A

PROJECT REPORT

on

FitLife : A Encyclopedia

Submitted in partial fulfillment of the Requirements

For the Degree of

Bachelor of Technology

In

Information Technology (NBA Accredited)

Submitted By

Vaibhav(2100330130035)
Kuldeep(2100330130015)
Nakul (2000330130032

Under the Supervision of

Mr. Sandeep Kumar

Raj Kumar Goel Institute of Technology

GHAZIABAD, UTTAR PRADESH

AFFILIATED TO DR. A.P.J. ABDUL KALAM TECHNICAL UNIVERSITY

LUCKNOW, UTTAR PRADESH

(SESSION: May 2025)

DECLARATION

We hereby declare that this submission is our own work and that, to the best of our knowledge and belief, it contains no material previously published or written by another person nor material which to a substantial extent has been accepted for the award of any other degree of the university or other institute of higher learning, except where due acknowledgment has been made in the text**.**

Signature:

Name: Vaibhav Tyagi

Roll No: 2100330130035

Date:

Signature:

Name: Kuldeep Lodhi

Roll No: 21003300130015

Date:

Signature:

Name: Nakul

Roll No: 2000330130032

Date:

Project Guide name: Mr. Sandeep Kumar

CERTIFICATE

This is to certify that Project Report entitled "**Fit Life : A Encylopedia**" which is submitted by **Vaibhav,Kuldeep Lodhi, Nakul** in partial fulfillment of the requirement for the award of degree B.Tech. in Department of **Information Technology** of Dr. A.P.J. Abdul Kalam Technical University, Lucknow, is a record of the candidate own work carried out by him under my supervision. The matter embodied in this thesis is original and has not been submitted for the award of any other degree**.**

Date: **Supervisor**
Signature

ACKNOWLEDGEMENT

It gives us a great sense of pleasure to present the report of the B. Tech Project undertaken during B.Tech. (VIII- Semester) Final Year. We owe special debt and gratitude to **Mr Sandeep Kumar** for her constant support and guidance throughout the course of our work. His sincerity, thoroughness and perseverance have been a constant source of inspiration for us. It is only his cognizant efforts that our endeavors have seen light of the day.

We also take the opportunity to acknowledge the contribution of

Dr. Shalini Goel (Head of Department) for her full support and assistance during the development of the project.

We also do not like to miss the opportunity to acknowledge the contribution of all faculty members of the department for their kind assistance and cooperation during the development of our project. Last but not the least, we acknowledge our friends for their contribution in the completion of the project.

Signature:

Name: Vaibhav Tyagi

Roll No: 2100330130035

Date:

Signature:

Name: Kuldeep Lodhi

Roll No: 2100330130015

Date:

Signature:

Name: Nakul

Roll No: 2000330130032

Date:

Project Guide name: Mr Sandeep Kumar

ABSTRACT

This project addresses the fragmentation in digital health tools by developing an integrated health encyclopedia application that unifies disease information, symptom-based prediction, peer communication, and fitness tracking into a single platform. While existing solutions like symptom checkers, fitness apps, and medical encyclopedias operate in isolation, this application provides a seamless experience for users seeking proactive and comprehensive health management.

Key features include disease search by name, prediction of possible conditions based on up to four symptoms, real-time user chat for community support, and detailed fitness logging with filtering, analysis, and CSV export options. The system is built using the MERN stack for scalability and modularity.

A FFNN model implemented via TensorFlow.js enables in-browser disease prediction, reducing latency and enhancing user privacy. Image uploads are handled using Multer and securely stored in Cloudinary, ensuring efficient media management without server overhead.

The methodology follows a user-centric, iterative development process with extensive testing to ensure functionality, responsiveness, and accuracy. Literature and market research emphasize the growing need for integrated, AI-driven, and privacy-respecting health solutions—needs this platform directly addresses.

By combining artificial intelligence, cloud infrastructure, and modern web technologies, the application demonstrates a novel approach to digital healthcare. It also lays the groundwork for future enhancements, including wearable device integration, encrypted telehealth features, and blockchain-based health record management.

This project ultimately empowers users to manage their health more effectively by offering a unified, intelligent, and user-friendly platform.

TABLE OF CONTENTS

.

TITLE **PAGE NO.**

REFERENCES

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

CSV	Comma-Separated Values
NHS	National Health Service
MVC	Model-View-Controller
API	Application Programming Interface
FFNN	Feed Forward Neural Network
AI	Artificial Intelligence
MERN	Mongodb ExpressJs ReactJs NodeJs
IOMT	Internet of Medical Things
UI	User Interface
UX	User Experience
HIPAA	Health Insurance Portability and Accountability Act
CI/CD	Continuous Integration/Deployment
GDPR	General Data Protection Regulation
DFD	Data flow Diagram

CHAPTER - 1

Introduction

The intersection of technology and healthcare has transformed the way individuals access, understand, and manage their health. With the proliferation of smartphones and the internet, people now expect instant access to reliable medical information, tools for self-diagnosis, and platforms for tracking their fitness journeys. However, despite this technological progress, most existing solutions remain fragmented-disease encyclopedias, symptom checkers, fitness trackers, and social support networks typically exist as separate entities. This fragmentation often leads to a disjointed user experience, where individuals must juggle multiple apps and sources to manage their health comprehensively.

This project aims to address this gap by developing an integrated encyclopedia application that empowers users to:

- Search for detailed information about diseases by name,
- Predict possible diseases by entering up to four symptoms,
- Engage in chat support with other users,
- Map and monitor their fitness activities (including activity type, duration, date, and calories burned),
- Analyze their workout data using various filters (week, 5 weeks, 52 weeks, all time),
- And download their fitness logs in CSV format.

By unifying these features into a single platform, the application seeks to provide a one-stop solution for health information, disease prediction, peer support, and fitness management.

1.1 Introduction of the problem

In recent years, the demand for accessible, accurate, and actionable health information has surged. People frequently turn to the internet for answers to their health-related questions, whether they are experiencing symptoms, seeking information about a diagnosis, or looking for advice on maintaining a healthy lifestyle. Unfortunately, the overwhelming volume of information-often of varying quality-can make it challenging for users to find trustworthy and relevant content. Furthermore, most health encyclopedias and symptom checkers are limited in

scope: they may provide information about diseases or allow basic symptom searches, but rarely offer personalized predictions or connect users with others facing similar health concerns.

Simultaneously, the importance of physical fitness and activity tracking has become widely recognized as a cornerstone of preventive healthcare. Numerous apps and wearables help users log their workouts, monitor calories burned, and track progress over time. However, these fitness tools typically operate in isolation from medical information platforms, making it difficult for users to correlate their fitness activities with their overall health status or potential medical risks.

The lack of integration between disease information, predictive tools, social support, and fitness tracking creates a fragmented experience for users. For example, a user experiencing symptoms may consult an online encyclopedia for information, use a separate symptom checker for self-diagnosis, join a forum for peer support, and log their workouts in a fitness app-all without any connection between these activities. This siloed approach not only wastes time but may also lead to missed opportunities for holistic health management.

Recognizing these challenges, this project was conceived to create a unified platform that addresses the following key needs:

- **Centralized Disease Information:** Allowing users to easily search for and access reliable details about a wide range of diseases.
- **Symptom-Based Disease Prediction:** Enabling users to input up to four symptoms and receive a list of potential diseases, thus supporting early detection and informed decision-making.
- **Peer Support via Chat:** Fostering a community where users can share experiences, ask questions, and offer support to one another.
- **Comprehensive Fitness Mapping:** Providing robust tools for users to log, filter, analyze, and export their workout data, encouraging sustained engagement in physical activity.

1.2 Summarize previous Research

The development of digital health platforms has been the focus of extensive research and innovation over the past decade. Several key trends and findings from the literature and market analysis have shaped the direction of this project:

1.2.1 Disease Encyclopedias and Symptom Checkers:

Online medical encyclopedias such as WebMD, Mayo Clinic, and MedlinePlus have become

go-to resources for millions seeking information about diseases, symptoms, and treatments [1]. These platforms offer vast databases of medical knowledge, often curated by experts. However, their interactivity is limited, and they rarely provide personalized predictions based on user-entered symptoms. Symptom checker tools, such as those provided by Ada, Babylon Health, and the NHS, leverage algorithms and, in some cases, artificial intelligence to suggest possible conditions based on user input [2]. While these tools offer valuable guidance, they often restrict the number of symptoms that can be entered, may lack transparency in their reasoning, and typically do not integrate with other aspects of health management, such as fitness tracking or peer support.

1.2.2. Fitness Tracking Applications:

Fitness apps like MyFitnessPal, Google Fit, and Strava have revolutionized personal health management by enabling users to log workouts, monitor progress, and set goals [3]. These platforms often include features for tracking calories, analyzing performance over time, and exporting data. However, they focus primarily on physical activity and nutrition, with little or no integration with medical information or symptom analysis.

1.2.3. Social Support and Community Features:

Research shows that social support is a significant motivator for sustained engagement in health and fitness activities [4]. Platforms like PatientsLikeMe and various online forums provide spaces for users to share experiences and advice. However, these communities are often separate from the tools used for disease information and fitness tracking.

1.2.4. The Need for Integration:

Studies and user feedback consistently highlight the desire for integrated solutions that bring together medical information, self-diagnosis tools, fitness tracking, and community support [5]. Such integration can enhance user engagement, promote better health outcomes, and provide a more seamless and satisfying user experience.Despite these advances, no single platform has successfully combined all these elements in a coherent, user-friendly manner. This project seeks to fill that gap by building a comprehensive encyclopedia application that addresses the limitations of existing solutions and meets the evolving needs of users.

1.3. Researching the Problem

To design a solution that effectively addresses these challenges, an extensive review of current literature, existing applications, and user needs was conducted. Academic journals, reputable

health websites, and technology reports were examined to understand the strengths and limitations of current disease encyclopedias, symptom checkers, and fitness trackers.
Key findings from this research include:

- Users value accurate, easily accessible health information but often find it difficult to interpret symptoms without professional guidance.
- Community support features, such as chat forums, enhance user engagement and provide emotional support, which is especially important for individuals managing chronic conditions.
- Fitness tracking is most effective when users can analyze their data over various periods and export it for personal records or sharing with healthcare professionals.
- There is a growing demand for platforms that protect user privacy while offering personalized recommendations and insights.

Based on these insights, the project was conceptualized to deliver an all-in-one encyclopedia application. The platform was designed to allow users to:

- Search for and learn about diseases by name,
- Predict potential diseases by entering up to four symptoms,
- Engage in chat support with other users,
- Log detailed workout information (name, activity, duration, date, calories burned),
- Filter and analyze fitness data across different timeframes,
- And download workout logs in CSV format for further use.

This integrated approach not only addresses the limitations of existing solutions but also aligns with the evolving expectations of users seeking proactive and holistic health management.

CHAPTER - 2

Literature Survey

2.1 Digital Health Transformation and Patient Empowerment

The digital transformation of healthcare has fundamentally shifted the landscape of patient care and self-management. E-health technologies, including mobile applications and online platforms, are increasingly utilized to provide resource-efficient, patient-oriented care [1].

These technologies empower patients to participate actively in their health management, facilitating informed decision-making and promoting a shift toward the “health service consumer” model. As patients seek greater control over their health, there is a growing demand for personalized, convenient, and immediate access to health information and services.

Digital health solutions are thus expected to prioritize user experience, personalization, and integration of multiple functionalities to meet evolving patient expectations.

2.2 Wearable Technology and Fitness Tracking Integration

Wearable devices and fitness tracking applications have emerged as pivotal tools in monitoring and improving health outcomes. Recent studies highlight the effectiveness of behavior change techniques, such as goal setting, when combined with wearable feedback.

For instance, Yang et al. demonstrated that participants using wearables to set and monitor personalized activity goals experienced significant increases in physical activity levels over a 12-week period compared to controls [4].

This underscores the potential of wearables not just for tracking, but also for motivating and sustaining healthier behaviors.

The integration of wearable data with fitness tracking apps enables comprehensive monitoring of health metrics, allowing users and healthcare providers to identify trends, assess the impact of lifestyle changes, and receive tailored guidance.

Such integration has been shown to improve users’ health metrics significantly, with some studies reporting up to an 80% enhancement in health indicators within a month of use. Wearable technology also facilitates real-time tracking of patient and provider movements within healthcare facilities, enhancing care delivery and patient management.

2.3 Predictive Analytics and Health Outcome Prediction

A growing body of literature explores the use of wearable technology data in predictive models for clinical outcomes. A systematic review identified a limited but promising set of studies where wearable data were incorporated into models predicting mortality, hospital readmission, and emergency department visits.

For example, Bae et al. used 89 features from Fitbit data to predict hospital readmission with 88.3% accuracy, outperforming traditional models based solely on retrospective clinical data [3]. These findings suggest that leveraging the full spectrum of wearable data can enhance the accuracy and utility of predictive health models, although further research with larger and more diverse populations is needed to validate these approaches

2.4 Privacy and Data Sharing Considerations

While the benefits of integrating fitness and health data are substantial, privacy and security concerns remain significant barriers. Studies indicate that sharing fitness data with healthcare providers can lead to better health outcomes and more informed recommendations.

However, the risk of data breaches is high, with over 93% of healthcare institutions experiencing security incidents in recent years [6]. Users' willingness to share fitness information is influenced by the level of control they have over their data, emphasizing the need for granular privacy management in digital health platforms.

2.5 The Need for Integrated, User-Centric Solutions

Despite the advancements in digital health, wearable technology, and predictive analytics, most solutions remain siloed, offering either disease information, symptom prediction, or fitness tracking, but rarely all in one platform. Research consistently points to the value of integrated systems that combine these functionalities, providing users with a holistic view of their health and facilitating proactive management [5].

Such platforms can enhance user engagement, improve health outcomes, and address the growing expectations of patients as active participants in their healthcare journey.

In summary, the literature underscores the promise of integrating disease encyclopedias, symptom-based prediction, chat support, and fitness mapping into a unified application. This approach aligns with current trends in digital health transformation, leverages the motivational power of wearables and behavior change techniques, and addresses the critical need for privacy, personalization, and user empowerment in modern healthcare solutions.

CHAPTER - 3

Methodology and Technology

3.1 Methodology

The methodology adopted for developing the integrated encyclopedia application is grounded in a user-centric, modular, and iterative design approach. The project aims to provide a comprehensive platform combining disease information retrieval, symptom-based disease prediction, user communication, and fitness activity tracking. The methodology encompasses the following key phases:

3.1.1 Requirement Gathering and Analysis:

The initial phase involved identifying the functional and non-functional requirements through literature review, user surveys, and stakeholder consultations. The primary requirements included enabling users to search diseases by name, predict diseases based on symptoms (up to four), facilitate real-time chat support among users, and provide detailed fitness tracking with filtering and export capabilities.

3.1.2 System Architecture Design:

The system architecture was designed following the MVC paradigm to ensure modularity, maintainability, and scalability. The architecture separates the user interface (frontend), business logic and API layer (backend), and data storage (database). This separation facilitates independent development and testing of components and allows for future enhancements[13].

3.1.3 Data Modeling:

Data models were designed to represent users, diseases, symptoms, fitness activities, and chat messages. MongoDB, a NoSQL database, was selected for its flexibility in handling heterogeneous data and scalability. The schemas were optimized for efficient querying, especially for symptom-based disease prediction and fitness data filtering.

3.1.4 AI Model Development for Disease Prediction:

A core component of the system is the symptom-based disease prediction engine. This

was implemented using a FFNN trained with TensorFlow.js [2]. The choice of TensorFlow.js enables training and inference directly in the browser or server-side JavaScript environment, providing real-time responsiveness and reducing server load.

The FFNN model was trained on a curated dataset comprising symptom-disease mappings. The model architecture includes an input layer corresponding to symptom features, multiple hidden layers with nonlinear activation functions, and an output layer representing disease classes. The training process involved optimizing the model to minimize classification error, ensuring high prediction accuracy.

3.1.5 Development of RESTful APIs:

Backend APIs were developed using Express.js to handle requests related to disease lookup, symptom input for prediction, user authentication, chat messaging, and fitness data management. The APIs follow REST principles, ensuring stateless communication and scalability.

3.1.5 Frontend Development:

The frontend was developed using React.js to deliver a dynamic, responsive, and intuitive user interface. React's component-based architecture facilitates reusability and efficient state management, enabling features such as symptom input forms, disease information display, chat interfaces, and fitness dashboards.

3.1.6 Image Upload and Management:

User profile image uploading was implemented using Multer, an Express middleware for handling multipart/form-data. Uploaded images are stored securely on Cloudinary, a cloud-based image management service, which provides optimized delivery, scalability, and secure access. This approach offloads storage and bandwidth demands from the application server.

3.1.7 Testing and Validation:

The system underwent rigorous testing, including unit tests for individual modules, integration tests for API endpoints, and user acceptance testing to validate usability and functionality. Performance testing ensured the AI model's responsiveness and the application's scalability under concurrent user loads.

3.1.8 Deployment:

The complete application was deployed on a cloud platform to ensure high availability, scalability, and ease of maintenance. Continuous integration and deployment pipelines were configured to facilitate ongoing updates and improvements.

3.2 Technology Stack

The project leverages modern web development technologies and AI frameworks to deliver a robust and scalable solution:

3.2.1 MongoDB:

A document-oriented NoSQL database was chosen for its schema flexibility and ability to efficiently store complex, nested data structures such as user profiles, disease information, and fitness records. MongoDB's indexing and aggregation capabilities support fast retrieval and filtering operations critical for symptom-based predictions and workout analytics.

3.2.2 Express.js:

Serving as the backend framework, Express.js provides a lightweight, flexible environment for developing RESTful APIs. Its middleware architecture allows seamless integration of functionalities such as authentication, request parsing, and file uploads (via Multer).

3.2.3 React.js:

React's declarative UI paradigm enables the development of a highly interactive frontend. Components encapsulate UI elements and logic, facilitating maintainability and scalability. React's virtual DOM optimizes rendering performance, enhancing user experience.

3.2.4 Node.js:

Node.js powers the backend runtime environment, enabling JavaScript execution on the server. Its event-driven, non-blocking I/O model supports efficient handling of multiple concurrent requests, essential for real-time chat and prediction services.

3.2.5 TensorFlow.js:

TensorFlow.js, a JavaScript library for machine learning, is utilized to implement and train the Feedforward Neural Network for disease prediction. Its ability to run both in the browser and on the server allows flexible deployment options and reduces latency in prediction tasks. The FFNN model processes symptom inputs to classify probable diseases, providing users with immediate, data-driven insights.

3.2.6 Multer:

Multer is employed as middleware to handle multipart/form-data, specifically for uploading user profile images. It processes incoming files, temporarily stores them, and passes them to the backend logic for further handling.

3.2.7 Cloudinary:

Cloudinary provides cloud-based image storage, transformation, and delivery services [12]. By offloading image storage and management to Cloudinary, the application benefits from optimized image loading times, scalability, and secure access control, enhancing overall performance and user experience.

CHAPTER - 4

Conclusions and future work

4.1 Conclusion

The integration of digital health resources, artificial intelligence, and fitness tracking into a unified platform represents a significant advancement in the landscape of personal health management [5] . This project set out to address the fragmentation seen in existing digital health solutions by developing a comprehensive encyclopedia application that empowers users to access disease information, predict potential illnesses based on symptoms, interact with peers, and monitor their fitness activities-all within a single, user-friendly interface.

The methodology adopted in this project was both systematic and user-centric, beginning with a thorough requirement analysis and progressing through modular system design, robust data modeling, AI model development, and iterative testing. The use of the MERN stack (MongoDB, Express.js, React.js, Node.js) provided a solid technological foundation, ensuring scalability, maintainability, and a seamless user experience. The integration of TensorFlow.js enabled the implementation of a FFNN)capable of providing real-time disease predictions based on user-input symptoms, enhancing the platform's intelligence and responsiveness.

A notable innovation in this project is the in-browser deployment of the AI model using TensorFlow.js, which allows users to receive instant predictions without the latency or privacy concerns associated with server-side computation. This approach, combined with the robust backend and intuitive frontend, ensures that users can access critical health insights quickly and securely.

The incorporation of Multer and Cloudinary for user profile image management further streamlined the user experience, enabling efficient, secure, and scalable image uploads and storage [12]. This technical choice offloaded the resource burden from the application server and leveraged cloud infrastructure for media management, contributing to the overall performance and reliability of the platform.

From a functional perspective, the application successfully bridges the gap between disease encyclopedias, symptom checkers, fitness trackers, and social health communities. Users can search for diseases by name, receive AI-powered predictions by entering up to four symptoms,

engage in real-time chat with other users for support and advice, and comprehensively track their fitness activities with advanced filtering and export options. This holistic approach not only enhances user engagement but also supports preventive healthcare by promoting awareness, early detection, and healthy lifestyle choices.

The project's literature review and technology survey highlighted the absence of such integrated solutions in the current digital health ecosystem. By addressing this gap, the application stands as a model for future personal health management systems, demonstrating the value of combining AI, cloud technology, and modern web development frameworks.

Nevertheless, the project also encountered challenges, including the need for high-quality, diverse training data for the AI model, ensuring data privacy and security, and designing a user interface that balances functionality with simplicity. Addressing these challenges required careful planning, iterative testing, and the adoption of best practices in software development and machine learning.

Looking forward, the platform offers a robust foundation for further enhancements. Potential future work includes expanding the disease and symptom database, integrating additional AI models for more complex health predictions, supporting wearable device data integration, and enhancing community features for richer peer interaction. Additionally, ongoing research into privacy-preserving AI and secure data sharing will be vital as the platform scales and attracts a larger user base.

In conclusion, this project demonstrates the feasibility and benefits of an integrated, AI-driven health encyclopedia and fitness platform. By leveraging modern technologies and a user-focused methodology, it provides a powerful tool for individuals seeking to take control of their health in an informed, proactive, and connected manner. The success of this application underscores the transformative potential of digital health innovation in improving personal and public health outcomes.

4.2 Future Work

The future of digital health platforms is rapidly evolving, driven by advances in artificial intelligence, connectivity, privacy, and user-centric design. Building on the current capabilities of the encyclopedia application, several promising directions for future work are identified, aligned with industry trends and best practices:

4.2.1 Advanced AI and Hyper-Personalized Health Services

- **AI-Powered Personal Health Assistants:** Future versions can integrate more sophisticated AI, including generative AI and large language models, to provide proactive, always-on health guidance, personalized recommendations, and real-time analysis of user data [9].
- .**Predictive and Preventive Healthcare:** Leveraging predictive analytics to identify health risks and suggest preventive measures before issues arise will further empower users to manage their health proactively.

4.2.2 Enhanced Communication and Social Features

- **End-to-End Message Encryption:** To ensure the privacy and security of sensitive health conversations, implementing robust end-to-end encryption for all chat and messaging features is essential.
- .**Block and Remove Friend Functionality:** Introducing options for users to block or remove friends will give users greater control over their interactions, fostering a safer and more comfortable community environment.
- **Video Calling Between Users:** Integrating secure, high-quality video calling will enable real-time consultations, peer support, and virtual health sessions, aligning with the growing trend of telemedicine and remote care.

4.2.3. Integration with Wearables and IoT Devices

- **Real-Time Health Monitoring:** Connecting with wearable devices (e.g., smartwatches, fitness bands, medical sensors) will enable continuous tracking of vital signs, activity, and health metrics, offering users and providers actionable insights.
- **Internet of Medical Things (IoMT):** Expanding the platform’s interoperability to include a broader range of connected health devices will enhance chronic disease management, emergency alerts, and personalized coaching.

4.2.4. Data Security, Privacy, and Blockchain

- **Blockchain for Health Data Security:** Exploring blockchain technology can provide tamper-proof, transparent, and user-controlled health records, enhancing trust and compliance with data protection regulations [14].
- .**Granular Privacy Controls:** Allowing users to manage data sharing preferences and access controls will further enhance trust and adoption.

4.2.5. Telemedicine and Remote Consultations

- **Virtual Healthcare Integration:** Embedding telemedicine features such as video consultations, remote diagnostics, and digital prescriptions will make healthcare more accessible, especially for users in remote or underserved areas.
- .**Automated Scheduling and Follow-Ups:** AI-driven scheduling, reminders, and follow-up management will streamline patient-provider interactions and improve care continuity.

4.2.6. Gamification and Community Engagement

- **Gamified Health Challenges:** Incorporating rewards, challenges, and progress tracking can motivate users to maintain healthy habits and engage more deeply with the platform.
- **Moderation and Reporting Tools:** Advanced moderation, reporting, and content filtering will help maintain a positive and supportive community environment.

4.2.7. Clinical Validation and Expansion

- **Collaboration with Healthcare Providers:** Partnering with clinicians and institutions for clinical validation of AI predictions and platform features will ensure accuracy, safety, and regulatory compliance.
- **Expanding Disease and Symptom Database:** Continuously updating and broadening the knowledge base to cover more conditions, languages, and emerging health threats will keep the platform relevant and valuable

CHAPTER - 5

PROGRESS SCHEDULE SEMESTER WISE

A well-structured development schedule plays a crucial role in the timely and efficient completion of any software project. This chapter presents the **semester-wise progress schedule** of the project titled *"Predictive Valuation Model for Pre-owned Vehicles."* The development process was divided into two major phases—**Planning, Research, and Initial Development (Semester 1)** and **Development, Integration, and Final Evaluation (Semester 2)**. Each phase contributed to the gradual and systematic progress of the project.

5.1 Semester 7: Planning, Research, and Initial Development

The first semester focused on building a strong foundation by understanding the domain, conducting detailed research, gathering requirements, and planning the architecture and methodology. The key activities completed during this phase are:

5.1.1 Problem Identification & Literature Survey

- Studied fragmentation in existing digital health tools (fitness trackers, symptom checkers, health encyclopedias).
- Identified gaps in interoperability, real-time community support, AI integration, and data privacy.
- Reviewed related academic research and market products (e.g., WebMD, Ada Health, MyFitnessPal).

5.1.2 Requirement Analysis

- Functional Requirements:
    - Disease search engine.
    - Symptom-based condition prediction.
    - Peer-to-peer health chat.
    - Fitness data tracking and visualization.

- Non-functional Requirements:
    - Responsive UI
    - Scalable architecture (MERN)
    - Privacy and security compliance
    - Offline-first disease lookup

5.1.3. Technology Stack Finalization

- Selected **MERN Stack** (MongoDB, Express.js, React.js, Node.js) for full-stack development.
- Chose **TensorFlow.js** for browser-based AI model deployment.
- Decided on **Multer** for image handling and **Cloudinary** for media storage.

5.1.4 System Design

5.1.4.1 Application Architecture

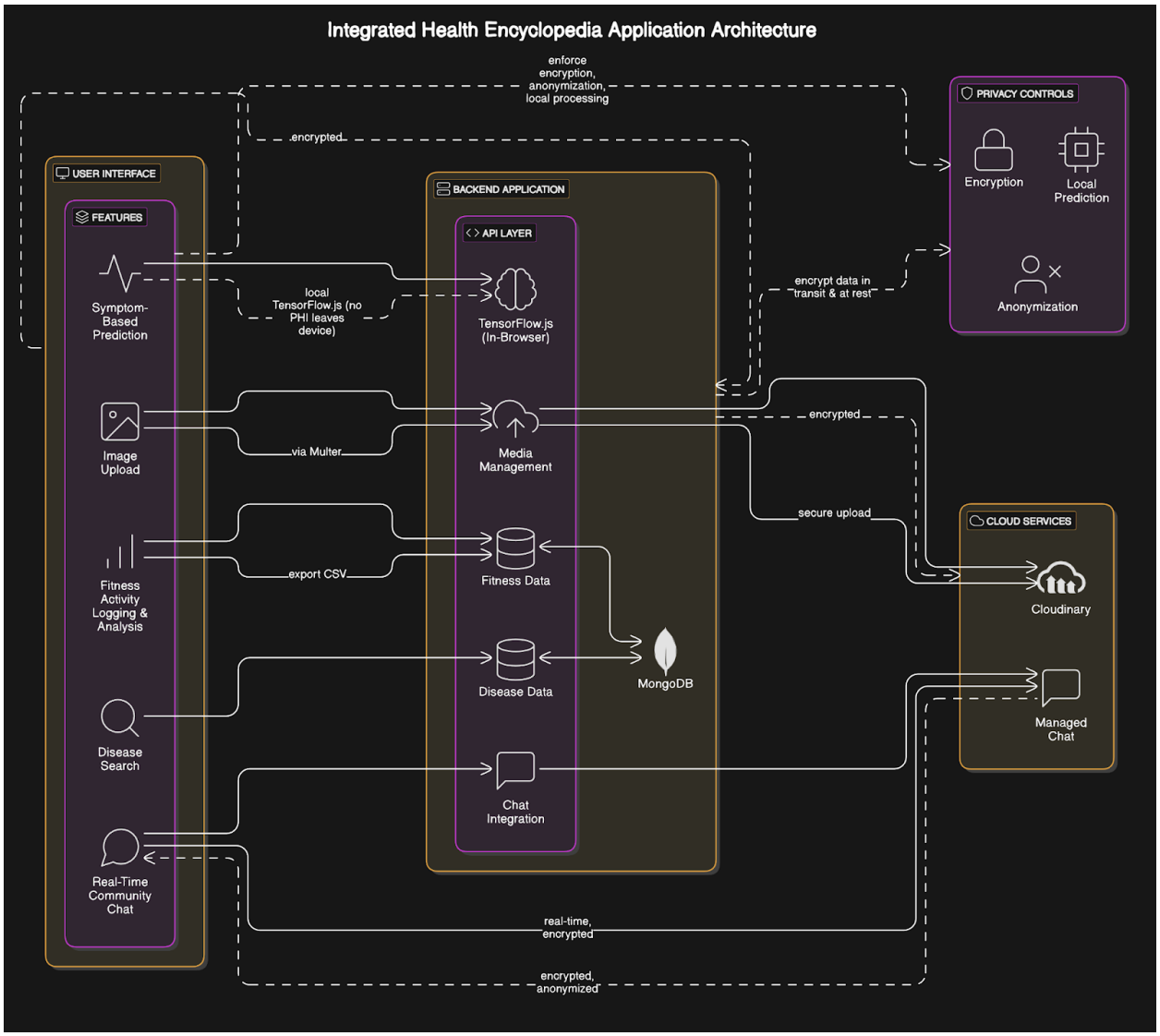

Fig 5.1.4.1 Architecture Diagram

5.1.4.2 DFD

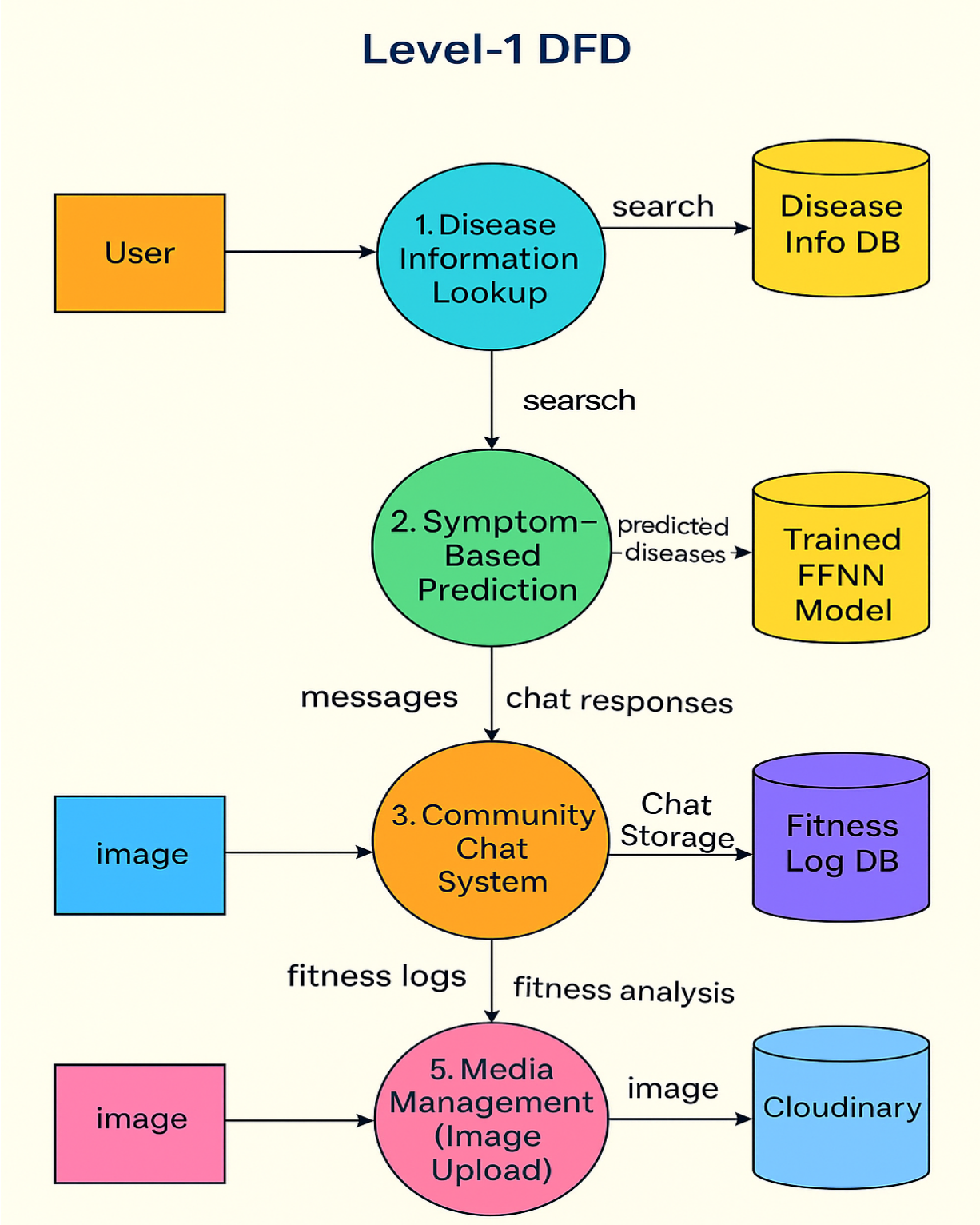

Fig 5.1.4.2 DFD

5.1.4.3 Use Case Diagram

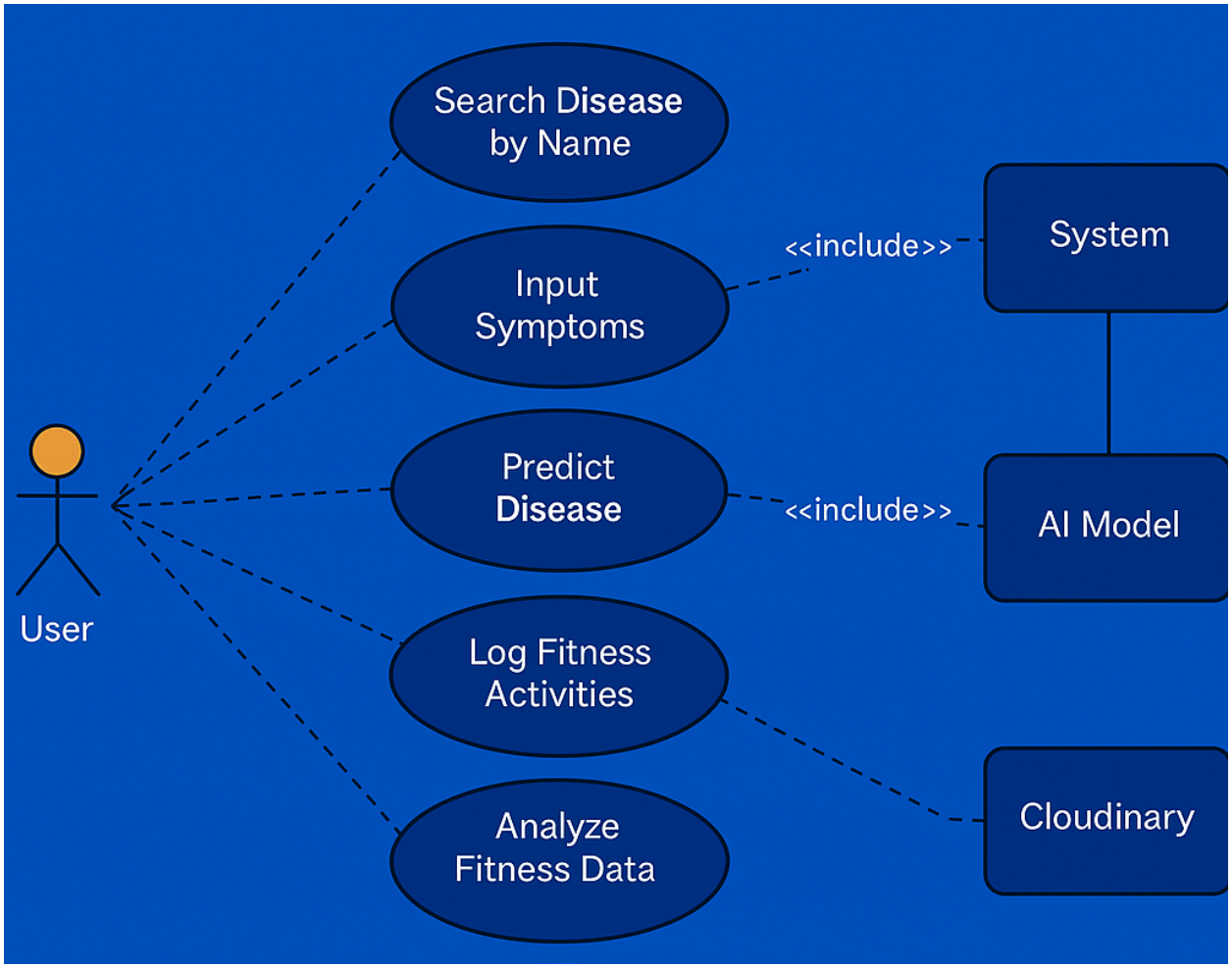

Fig 5.1.4 Use Case Diagram

5.1.5 Dataset Collection and Preparation

- Collected medical datasets for disease descriptions and symptom-condition mappings.
- Preprocessed and structured data for use in training the FFNN prediction model.

```
{
  "name": "Acne",
  "description": "...",
  "symptoms": ["..."],
  "causes": ["..."],
  "stages": [{...}],
  "treatment": ["..."],
  "image_url": "..."
}
```

Fig 5.1.5.1 Disease DataSet

5.1.6 Database Schema Design (MongoDB)

Collections:

- users: stores credentials, health data, fitness logs.
- conditions: disease encyclopedia entries with symptoms, causes, treatments, stages,name,etc.
- chat_messages: real-time chat records.
- fitness_logs: calories, steps, workouts, filterable & exportable.
- images: metadata of uploaded images (via Cloudinary).
- Requests: store friend requests of the user.

5.1.7 AI Disease Prediction Module

- Created a symptom vectorizer that maps up to 4 symptoms into binary input for FFNN.
- Model trained on labeled synthetic dataset (~200 diseases, 600+ symptom combinations).
- Deployed via TensorFlow.js in-browser for privacy-preserving inference.

Model Architecture:

- Input: Symptom vector (1xN).

- Hidden: 2 dense layers (ReLU activation).
- Output: Softmax for disease classification.

5.1.8 Frontend UI/UX Prototyping

- Developed mockups using Figma.
- Responsive design using Material UI + React.
- Prototypes for:
    - Home, Disease Search, Chatroom, Fitness Dashboard, Profile.

5.1.9 Initial Implementation

- Setup Git repo with structured MERN boilerplate.
- Implemented user authentication (JWT).
- Created disease encyclopedia search and condition viewer.

5.2 Semester 2: Development, Integration, and Final Evaluation

5.2.1 Full Stack Integration

- Connected all modules (frontend, backend, database, AI, image handling).
- Integrated Cloudinary via Multer for user-uploaded health or fitness images.
- Implemented CSV export & advanced filtering in fitness logs.

5.2.2 Community Chat Module

- Real-time communication using Socket.io.
- Users talk to each others by making friends.

5.2.3 Finalized AI Integration

- Optimized TensorFlow.js FFNN for mobile devices.
- Added symptom autocomplete from a dictionary.
- In-browser prediction output is cached locally.

5.2.4 Advanced Features

- Fitness Module Enhancements:
    - Calorie tracker, step counter, workout logger.
    - Graphs using Chart.js for daily/weekly trends.
    - Export logs to CSV.

- Search Enhancements:
    - Predictive search suggestions for diseases.
    - Highlighting symptom matches.

5.2.4 Testing and Debugging

- Performed unit, integration, and system testing.
- Conducted user testing sessions to gather feedback on usability, speed, and functionality.
- Applied iterative improvements based on testing results.

5.2.5 Documentation & Report Writing

- Compiled system architecture, flow diagrams, and user manuals.
- Documented the AI model architecture, training process, and validation results.
- Prepared final report including methodology, research insights, and future scope.

5.2.6 Future Scope and Enhancements Planning

- Outlined potential extensions:
    - Wearable device data integration (e.g., heart rate, sleep tracking).
    - Blockchain-based secure health records.
    - Encrypted telehealth and video consultations.
- Drafted proposals for further academic or commercial development.

5.3 Database design

The **database design** is crucial for the **FitLife : A Encylopedia** to store and retrieve the necessary data efficiently. A well-structured database ensures data integrity, security, and quick access.

The system comprises **seven major collections**, each representing a key functional domain of the platform:

i. Users Collection

Field Name	Type	Description
_id	**ObjectId**	**Primary key (auto-generated)**
name	**String**	**Full name of the user**
email	**String**	**Unique user email**
password_hash	**String**	**Encrypted password**
avatar	**string**	**Profile of the user**
gender	**String**	**Optional, for analytics**
created_at	**Date**	**Account creation timestamp**

Table 5.3.1 Users Collection

ii. Conditions Collection

Field Name	Type	Description
_id	**ObjectId**	**Primary key**

name	**String**	**Disease/condition name**
description	**String**	**Medical overview**
causes	**Array of Strings**	**Root causes of the condition**
symptoms	**Array of Strings**	**Typical symptoms**
risk_factors	**Array of Strings**	**Risk factors for the condition**
treatment	**Array of Strings**	**Treatment approaches**
prevention	**Array of Strings**	**Preventive measures**
stages	**Array of Objects**	**Staged details (e.g., mild, moderate, severe)**
image_url	**String**	**Link to representative image (via Cloudinary)**

Table 5.3.2 Disease Collection

iii. FitnessLogs Collection

Field Name	Type	Description
_id	ObjectId	Primary key
user_id	ObjectId	Reference to associated user
date	Date	Log entry date
calories_burned	Number	Estimated calories burned
steps_walked	Number	Total steps recorded
exercise_type	String	(e.g., walking, running, gym)
duration_minutes	Number	Duration of exercise in minutes
notes	String	Optional notes from user

Table 5.3.3 Fitness Collection

iv. ChatMessages Collection

Field Name	Type	Description
_id	ObjectId	Primary key
user_id	ObjectId	Reference to the message sender
room_id	ObjectId	Reference to the chat room
message_text	String	Message content
timestamp	Date	Sent time

deletedBy	**ObjectId**	**Reference who delete the message**
deletedFor	**Array of ObjectId**	**Array of Reference for whom the message is deleted**
readBy	**ObjectId**	**User who reads the message**

Table 5.3.4 Messages Collection

v. ChatRooms Collection

Field Name	Type	Description
_id	**ObjectId**	**Primary key**
members	**Array of ObjectId**	**Contains members of this chat**

Table 5.3.5 ChatRooms Collection

vi. Notifications

Field Name	Type	Description
_id	**ObjectId**	**Primary key**
Sender	**ObjectId**	**Sender of the notification**
Receiver	**ObjectId**	**Receiver of the notification**
Content	**String**	**Content of the notification**

Table 5.3.6 Notifications Collection

vii.Request

Field Name	Type	Description
_id	**ObjectId**	**Primary key**
Sender	**ObjectId**	**Sender of the notification**
Receiver	**ObjectId**	**Receiver of the notification**
isActive	**Boolean**	**Whether request accepted or not**

Table 5.3.7 Requests Collection

5.3.1 Database Relationships

- Users Collection → FitnessLogs Collection: One-to-many relationship (one user can have multiple fitness logs).
- Users Collection → SymptomQueries Collection: One-to-many relationship (one user can submit multiple symptom queries).
- SymptomQueries Collection → Conditions Collection: Many-to-one relationship (multiple queries can predict the same condition).
- Users Collection → ImageUploads Collection: One-to-many relationship (one user can upload multiple images).
- Users Collection → ChatMessages Collection: One-to-many relationship (one user can send multiple chat messages).
- ChatRooms Collection → ChatMessages Collection: One-to-many relationship (one room can contain multiple messages).
- Users Collection ↔ ChatRooms Collection: Many-to-many relationship (indirectly through chat messages; users can participate in multiple rooms).

5.4 Coding

Train.js

```
 const tf = require('@tensorflow/tfjs-node');

const fs = require('fs');

// Load the cleaned dataset

const diseases = JSON.parse(fs.readFileSync('diseases.json', 'utf-8'));

// Build unique symptom list

const allSymptoms = Array.from(new Set(diseases.flatMap(d => d.symptoms.map(s => s.toLowerCase()))));

// Map symptoms to indexes

const symptomToIndex = {};

allSymptoms.forEach((symptom, idx) => symptomToIndex[symptom] = idx);

// Map diseases to labels

const labelToIndex = {};

const indexToLabel = {};

diseases.forEach((disease, idx) => {

labelToIndex[disease.name] = idx;

indexToLabel[idx] = disease.name;

});

// Prepare input and output
```

```
const inputs = [];

const labels = [];

diseases.forEach(disease => {

const symptomVector = Array(allSymptoms.length).fill(0);

disease.symptoms.forEach(symptom => {

if (symptomToIndex.hasOwnProperty(symptom.toLowerCase())) {

symptomVector[symptomToIndex[symptom.toLowerCase()]] = 1;

}

});

inputs.push(symptomVector);

labels.push(labelToIndex[disease.name]);

});

// Convert to tensors

const xs = tf.tensor2d(inputs);

const ys = tf.tensor1d(labels, 'float32');

// Build model

const model = tf.sequential();

model.add(tf.layers.dense({ inputShape: [allSymptoms.length], units: 256, activation: 'relu' }));

model.add(tf.layers.dense({ units: 128, activation: 'relu' }));

model.add(tf.layers.dense({ units: diseases.length, activation: 'softmax' }));
```

```
model.compile({
optimizer: tf.train.adam(),
loss: 'sparseCategoricalCrossentropy',
metrics: ['accuracy']
});
// Train and save
async function train() {
await model.fit(xs, ys, {
epochs: 100,
batchSize: 16,
validationSplit: 0.1,
callbacks: {
onEpochEnd: (epoch, logs) => {
console.log(`Epoch ${epoch + 1}: loss=${logs.loss.toFixed(4)}, acc=${logs.acc.toFixed(4)}`);
}
}
});
await model.save('file://model');
fs.writeFileSync('symptoms_list.json', JSON.stringify(allSymptoms));
fs.writeFileSync('labels.json', JSON.stringify(indexToLabel));
```

```
console.log('✅ Model training completed and files saved!');

}

train();
```

Predict.js

```
const tf = require('@tensorflow/tfjs-node');

const diseases = require('./diseases.json');

const allSymptoms = [...new Set(diseases.flatMap(d => d.symptoms))];

const allDiseases = diseases.map(d => d.name);

function vectorize(userSymptoms) {

const lowerUserSymptoms = userSymptoms.map(s => s.toLowerCase());

return allSymptoms.map(symptom =>

lowerUserSymptoms.some(userSymptom =>

symptom.toLowerCase().includes(userSymptom)                                    ||
userSymptom.includes(symptom.toLowerCase())

) ? 1 : 0

);

}

async function predictDisease(userSymptoms) {

const model = await tf.loadLayersModel('file://./model/model.json');

const input = tf.tensor2d([vectorize(userSymptoms)],[1, allSymptoms.length], 'float32');
```

```
const prediction = model.predict(input);

const result = await prediction.argMax(-1).data();

const predictedDisease = allDiseases[result[0]];

const diseaseInfo = diseases.find(d => d.name === predictedDisease);

return diseaseInfo;

}

module.exports = predictDisease;
```

UserController.js

```
const ApiError = require('../Handlers/ApiError');

const ApiResponse = require('../Handlers/ApiResponse');

const asyncHandler = require('../Handlers/AsyncHandler');

const users=require('../Models/UserSchema')

const jwt=require('jsonwebtoken')

const uploadOnCloudinary=require('../MiddleWares/Cloudinary');

const contact = require('../Models/ContactSchema');

const reviews=require('../Models/ReviewSchema')

const signup=asyncHandler(async(req,res)=>{

const user=await users.findOne({$or:[{name:req.body.name},{email:req.body.email}]})

if(user) throw new ApiError(401,"User Already Exist")

req.body.avatar= await uploadOnCloudinary(req.file.path)
```

```
await users.create(req.body);
res.json(new ApiResponse(201,null,"SignUp Successfully"))
})
const review=asyncHandler(async(req,res)=>{
const user=await users.findById(req.user._id)
const rev=await reviews.findOne({name:user.name})
if(rev) await reviews.findOneAndUpdate({name:user.name},{$set:{text:req.body.text}})
else
await reviews.create({name:user.name,text:req.body.text});
res.json(new ApiResponse(201,null,"Review Successfully"))
})
const getReview=asyncHandler(async(req,res)=>{
const user=await reviews.find({}).sort({ updatedAt: -1 }).limit(3)
res.json(new ApiResponse(201,user,"Review Successfully"))
})
const login=asyncHandler(async(req,res)=>{
const user=await users.findOne({email:req.body.email})
if(!user) throw new ApiError(401,"Not Registered")
const bol=await user.getPassword(req.body.password)
if(!bol) throw new ApiError(401,"Wrong Password")
```

```
const token= await user.generateAccessToken()

res.cookie("access",token,{httpOnly:true,secure:false}).json(new    ApiResponse(200,user,"Login Successfully"))

})

const logout=asyncHandler(async(req,res)=>{

res.clearCookie("access").json(new ApiResponse(200,null,"Logout Successfully"))

})

const getUser=asyncHandler(async(req,res,next)=>{

if(!req.cookies.access){

throw new ApiError(401,"Unauthorized Access")

}

const token= jwt.verify(req.cookies.access,'Vaibhav')

req.user=token

if(token){

const user= await users.findOne({_id:token})

res.json(new ApiResponse(204,user,"Fetched Successfully"))

}

})

const updateUser=asyncHandler(async(req,res)=>{

var user={}
```

```
if(req.body.name){
user= await users.findOneAndUpdate({_id:req.user},{$set:{name:req.body.name}})
}
else{
user= await users.findOneAndUpdate({_id:req.user},{$set:{email:req.body.email}})
}
res.json(new ApiResponse(200,user,"Updated Successfully"))
})
const updateUserProfile=asyncHandler(async(req,res)=>{
const re=await uploadOnCloudinary(req.file.path)
const user= await users.findOneAndUpdate({_id:req.user},{$set:{avatar:re}})
const us=await users.findOne({_id:req.user})
res.json(new ApiResponse(200,us,"Updated Successfully"))
})
const contactUs=asyncHandler(async(req,res)=>{
console.log(req.body)
const r= await contact.create(req.body)
res.json(new ApiResponse(200,r,"Saved Successfully"))
})
```

module.exports={signup,login,logout,getUser,updateUser,updateUserProfile,contactUs,review,getReview}

App.jsx

```
import { useContext, useEffect, useState } from "react";

import { BrowserRouter, createBrowserRouter, Route, RouterProvider, Routes } from "react-router-dom";

import { toast, ToastContainer } from "react-toastify";

import { getUser } from "./Api/userApi";

import "./App.css";

import BodyShapingPage from "./Components/BodyShaping";

import ChatPage from "./Components/ChatPage";

import Fitness from "./Components/Fitness";

import Hero from "./Components/Hero";

import Login from "./Components/Login";

import { useDispatch } from "react-redux";

import { socketContext } from "./Context/SocketContext";

import { userContext } from "./Context/UserContext";

import { Protected, Public } from "./Components/ProtectedRoutes";

import ChatApp from "./Components/ChatBot";

import NotFoundPage from "./Components/ErrorPage";

function App() {
```

```
const { user, setUser, isuser, setIsUser } = useContext(userContext);

const { socket } = useContext(socketContext);

const[onlineUsers,setOnlineUsers]=useState([])

const dispatch = useDispatch();

useEffect(()=>{

socket.on('friend-online',(data)=>{

setOnlineUsers(data)

})

socket.on('friend-offline',(data)=>{

setOnlineUsers(prev=>prev.filter(val=>val!=data))

})

socket.on('error',(error)=>{

toast.error(error)

})

},[])

useEffect(()=>{

localStorage.setItem('onlineUsers',JSON.stringify(onlineUsers))

},[onlineUsers])

useEffect(() => {

const fetch = async () => {
```

```
const user = await getUser();
if(user.data.message=="Fetched Successfully"){
setUser(user.data.data);
socket.emit("join", user.data.data._id);
setIsUser(true)
}
else {
localStorage.setItem('isAuthenticated',false)
setIsUser(false)
}
};
fetch();
}, []);
return (
<>
<ToastContainer />
{/* <RouterProvider router={router}></RouterProvider> */}
<BrowserRouter future={{ v7_startTransition: true }}>
<Routes>
<Route path="/" element={<Fitness/>}/>
```

<Route path="/shape" element={<Protected><BodyShapingPage/></Protected>}/>

<Route path="/disease" element={<Protected><Hero/></Protected>}/>

<Route path="/chats/id=?" element={<Protected><ChatPage onlineUsers={onlineUsers}/></Protected>}>

</Route>

<Route path="/login" element={<Public><Login/></Public>}/>

{/* <Route path='/chatbot' element={<Protected><ChatApp/></Protected>}/> */}

<Route path="*" element={<NotFoundPage/>} ></Route>

</Routes>

</BrowserRouter>

</>

);

}export default App;

5.5 Output

Home Page

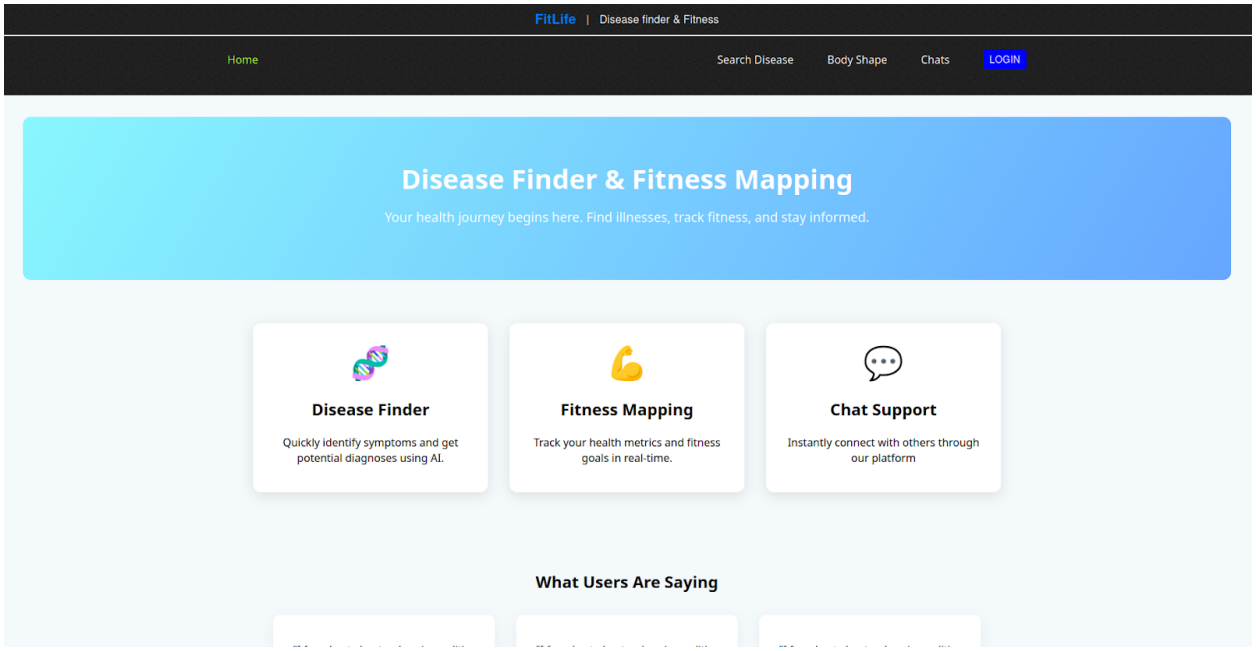

Fig 5.5.1 Home Page

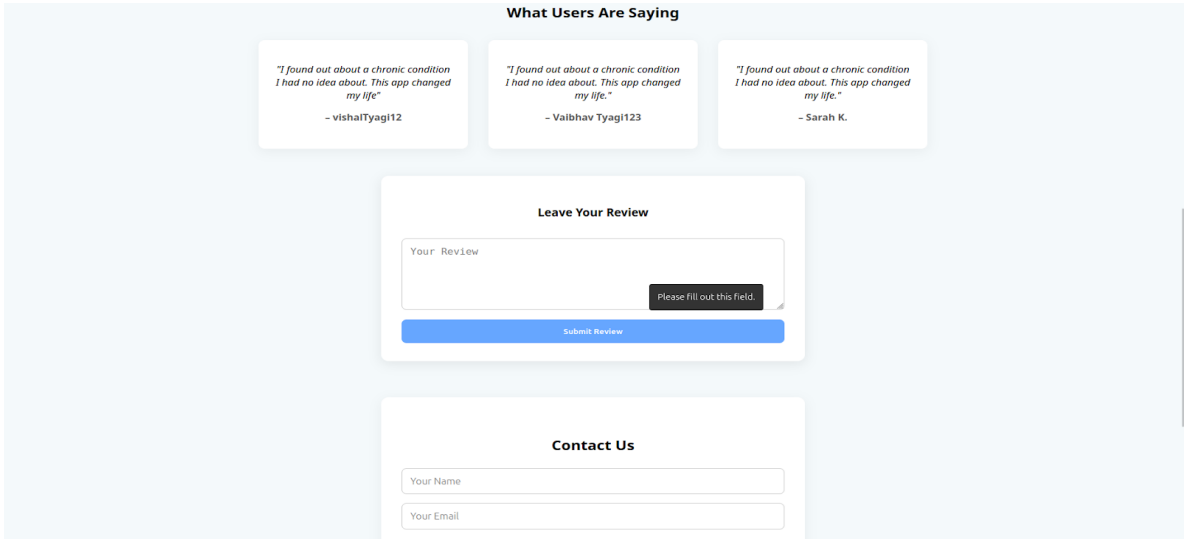

Fig 5.5.2 Home Page

Login Page

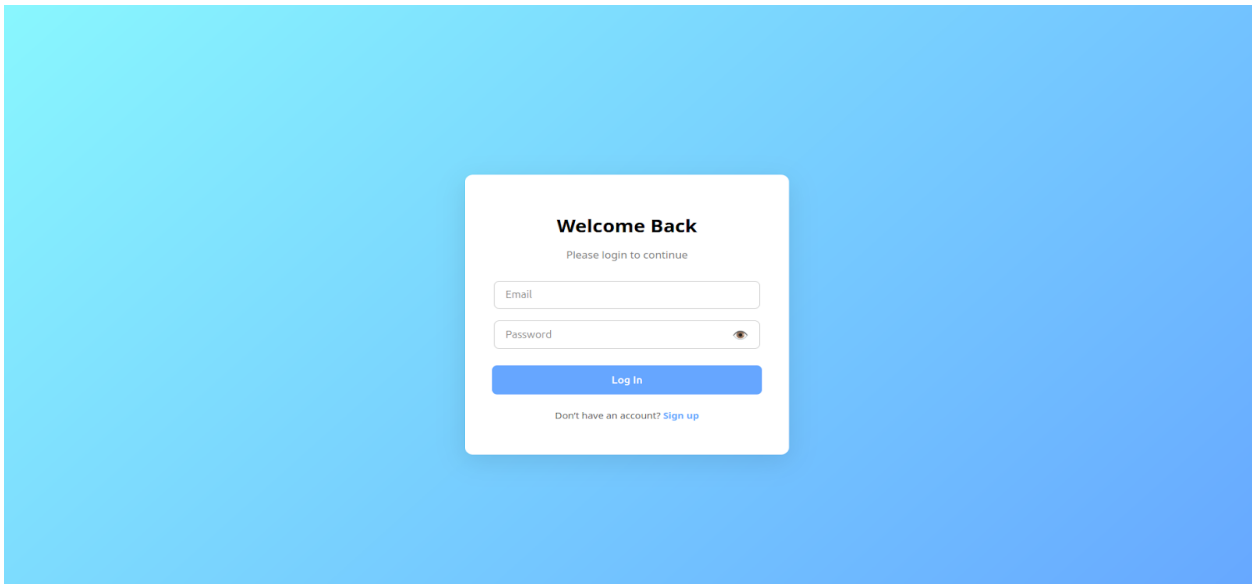

Fig 5.5.3 Login Page

Signup Page

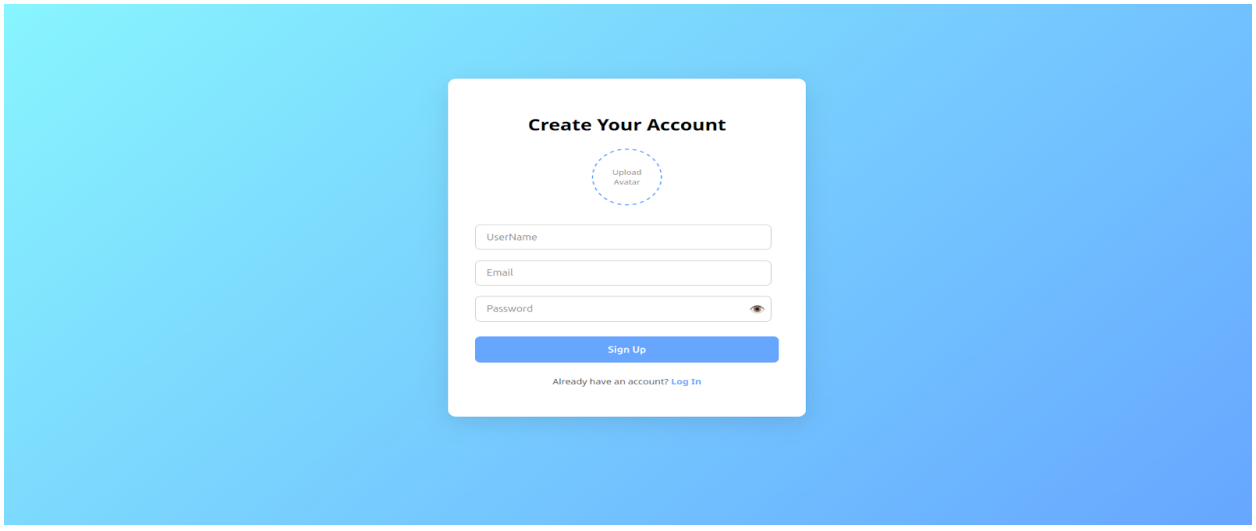

Fig 5.5.4 Signup Page

Disease Page without data

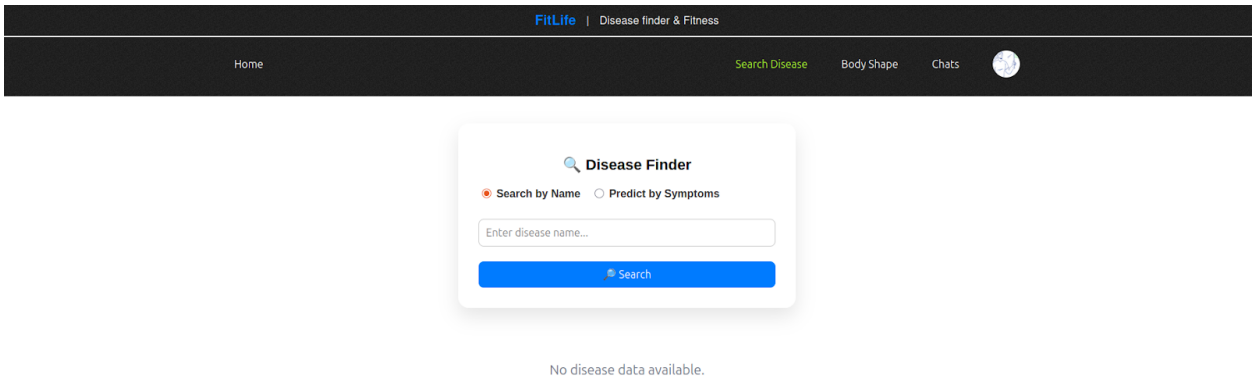

Fig 5.5.5 Disease Page without data

Disease Page with data

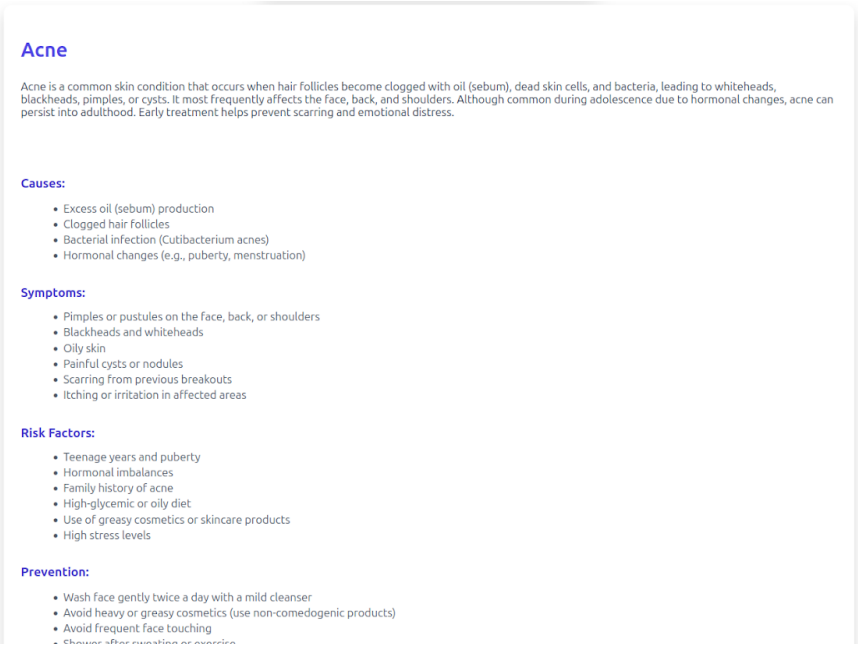

Fig 5.5.6 Disease Page with data

Fitness Map Page with Monthly View

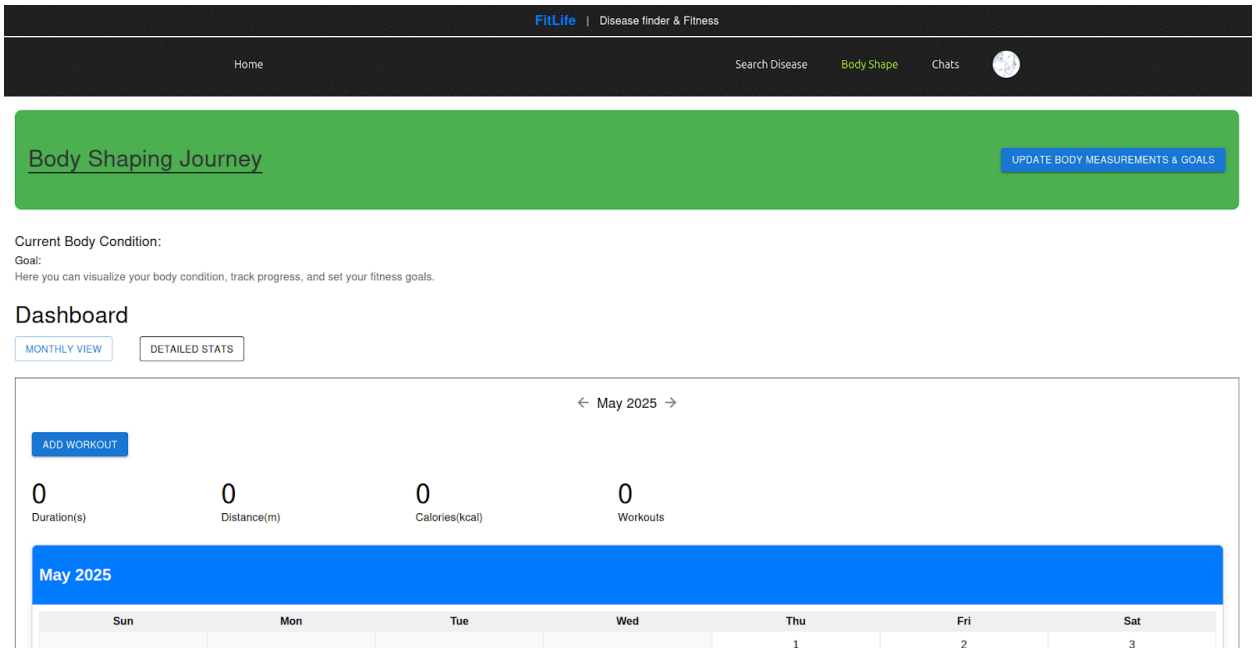

Fig 5.5.7 Fitness Map Page with Monthly View

Fitness Map Page with Detailed View

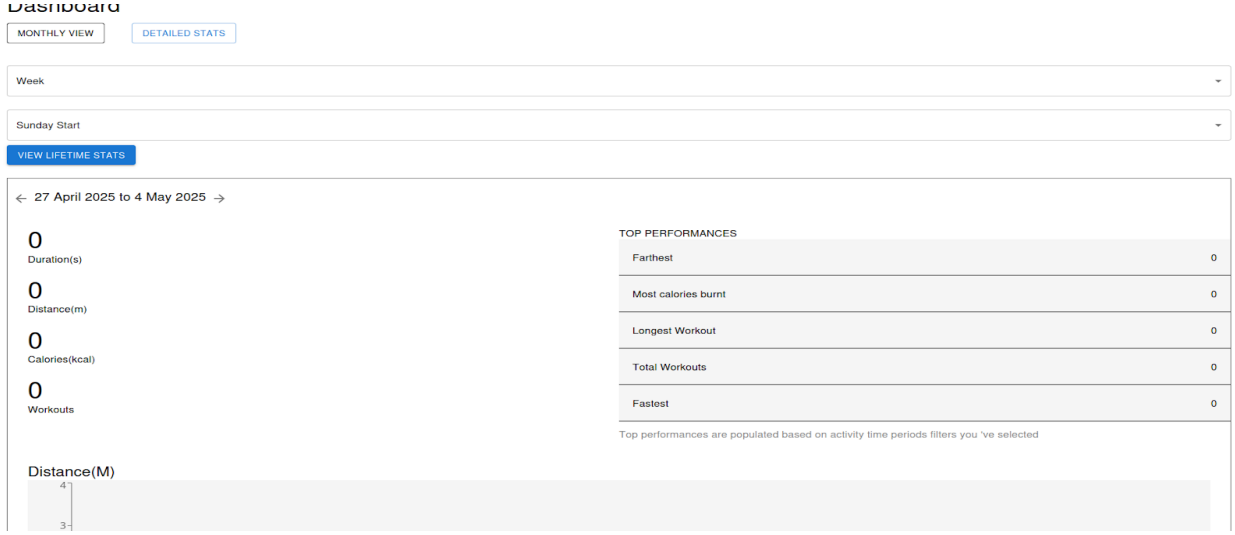

Fig 5.5.8 Fitness Map Page with Detailed View

Fig 5.5.9 Fitness Map Page with Detailed View(charts)

Chat Page

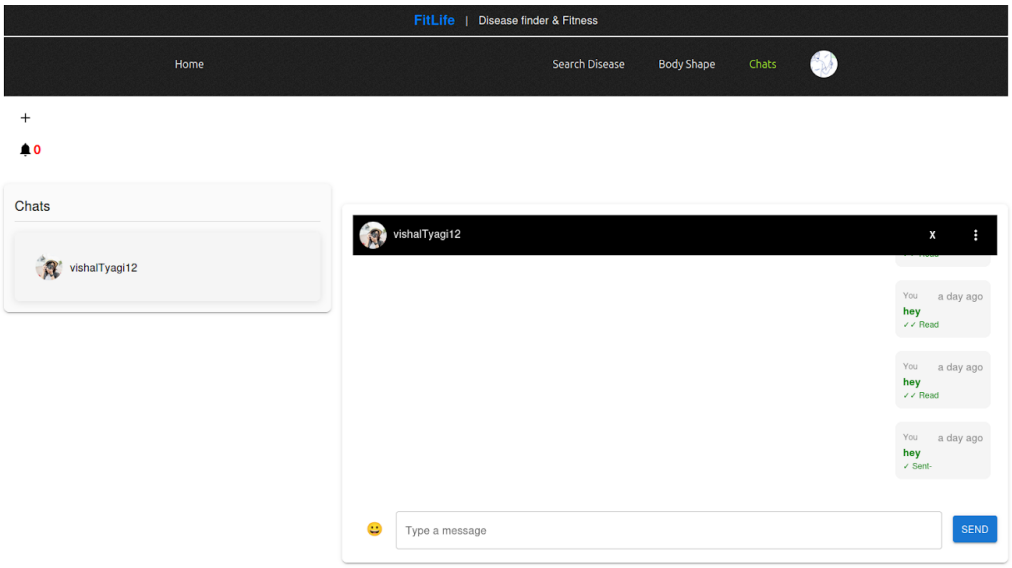

Fig 5.5.10 Chat Page

UserProfile

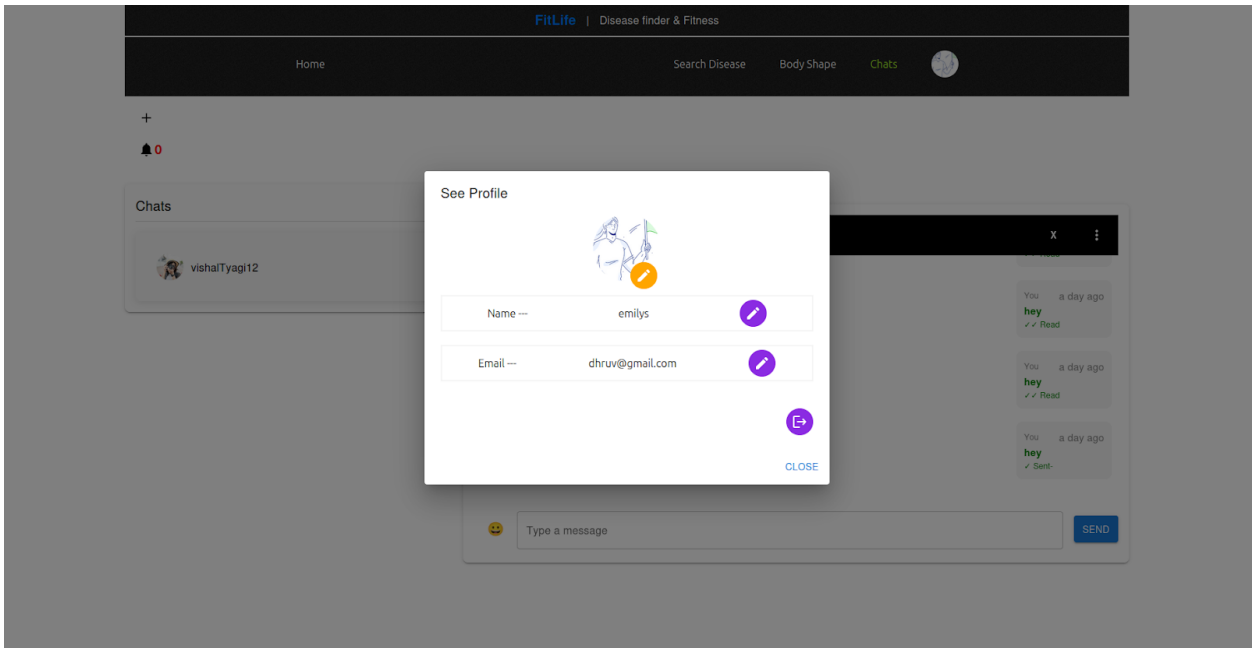

Fig 5.5.11 UserProfile Page

5.6 Software Features

The application offers a comprehensive suite of features designed to empower users with tools for disease awareness, health prediction, fitness monitoring, and community support. These features are developed with a focus on usability, privacy, and real-time interactivity.

5.6.1 Disease Search Module

- Search diseases by name with autocomplete support.
- View structured disease information including:
    - Description, symptoms, causes, risk factors, prevention, treatment, and stages.

5.6.2 Symptom-Based Disease Prediction

- Accepts up to four symptoms entered by the user.

- Uses a trained FFNN (Feedforward Neural Network) model via TensorFlow.js for real-time, in-browser disease prediction.
- Reduces latency and protects user privacy (no external API calls).

5.6.3 Fitness Log Tracker

- Users can log:
    - Exercise type, duration, calories burned, steps walked, and custom notes.

- Features:
    - Filter logs by date or activity.
    - Export fitness data as CSV for external use or analysis.

5.6.4 Community Chat Room

- Real-time chat system for user discussion and peer support.

5.6.5 Secure Image Upload

- Users can upload images (e.g., for dermatological conditions).
- Images are stored securely in **Cloudinary** using **Multer** for handling file uploads.
- Supports preview and deletion of user-uploaded images.

5.6.6 Authentication and Authorization

- User registration and login with hashed password storage.
- Token-based session management (e.g., JWT) for secure interactions.

5.6.7 Responsive User Interface

- Built with **React.js** and **Material UI** for a modern, mobile-friendly design.
- Seamless navigation between modules.
- Responsive layout with optimized loading for various screen sizes.

REFERENCES

[1] World Health Organization, *Global strategy on digital health 2020–2025*. WHO, 2021. [Online]. Available: https://www.who.int/publications/i/item/9789240020924

[2] J. W. Hinton, D. S. King, and M. P. Johnson, "Artificial intelligence in healthcare: transforming the practice of medicine," *Future Healthcare Journal*, vol. 8, no. 2, pp. 1–7, 2021. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8285156

[3] S. S. Patil *et al.*, "Disease prediction from various symptoms using machine learning," *SSRN*, Jul. 2020. [Online]. Available: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3661426

[4] J. A. Brickwood *et al.*, "Effectiveness of wearable activity trackers to increase physical activity: systematic review and meta-analysis," *The Lancet Digital Health*, vol. 3, no. 8, pp. e653–e665,2021.[Online].Available: https://www.thelancet.com/journals/landig/article/PIIS2589-7500(21)00103-1

[5] IBM iX, "8 digital trends in healthcare in 2025," IBM iX, 2025. [Online]. Available: https://www.ibm.com/industries/healthcare/insights/digital-trends-2025

[6] CSIRO, "Five trends shaping the future of digital health in 2025," CSIRO, 2025. [Online]. Available: https://www.csiro.au/en/news/2025/digital-health-trends

[7] MyDigiRecords, "Top 5 Digital Health Trends to Watch in 2025," MyDigiRecords, 2025. [Online]. Available: https://www.mydigirecords.com/blog/digital-health-trends-2025

[8] J. Bocas, "5 Digital Health Trends for 2025: Revolutionizing Healthcare," *LinkedIn*, Nov. 27, 2024.[Online].Available: https://www.linkedin.com/pulse/5-digital-health-trends-2025-revolutionizing-healthcare-joão-bocas/

[9] Boston Consulting Group, "How Digital & AI Will Reshape Health Care in 2025," BCG, Feb. 5, 2025. [Online]. Available: https://www.bcg.com/publications/2025/digital-ai-healthcare

[10] Emerline, “The Future of Healthcare: Mobile Apps Trends in 2025,” Emerline, 2025. [Online]. Available: https://www.emerline.com/blog/healthcare-mobile-app-trends-2025

[11] Codiant, “Healthcare Mobile App Trends for Smarter Patient Care 2025,” Codiant, 2025. [Online]. Available: https://www.codiant.com/blog/healthcare-app-trends-2025

[12] Quokka Labs, “Guide To Healthcare App Development in 2025,” Quokka Labs, 2025. [Online]. Available: https://quokkalabs.com/blog/guide-to-healthcare-app-development-in-2025

[13] Peerbits, “Top 10 Healthcare Software Development Trends Shaping 2025,” Peerbits, 2025. [Online].Available: https://www.peerbits.com/blog/healthcare-software-development-trends-2025.html

[14] S. M. Mehta *et al.*, “Blockchain technology in healthcare: A comprehensive review and directions for future research,” *Applied Sciences*, vol. 12, no. 3, pp. 1–20, 2022. [Online]. Available: https://www.mdpi.com/2076-3417/12/3/1102