is stable, predictable, and ideal for academic practice or demonstrations. Since it runs locally, it provides a safe space for learning without needing deployment, hosting, or cloud resources.

b) Aims & Objectives

This project was developed as part of the semester coursework to help understand how to build a functional web application using basic development tools. The main aim was to design a system that can collect user responses, process them, calculate a meaningful score, and store the results securely.

Main objectives:

- To develop a questionnaire-based mental health assessment using HTML, CSS, and JavaScript.
- To implement backend functionality using Node.js and Express for managing users and submissions.
- To store and retrieve assessment results using an SQLite database.
- To provide an admin panel that can display all student submissions with zone-based results.
- To ensure basic data privacy through hashed passwords and protected routes.

c) Manpower

The entire project was independently developed by Abhinav Mishra (BCA AI/ML, 2024–2027), under the academic supervision of the assigned faculty. All research, coding, testing, and documentation work has been carried out individually as part of the Minor Project requirement.

System Study

a) Existing System and Limitations

In the current scenario, students who want to check their mental well-being usually rely on online self-assessment tools, scattered psychological quizzes, or general health websites. These platforms are often difficult to navigate, require account creation, or present complicated questionnaires that may not be beginner-friendly. Many existing tools are hosted on external servers, which raises concerns about privacy, especially when dealing with sensitive mental health responses.

Another problem is that students rarely get access to simple, offline-capable systems that allow them to evaluate their mental health privately. Most available assessments depend heavily on internet connectivity or professional consultation, which may not always be feasible. Beginners who want to understand the basic working of mental health assessment systems also struggle due to the lack of simple, small-scale models that demonstrate both frontend and backend logic together.

In summary, the existing system presents these limitations:
• No simple combined system offering a private, offline mental health assessment.
• Dependence on external websites that may not maintain user privacy.
• Requirement of internet connection for most online self-check tools.
• Higher complexity for beginners who want to learn how such systems work internally.

b) Proposed System and Advantages

The proposed project provides a unified, easy-to-use Mental Health Assessment Portal that runs entirely on a local machine and requires no external hosting or internet access. It is designed for both educational and personal usage, allowing students to take a mental health assessment privately while also letting developers observe how data flows through a complete system.

The system includes:

1. **Mental Health Questionnaire Module**
   A structured set of questions that evaluates stress, mood patterns, sleep quality, anxiety indicators, and overall mental state. The frontend captures responses, calculates the score, and assigns a zone based on the result.

2. **User Login and Registration**
   A simple user management system using Node.js and SQLite where passwords are securely hashed. It ensures that every user's assessment history is stored separately.

3. **Zone-Based Scoring System**
   Based on the final score, the system categorizes users into zones such as Green (Healthy), Yellow (Mild Stress), Orange (Moderate Stress), and Red (High Risk). These zones provide clear and straightforward feedback.

4. **Admin Panel**
   A dedicated admin interface that allows authorized personnel to view and evaluate submissions entered by students. It helps understand patterns and gives an overview of user results stored in the database.

Key Advantages of This System:
• Runs locally: Works offline using localhost without needing any deployment or hosting.
• Beginner-friendly: Uses basic web technologies, making it easy for students to understand.
• Privacy-focused: All data stays on the user's local device, reducing privacy concerns.
• Modular design: Each component (login, test, scoring, admin panel) is independent and easy to extend.
• Cost-free: Uses open-source tools like Node.js, SQLite, and Visual Studio Code.

Overall, the proposed system helps students understand how mental health assessment tools work while providing a functional and private self-check platform.

Feasibility Study

1. Technical Feasibility

The technical requirements of this project are minimal and easily achievable on any basic personal computer. The system uses open-source technologies such as Node.js, Express, SQLite, HTML, CSS, and JavaScript, all of which are lightweight and run efficiently on standard hardware. Since SQLite is file-based, no separate server installation is required.
Because the backend runs locally and the frontend is static, the system does not depend on external hosting, making it technically simple and reliable for academic use.

2. Operational Feasibility

The system is operationally feasible because it is designed for everyday users with no technical background. The interface is simple: students can log in, answer the questionnaire, and view results without guidance. Admins can access submissions effortlessly through a clean dashboard.
No complex procedures or training are required, which makes the system easy to operate in a college environment.

3. Economic Feasibility

The project has **zero financial cost**.
All tools used like Node.js, SQLite, Visual Studio Code, and browsers they are completely free.
No paid hosting, premium APIs, or licensed software are necessary.
Because the system runs locally and stores everything in a small database, there are no maintenance or operational expenses.
Thus, the project is completely economically feasible for students and institutions.

4. Behavioural Feasibility

The system is behaviourally feasible because users can adapt to it quickly.
The questionnaire uses simple language, making it easy for students to understand and respond accurately.
The result output is color-coded (Green, Yellow, Orange, Red), which helps students interpret their mental-health score instantly.
From a developer's perspective, the modular code structure ensures smooth learning and easy modification.

5. Legal Feasibility

Since the project is used only for academic purposes and all data is stored locally on the user’s machine, there are no legal risks.
The system does not collect medically sensitive information or share any data online.

It does not claim to diagnose or replace professional mental-health services, so it remains legally safe and compliant with educational project norms.

(Not heavily relevant to this project; kept appropriately concise.)

6. Cultural and Political Feasibility

The system is culturally and politically neutral.
It does not favor any belief, group, or ideology and strictly focuses on mental-wellness awareness.
The questionnaire avoids sensitive or controversial topics, making it safe for students from diverse backgrounds.

(Not directly relevant; summarized briefly.)

7. Schedule Feasibility

The project was planned and implemented within a well-defined academic timeframe.
Each module such as backend, frontend, questionnaire, scoring system, admin panel was completed in small, manageable phases.
The Gantt chart used during planning ensured each task stayed on track.
Overall, the schedule was realistic and achievable within the semester duration.

8. Market Feasibility

Although the project is not intended for commercial deployment, similar mental-health screening tools exist online.
This shows that such systems have practical relevance and can be useful if expanded or deployed at an institutional level.
For academic purposes, the market feasibility is low-priority but still positive.

(Partially relevant; kept moderate.)

9. Resource Feasibility

All resources needed for development software tools, libraries, documentation are freely available.
No special hardware, servers, or external infrastructure were required.
This makes the project highly resource-efficient and accessible for students

Project Monitoring System

a) Gantt Chart

Task	Start Date	End Date	Duration
Planning	10/07/25	30/07/25	20 days
Learning & Setup	01/08/25	15/08/25	15days
Backend Development	16/08/25	28/08/25	12 days
Frontend Development	29/08/25	07/09/25	10 days
Database Integration	08/09/25	20/09/25	12 days
Assessment Module	21/09/25	25/09/25	4 days
Testing & Report	26/09/25	01/10/25	5 days

System Analysis

a) Requirement Specification

A clear understanding of the hardware and software requirements is necessary to develop and run the Mental Health Assessment Portal smoothly. Since the project is lightweight and runs locally, the requirements are minimal and suitable for any student or educational setup.

Hardware Requirements

• Processor: Intel Core i3 or equivalent (or higher)
• RAM: Minimum 4 GB (8 GB recommended)
• Storage: At least 500 MB free space for project files and tools
• Display: Any modern laptop or desktop screen
• Network: Internet required only during initial installation of Node.js or for viewing external references

Software Requirements

• Visual Studio Code: Used for writing, editing, and testing the project files
• Node.js with npm: Required for backend logic, Express routes, and handling API requests
• SQLite: Lightweight database used for storing user data and assessment submissions
• Web Browser: Google Chrome, Microsoft Edge, or Firefox for viewing and testing the frontend
• Optional Tools:

- Live Server extension for serving frontend files
- Diagramming tools like draw.io for creating DFDs and ERDs

These requirements allow the system to run efficiently on a basic personal computer without the need for any professional-grade software or expensive setup. This makes the project extremely feasible for academic environments.

b) System Flowcharts

Flowcharts help illustrate how data and actions move through different parts of the system. They simplify the logic and make the workflow understandable even before the code is written.

Below are descriptive versions of the flowcharts for the Mental Health Assessment Portal:

1. **User Login Flowchart**
   • User enters username and password
   • System validates credentials through backend
   • If correct → user enters assessment dashboard
   • If incorrect → system shows error message

2. **Assessment Process Flowchart**
   • User opens questionnaire page
   • Questions are displayed one-by-one or in a list
   • User selects answers
   • User submits the assessment
   • System calculates score
   • System assigns a zone (Green/Yellow/Orange/Red)

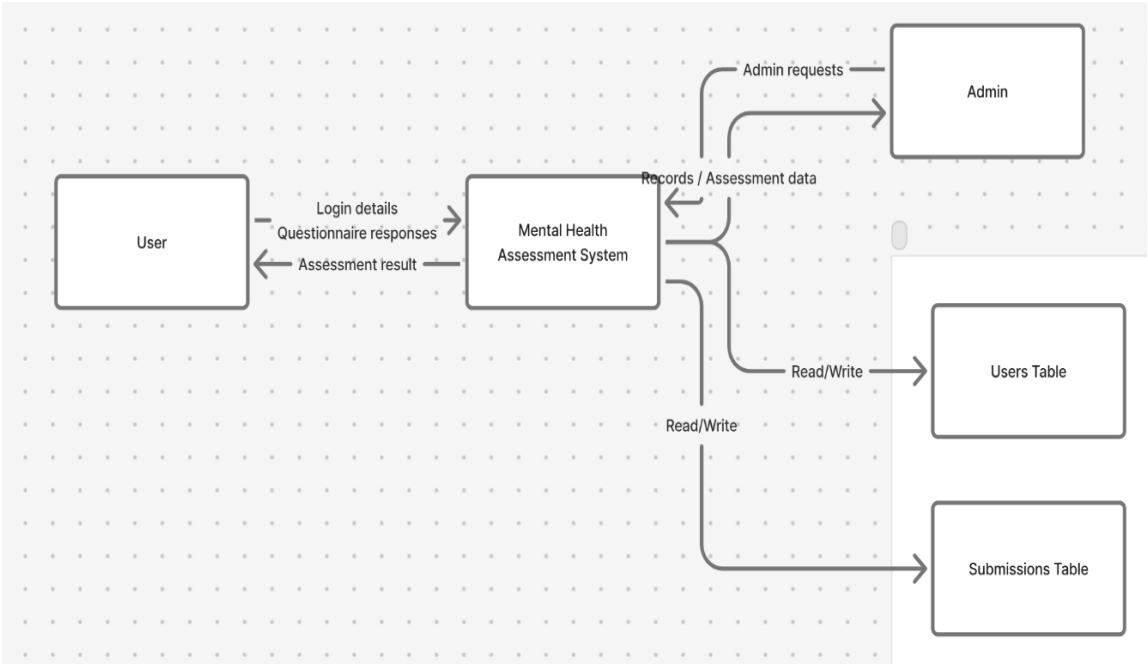
• Result is stored in the database
• Final score and zone displayed to the user

3. **Admin Panel Flowchart**
   • Admin logs in using admin credentials
   • Admin dashboard opens
   • System fetches all assessment submissions
   • Admin views list of users, scores, and zones

These flowcharts illustrate how the system handles user actions from login to result display, ensuring that each step is clear and easy to follow.

c) Data Flow Diagrams (DFDs) / ER Diagrams (ERDs)

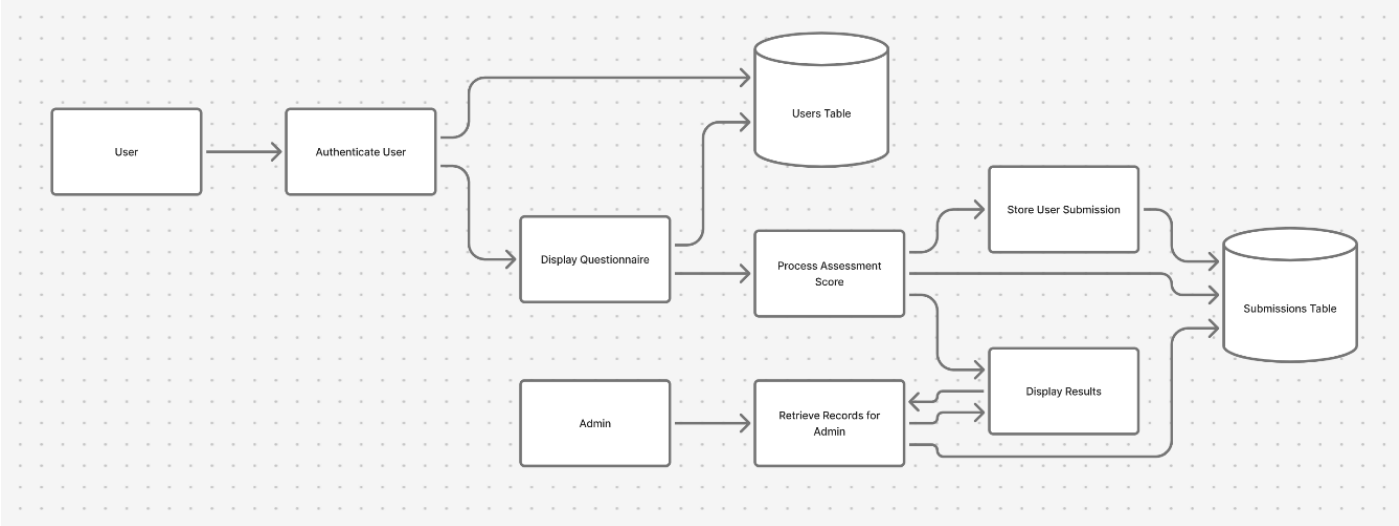
1. **Context-Level DFD**



The context-level DFD shows the entire Mental Health Assessment Portal as one main process that interacts with two external entities.
• Entities: User, Admin
• Process: Mental Health Assessment System
• Data Flows: Login details, questionnaire responses, assessment result, admin requests

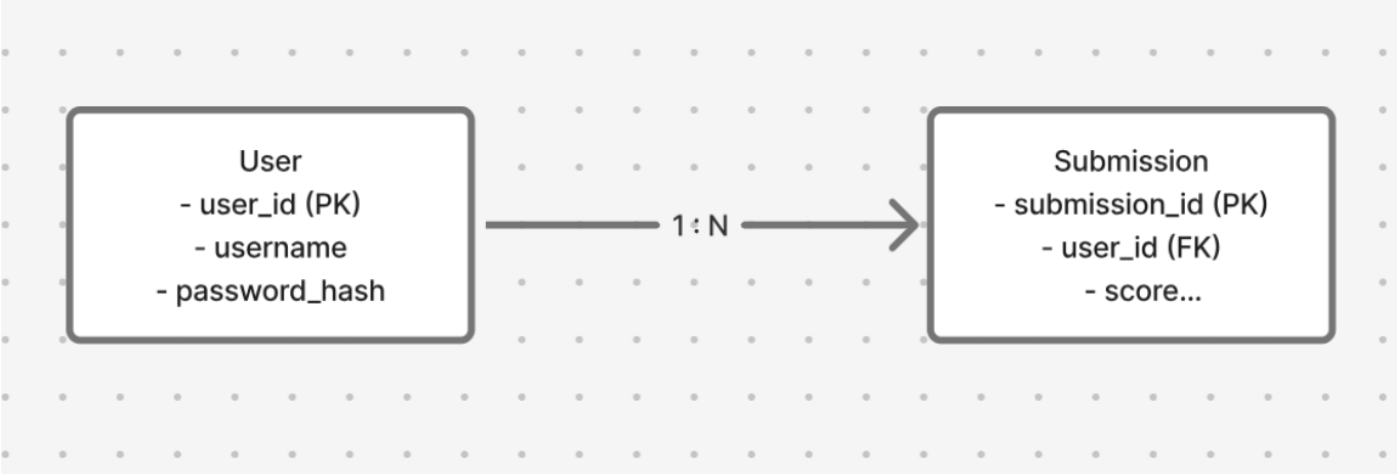
2. **Level 1 DFD**
   This diagram breaks the system into smaller processes:
   • Authenticate User
   • Display Questionnaire
   • Process Assessment Score
   • Store User Submission
   • Retrieve Records for Admin create image

• Display Results Data Stores:
• Users Table
• Submissions Table



Data Stores:
• Users Table
• Submissions Table

3.



Entity Relationship Diagram (ERD)
The ERD shows how data is structured inside the system.
Entities:
• User: Attributes – user_id, username, password_hash
• Submission: Attributes – submission_id, user_id, score, zone, date

Relationships:
• One User → Many Submissions
• Each Submission belongs to one User

System Design

a) File / Data Design

Proper file and data design ensures that the Mental Health Assessment Portal remains organized, easy to maintain, and scalable. Each module in the system is separated based on its responsibility to keep the structure clean and understandable.

1. Backend (Node.js + Express)

Files Used:

• server.js – Main backend file that initializes the Express server, defines routes, and manages API communication.
• database.js – Handles SQLite database connection and table creation.
• auth.js – Contains functions for hashing passwords and verifying login credentials.
• routes.js – Defines API endpoints for login, register, saving assessments, and fetching admin data.

Design Purpose:

The backend structure keeps authentication, routing, and data-handling logic clearly separated. This makes debugging easier and ensures that code is modular.

2. Frontend (HTML, CSS, JavaScript)

Files Used:

• index.html – Login page for both student and admin users.
• register.html – Page for creating new accounts.
• assessment.html – Displays mental health questionnaire using HTML and JavaScript.
• result.html – Shows score and mental health zone.
• admin.html – Admin dashboard to view submissions.
• style.css – Common stylesheet for layout and color-coded zone results.
• script.js – Contains logic for sending data to backend, processing answers, and redirecting pages.

Design Purpose:

The frontend is kept lightweight, separating structure (HTML), styling (CSS), and logic (JavaScript). This enhances readability and supports easy updates.

3. Database (SQLite)

Tables Used:

• users – Stores user details (username, password hash, role).
• submissions – Stores assessment results (user_id, score, zone, date, answers).

Design Purpose:

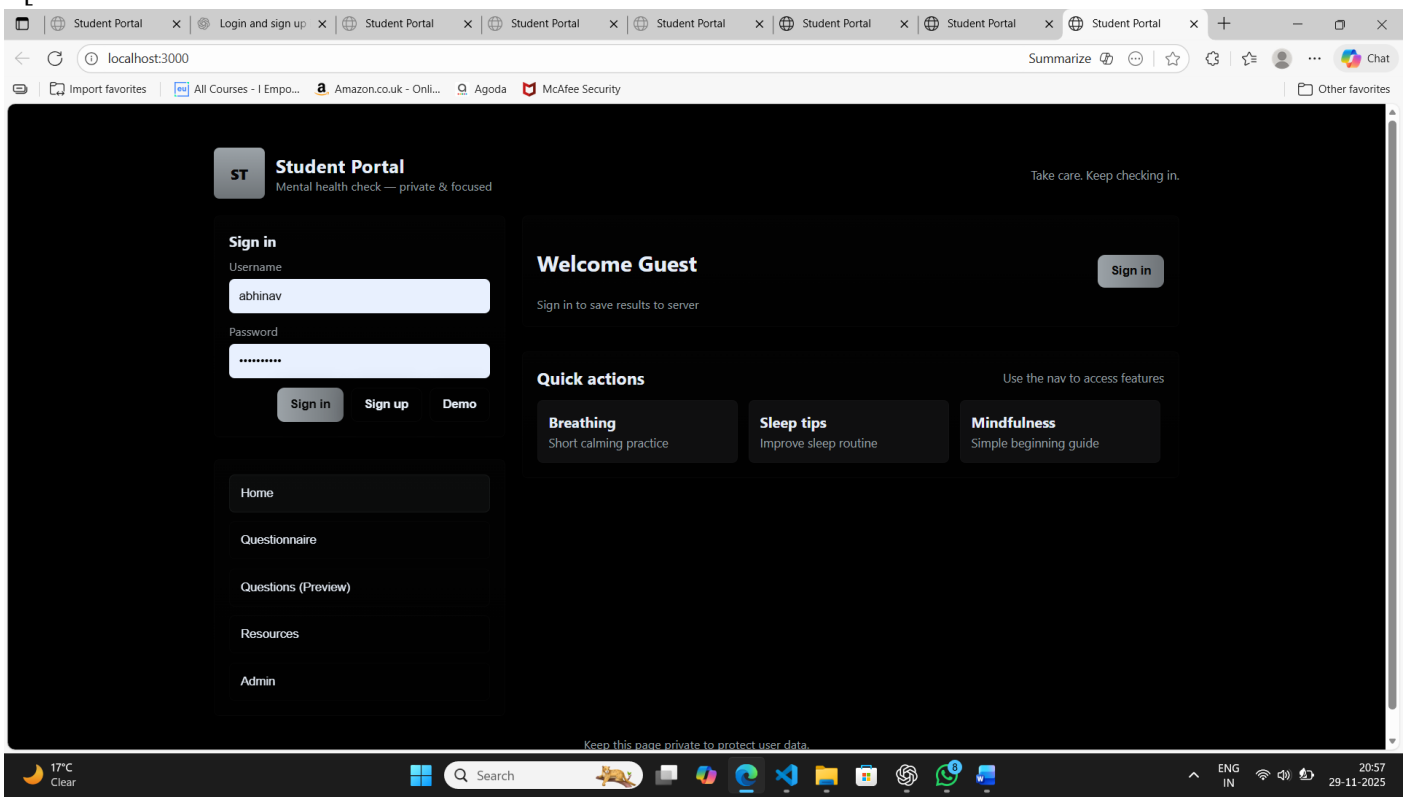
The database maintains a simple relational structure, ensuring efficient storage of user data and submissions.

Input / Output Form Design

a) Screen Design (Screenshots)

Screenshots are essential to demonstrate the working of the Mental Health Assessment Portal. Below are placeholders where you can insert screenshots in the final document:

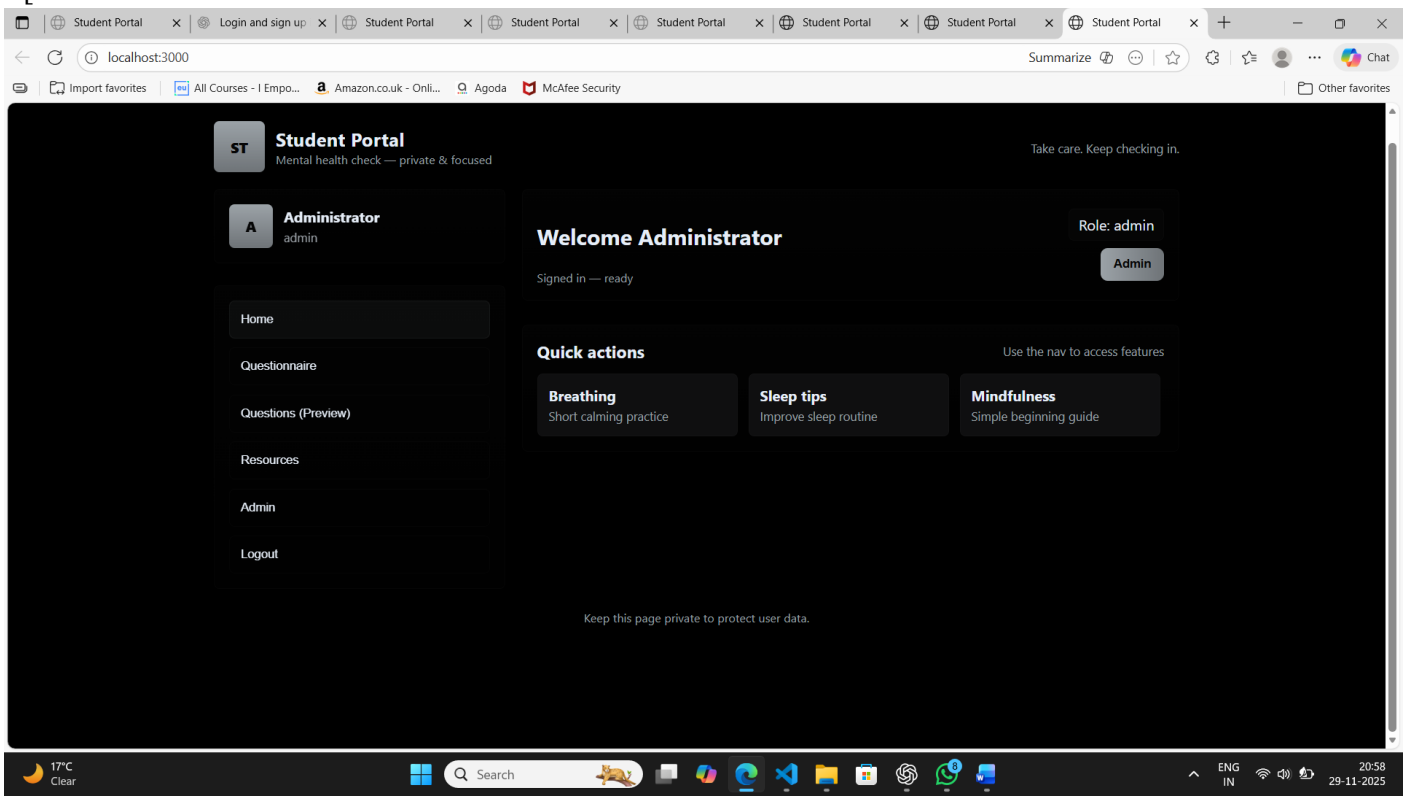
• [



]

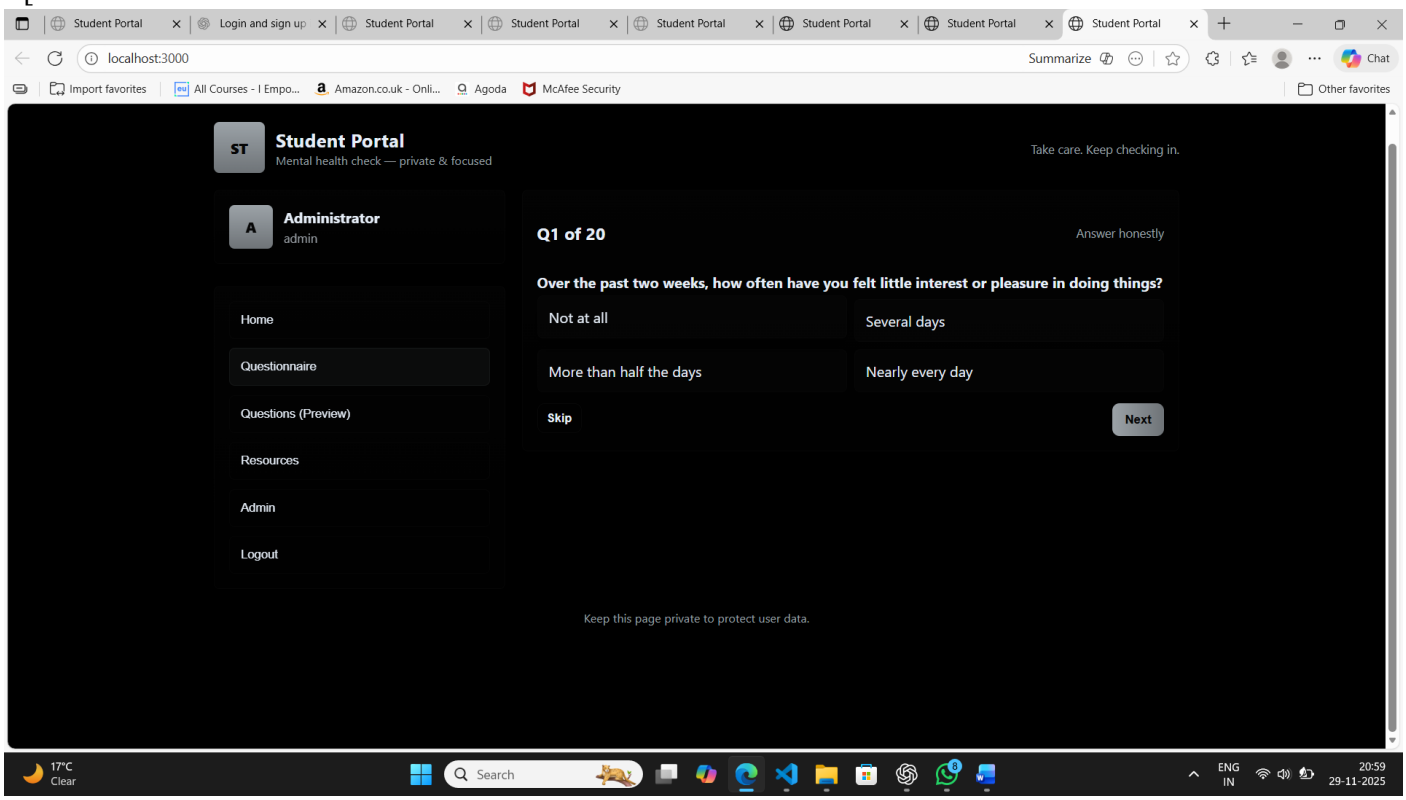
(Shows username, password, and login button)

• [



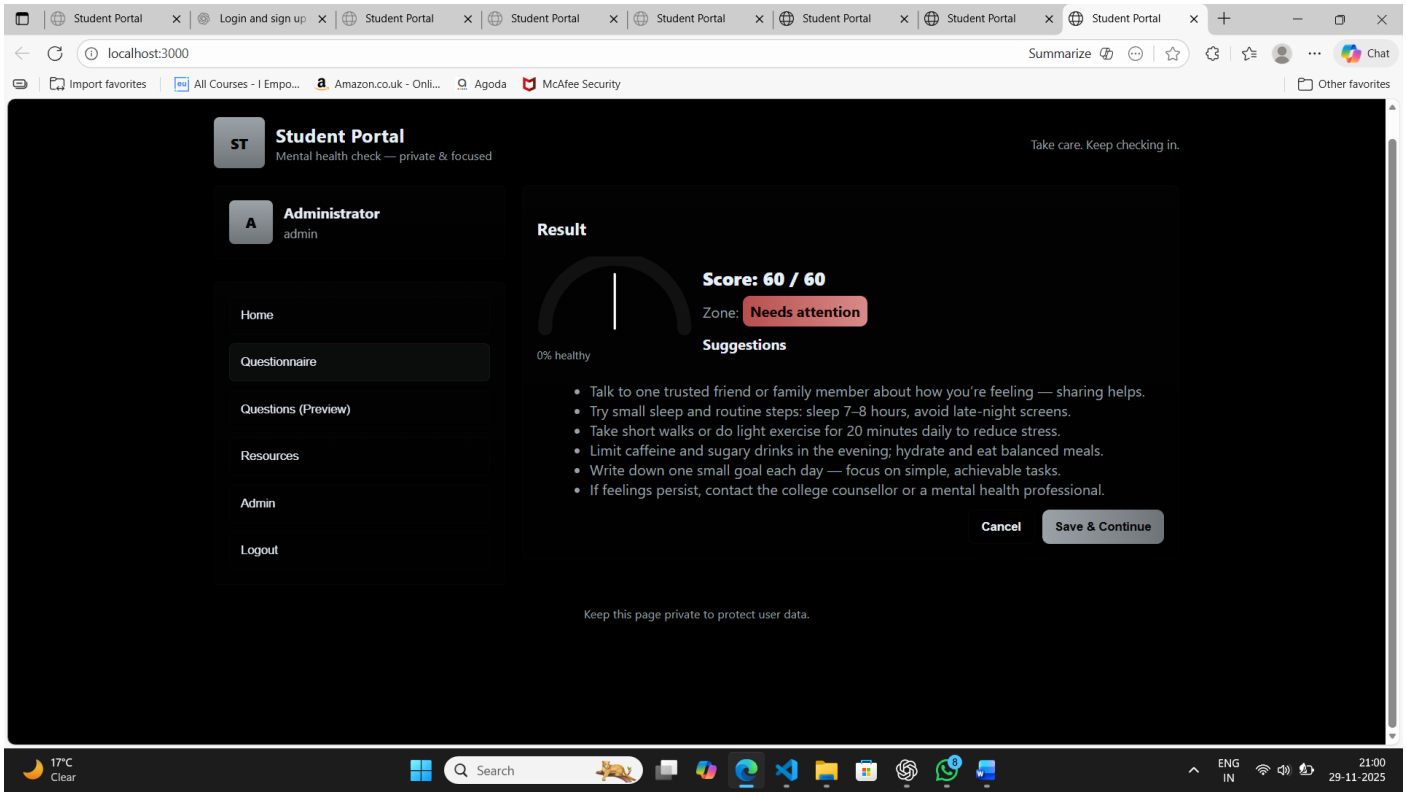
]
(Displays fields for username, password, and role selection)

• [



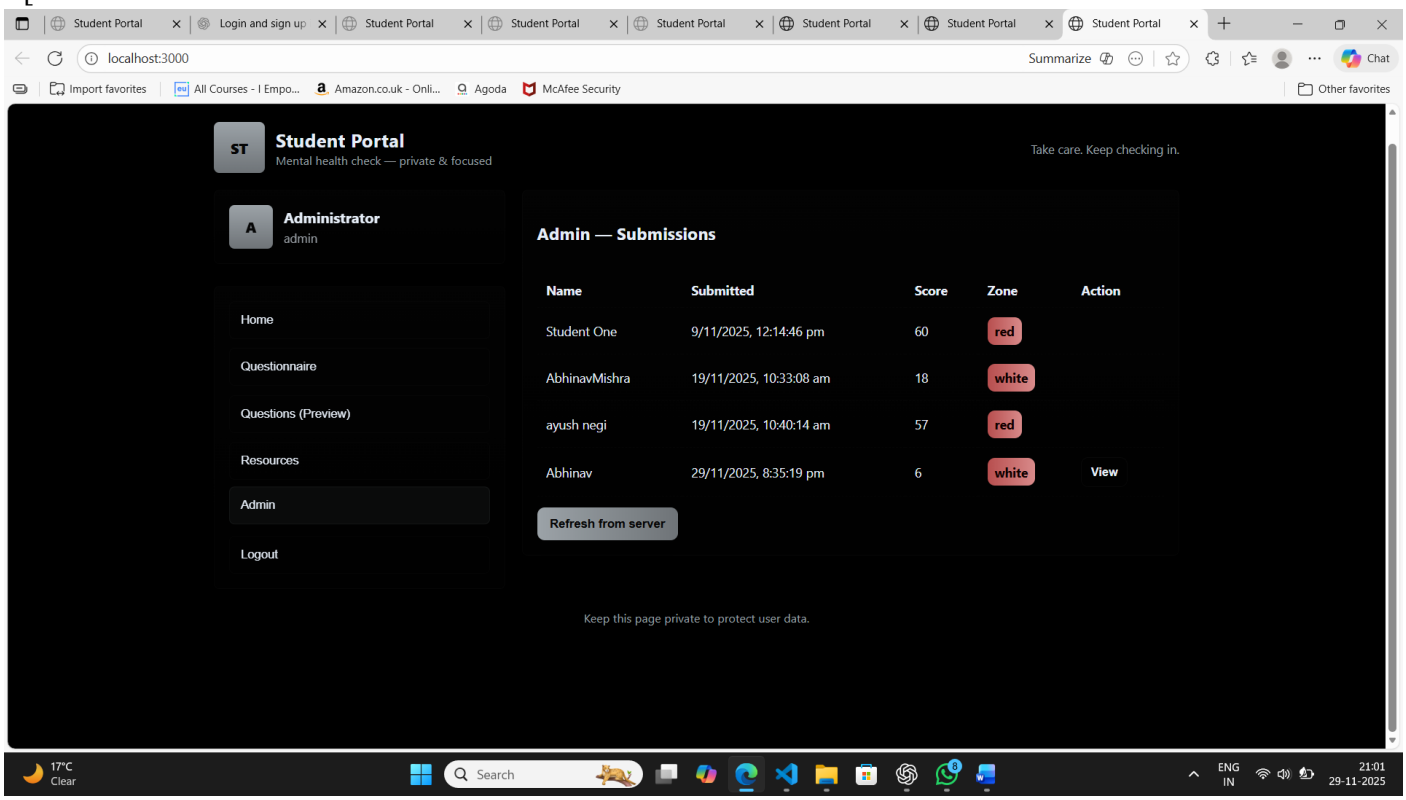
]
(Shows questionnaire with multiple-choice answers)

•



]
(Displays score, color-coded zone, and suggestions)

• [



]
(Shows table listing user submissions with date and zone)

b) Report Design

The report design refers to how the system displays outputs during interaction:

Assessment Output:
• Score based on total selected options
• Zone classification:

- Green (Healthy)
- Yellow (Mild Stress)
- Orange (Moderate Stress)
- Red (High Stress)
  • Summary message explaining the student's mental state

Admin Panel Output:
• Table showing all users' names, scores, zones, and submission dates
• Data fetched directly from SQLite
• Easy to analyze patterns and performance of users

These outputs make the system informative and user-friendly, fulfilling the project's goal of providing a clean and interactive assessment tool.

System Testing

a) Preparation of Test Data

Testing the Mental Health Assessment Portal required preparing different sets of inputs to simulate how real users would interact with the system. The test data was created to cover a range of situations including correct login attempts, incorrect entries, and various combinations of assessment responses.

Test data included:
• Usernames and passwords (both valid and invalid) to check authentication behaviour.
• Different combinations of questionnaire answers such as low-stress, medium-stress, and high-stress patterns.
• Edge-case data such as leaving questions unanswered or rapid submissions.
• Simulated admin queries to verify that stored submissions display correctly.

This carefully prepared data ensured that each feature of the system worked properly and behaved consistently under different conditions.

b) Testing With Live Data

The system was tested in a live environment using a local machine setup. The backend was launched through Node.js, and all frontend pages were opened using a browser or Live Server. Each module was tested repeatedly to confirm stability and accuracy.

Steps Taken:
• Launched server.js through the terminal to start the backend.
• Logged in as both student and admin to verify role-based access.
• Registered new users to confirm that password hashing and database insertion worked.
• Filled out the complete assessment using different answer combinations to verify scoring accuracy.
• Checked whether the result page correctly displayed the zone and score.
• Logged into the admin panel to confirm that all stored submissions were visible in a structured table.
• Tested on multiple browsers (Chrome, Edge) to ensure compatibility.

Live testing confirmed that the system operated without errors and responded correctly under normal usage.

c) Test Cases with Results

The table below summarizes various test cases performed during system testing

Test Case	Expected Result	Actual Result	Status
Login with correct details	User logged in successfully	Logged in	Pass
Login with wrong details	Error message shown	Error displayed	Pass
Register new user	User added to database	Registered successfully Pass	
Submit assessment	Score stored and zone displayed	Stored + zone shown	Pass

Score calculation	Output matches selected answers Correct score shown Pass	Rightly calculated	pass
Zone display	Color-coded zone appears	Correct zone displayed	Pass
Admin view records	Stored submissions visible	Records displayed	Pass
Database connection	Data saved without errors	Connection stable	Pass

All test cases were successfully passed during testing. Minor formatting bugs (if any) were corrected during development, and the final system was verified to be stable and responsive under normal usage.

System Implementation

a) System Requirements

A proper implementation of the Mental Health Assessment Portal requires basic hardware and software resources. Since the system is lightweight and runs on localhost, the requirements are minimal and suitable for any standard personal computer used by students.

Hardware Requirements

• Processor: Intel Core i3 or equivalent (minimum)
• RAM: 4 GB (8 GB recommended)
• Disk Space: Around 1 GB free space for Node.js, SQLite, and project files
• Internet: Required only for installation of tools or external references
• Display: Any modern laptop or desktop screen

Software Requirements

• Visual Studio Code: Used to create, modify, and test all project files
• Node.js with npm: Required for backend development and running Express routes
• SQLite: Used as the database to store user accounts and assessment submissions
• Web Browser: Chrome, Edge, or Firefox to open frontend pages
• Live Server extension (optional): Used to preview and auto-reload frontend files during development

These tools are open-source and freely available, allowing the project to be completed without any financial investment.

How to Run the Application

Each component of the project is designed to run smoothly on a local machine without any deployment steps. Below are the instructions for running the Mental Health Assessment Portal:

1. Backend (Node.js Server)

• Open the terminal in the backend folder.
• Run the command:

node server.js

• The backend server starts and listens on:

http://localhost:3000

2. Frontend (Assessment Portal UI)

• Open the frontend folder in Visual Studio Code.
• Right-click on index.html and select “Open with Live Server,” or open it manually in your browser.
• The frontend loads at:

http://127.0.0.1:5500

3. Using the System

• Users can register a new account or log in using existing credentials.
• After logging in, the assessment page loads and displays the questionnaire.
• When users submit the form, the backend calculates the score and returns a color-coded zone.

• The result page displays the final score and mental health zone.
• Submissions are stored in SQLite for future reference.

4. Admin Panel
• Log in with admin credentials.
• The admin dashboard displays all user submissions, including name, date, score, and zone.
• Admin can use this data for analysis or evaluation.

This simple implementation process ensures that anyone with basic technical knowledge can run and test the system without difficulty.

Documentation

The Mental Health Assessment Portal consists of several modules, each designed with specific functionality and a clear learning objective. Screenshots for each module can be added in Section XII to visually demonstrate the working of the system. Below is a detailed, module-wise technical explanation rewritten for your project.

1. User Authentication (Login & Registration Module)

Functionality:
This module manages user access by allowing individuals to register and log into the system. Registered users can take assessments, while admin users can view stored submissions.

Technology Stack:
Node.js, Express.js, SQLite, bcrypt, HTML, CSS, JavaScript

Files Used:
• index.html – Login interface
• register.html – Registration page
• server.js – Processes login and register API calls
• database.js – Stores user data securely
• auth.js – Handles hashing and password verification

Key Concepts Used:
Password hashing, backend validation, protected routes

Output:
A successfully logged-in user is redirected to the assessment page, while invalid users receive an appropriate error message.

Screenshot:
[Insert Login/Register Screenshot]

2. Mental Health Assessment Module

Functionality:
This is the core part of the system where students answer a set of mental health-related questions. After submission, the system calculates a score and assigns one of the color-coded mental health zones.

Technology Stack:
HTML, CSS, JavaScript (frontend)
Node.js and Express.js (backend)

Files Used:
• assessment.html – Displays questionnaire
• script.js – Handles scoring logic and API submission

• server.js – Stores results in SQLite
• submissions table – Stores user responses and score

Key Concepts Used:
Form handling, scoring algorithms, frontend-backend communication

Output:
The user sees:
• Total score
• Assigned mental health zone (Green, Yellow, Orange, Red)

Screenshot:
[Insert Assessment Screenshot]

3. Result Display Module

Functionality:
After completing the assessment, users are shown a clear and simple results page. The zone is displayed using color coding to make interpretation easy.

Technology Stack:
HTML, CSS, JavaScript

Files Used:
• result.html – Shows the output score and zone
• script.js – Retrieves and displays backend response

Key Concepts Used:
Dynamic content rendering, quick feedback system

Output:
Shows:
• Score
• Zone
• A short explanation for the user

Screenshot:
[Insert Result Page Screenshot]

4. Admin Panel

Functionality:
This module allows the admin to view all assessment submissions. It displays stored data like usernames,_scores, zones, and timestamps in a tabular format.

Technology Stack:
Node.js, Express.js, SQLite, HTML, JavaScript

Files Used:
• admin.html – Admin interface

• server.js – Fetches submissions via API
• database.js – Retrieves data from SQLite

Key Concepts Used:
Admin authorization, data retrieval, table display

Output:
A list of all users and their mental health scores, stored results, and zones.

Screenshot:
[Insert Admin Panel Screenshot]

5. Database Module

Functionality:
All user credentials and test submissions are stored securely within an SQLite database.

Tables Used:
• users – For storing username, password hash, and role
• submissions – For storing score, zone, answers, and timestamps

Key Concepts Used:
SQL queries, relational mapping, secure storage, table creation

Output:
Stable and reliable record of all assessment entries.

Scope of the Project

The scope of the Mental Health Assessment Portal defines what functionalities are included in the current version of the system and what features are intentionally excluded. The purpose of this project is to create a fully functional, easy-to-understand mental health evaluation tool suitable for academic learning and practical demonstration.

In Scope (Included):

• Development of a working mental health assessment system.
• Implementation of user login and registration using Node.js and SQLite.
• Creation of a structured questionnaire for mental health evaluation.
• Score calculation and classification into color-coded zones.
• Storing user assessment submissions in the SQLite database.
• Admin panel to view all submissions in a tabular format.
• Local testing and demonstration through a browser and terminal.
• Use of core web technologies: HTML, CSS, JavaScript, Node.js, Express, and SQLite.

Out of Scope (Not Included):

• Deployment to live servers or cloud platforms (e.g., Heroku, AWS, Vercel).
• Integration of AI-based analysis or predictive mental health models.
• Real-time chat or communication features between user and counselor.
• Use of advanced frontend frameworks like React, Angular, or Vue.js.
• Integration with medical professionals or certified diagnosis tools.
• User session tracking, logout scheduling, or multi-role permissions beyond admin and student.
• Long-term data analytics or graphical dashboards.

The defined scope ensures the project stays simple, understandable, and manageable for students. It covers essential full-stack concepts without overwhelming complexity, making it appropriate for a Minor Project.

Bibliography

Below are the online resources and references used for researching, learning, and developing the Mental Health Assessment Portal. These sources helped in understanding backend development, frontend structure, scoring logic, and database connectivity.

1. **MDN Web Docs – Mozilla Developer Network**
   Reference for understanding HTML, CSS, and JavaScript concepts, form handling, DOM operations, and web standards.
2. **Node.js Official Documentation**
   Used to learn backend development, server creation, HTTP methods, and Express fundamentals.
3. **Express.js Documentation**
   Primary reference for building routes, middleware structure, API endpoints, and backend logic.
4. **SQLite Documentation**
   Guided the creation of tables, database queries, and data storage techniques used in the users and submissions tables.
5. **bcrypt Library Documentation**
   Used for understanding how to hash passwords securely before storing them.
6. **W3Schools**
   Referred for basic frontend examples, CSS styling techniques, and form input structures.
7. **YouTube Tutorials**
   Used for understanding Node.js basics, routing methods, and integration of frontend with backend.
8. **GeeksforGeeks**
   Helpful for JavaScript logic building, understanding algorithms, and solving common development issues.
9. **FreeCodeCamp**
   Referred for conceptual clarity on HTTP requests, APIs, and beginner-level full-stack development workflows.

These references supported the development of the system and ensured that the Mental Health Assessment Portal adhered to correct programming practices and standard web development methods.