Alzheimer's disease detection system

AMAL JAMIL
01-135212-014
EISHA AWAIS
01-135221-012

Supervisor: Ms. Alyia Amir (Associate Professor)

Bachelor of Science in Information Technology

Department of Computer Science
Bahria University, Islamabad

December 2025

Certificate

We accept the work contained in the report titled ALZHEIMER'S DISEASE DETECTION SYSTEM, written by Ms. AMAL JAMIL AND Ms. EISHA AWAIS as a confirmation to the required standard for the partial fulfillment of the degree of Bachelor of Science in Information Technology.

Approved by:

Supervisor: Ms. Alyia Amir (Associate Professor)

_____

Internal Examiner: Dr. Adil Khan (Asst. Professor), Mr. Jawwad Ijaz (Assoc. Professor)

_____

External Examiner: Ms. Sanum Zaffar

_____

Project Coordinator: Dr. Kabeer Ahmed Bhatti (Assistant Professor)

_____

Head of the Department: Dr. Muhammad Khurram Ehsan (Senior Associate Professor)

_____

December 24^{th}, 2025

Abstract

Alzheimer's disease affects memory, thinking ability, and behavior, and over time it becomes difficult for patients to manage daily activities. Because of this, identifying the disease at an earlier stage is considered important. The traditional forms of detection usually use manual MRI data analysis and clinical examination. These methods take too much time and may produce inaccurate results. This project introduces an AI-assisted Alzheimer's disease stage detection system based on the MRI images of the brain and a mobile-based application to interact with users. The deep learning ResNet50 architecture is used to categorize MRI scans into stages of Alzheimer's disease. The model is deployed as a FastAPI backend which interacts with a React Native application so that users can upload MRI scans and get predictions of the stage immediately. In addition to the prediction, the application features cognitive and memory enhancing games which are aimed at evaluating and training the mental performance, and an administration dashboard to monitor and manage the system and data. Firebase is used to provide secure user authentication, cloud-based storage of prediction results and user activity logs. This system introduces a convenient, interactive solution to the problem of detecting and learning about the Alzheimer's disease at an early age through combining the capabilities of artificial intelligence, medical imaging, and cognitive engagement.

Acknowledgments

All gratitude is due to Almighty Allah, who gave us a little fraction of His vast wisdom, as well as the good health and wellbeing required to finish this project. We wish to express our sincere thanks to Ms. Alyia Amir, for guiding us at various stages of the project and providing us with all the necessary facilities for the research. Her availability, encouragement, and direction played an important role in keeping the project on track. We are also thankful to the faculty members of the department for their cooperation and assistance whenever required. Their input and support contributed positively to the completion of this report. Lastly, we would like to express our heartfelt gratitude to our parents for their constant support, motivation, and prayers, without which this work would not have been possible.

AMAL JAMIL, EISHA AWAIS
Islamabad, Pakistan

December 2025

Contents

List of Figures

List of Tables

Acronyms and Abbreviations

AD	Alzheimer's Disease
AI	Artificial Intelligence
AUC	Area Under the Curve
CNN	Convolutional Neural Network
CT	Computed Tomography
ERD	Entity Relationship Diagram
FastAPI	Fast API (Python Web Framework)
GUI	Graphical User Interface
JSON	JavaScript Object Notation
MRI	Magnetic Resonance Imaging
PET	Positron Emission Tomography
ResNet50	Residual Network 50
SDK	Software Development Kit
SQL	Structured Query Language
TC	Test Case
TS	Test Scenario
UI	User Interface
UC	Use Case
UID	Unique Identifier
UML	Unified Modeling Language

Chapter 1

Introduction

1.1 Background

Alzheimer's disease (AD) is a progressive neurodegenerative disease that mainly affects memory, reasoning and behavior. It is among the most common causes of dementia in older adults and early detection is very important in effective management and care [1]. As the life expectancy of people worldwide increases, the population of patients with Alzheimer's is also increasing [2]. Artificial intelligence (AI) and medical imaging have created new opportunities in early and automated diagnosis. Deep learning algorithms are capable of detecting minor patterns in MRI images that a human eye might not detect. The proposed project utilizes these features by combining an AI-based Alzheimer's detection model with a mobile-friendly application. Cognitive games and activity tracking are part of the system to help keep the brain healthy and facilitate preventive awareness.

1.2 Problem Description

Early diagnosis of Alzheimer's disease is the most challenging issue; the interpretation of MRI scans manually is time-consuming, subject to human error, and highly sophisticated diagnostic tools are not always affordable or accessible [1]. The majority of existing systems are based only on prediction and do not offer preventive and interactive options to users.

The proposed project fills these gaps by creating an end-to-end solution that predicts the stage of Alzheimer's from MRI scans and also combines cognitive assessment and tracking modules.

1.3 Objectives

The key objectives of this project are

- To build a deep learning model (ResNet50-based) that can classify MRI brain images accurately between various stages of Alzheimer's disease.
- To design a user-friendly mobile interface where users can upload MRI scans and view results in real-time.
- To develop cognitive and memory-enhancing games for early-stage users to monitor mental performance.
- To design an admin panel for tracking user statistics and managing system data.
- To deliver an accessible and cost-effective tool that raises awareness and early diagnosis among the public.

1.4 Methodology

The project is implemented using an Agile methodology, which is an iterative and component-based approach. The development is structured into short sprints, which enables constant integration, testing, and optimization of the system. This methodology guarantees the incremental development, dynamic testing, and flexibility to add the changes or feedback at any point.

- Sprint 1: Dataset preparation & Model Training
- Sprint 2: Backend API development
- Sprint 3: Mobile App UI & Firebase Integration
- Sprint 4: Cognitive Games & Admin Panel
- Sprint 5: Integration Testing & Bug Fixe

As indicated in Figure 1.1, the Agile approach is utilized to organize the project into adaptive, iterative sprints, which enables the project to be continuously developed and improved.

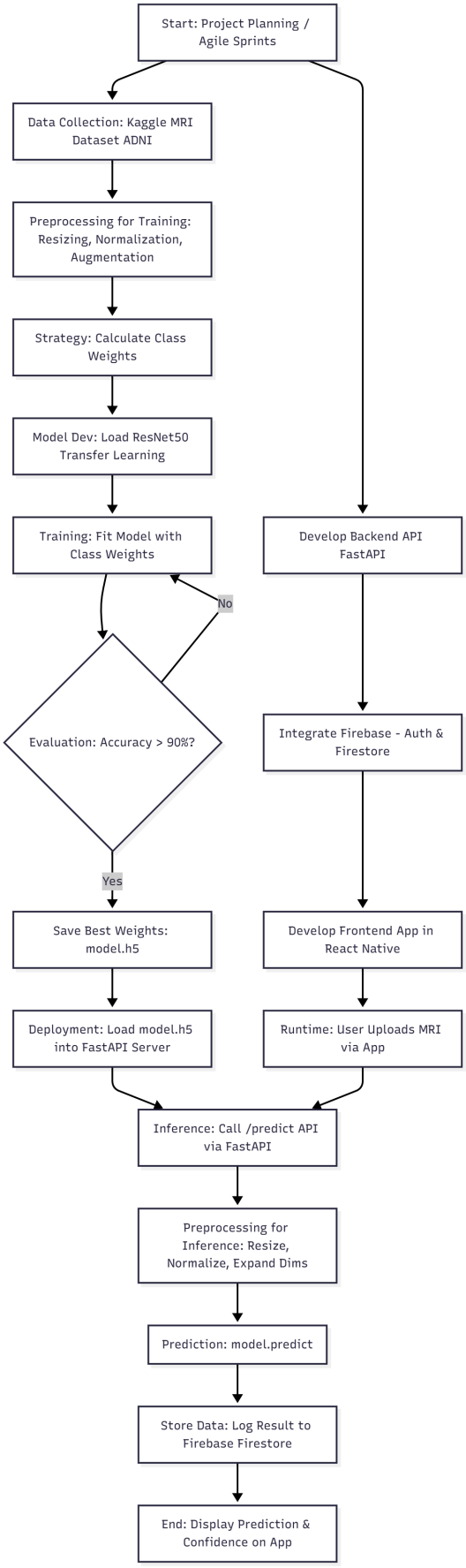

Figure 1.1: Methodology Diagram

1.5 Project Scope

The project aims to incorporate AI-driven Alzheimer's detection into a practical mobile platform. It includes:

- A machine learning backend for MRI classification.
- A React Native-based frontend for user interaction.
- Firebase integration for authentication and real-time cloud database (Firestore).
- A cognitive games module for mental health improvement.
- An admin dashboard for data monitoring.

The system is not a substitution for clinical diagnosis; it is a supportive application for helping with the creation of awareness, screening, and preventive tracking instead of being a certified medical tool.

1.6 Feasibility Study

The project was analysed in terms of technical, economic, and operational feasibility.

- **Technical Feasibility**: The system is developed with TensorFlow/Keras, FastAPI, Firebase, and React Native, all of which can be used effectively on both the local environment and the cloud-based platforms.
- **Economic Feasibility**: The cost of implementation is very low because it uses open-source libraries and free-tier cloud services (Firebase, Google Colab), which can be deployed by using a standard laptop and an internet connection.
- **Operational Feasibility**: The application has a user-friendly interface that can be used by non-technical people and can be managed by an administrator without having a high level of technical skills.

1.7 Solution Application Areas

This system can be applied in multiple domains:

- **Healthcare Research**: To support initial screening of Alzheimer's.
- **Caregiver Use**: To assist families to track and be aware at home.
- **Educational Institutions**: To research AI use in healthcare.
- **Public Health Campaigns**: To assist in the early mental health checkups and awareness.

1.8 Limitations

While the project provides a comprehensive solution for early detection support, several constraints exist which are

- **Clinical Interpretation**: It is a Level 1 screening tool only and its result is not a substitute for professional medical diagnosis.
- **Model Generalization**: CNN Accuracy might be reduced on data obtained from different MRI scanners or protocols.
- **Internet Dependency**: Constant internet is required to send MRI scans to the API and to synchronize real-time data.

1.9 Summary

This chapter provided an overview of Alzheimer's disease, emphasizing its growing global prevalence and the significance of early detection. It introduced the motivation behind developing an AI-assisted, mobile-based system that combines MRI-based prediction with cognitive assessment tools. The objectives, methodology, and scope were defined, describing how deep learning (ResNet50 model in TensorFlow/Keras) and Firebase-backed infrastructure enable classification and user interaction; experimental accuracy is documented in later chapters. A feasibility study confirmed the project's technical, economic, and operational viability. The chapter concluded by identifying the target application areas along with limitations that framed the foundation for the subsequent technical design and development phases.

Chapter 2

Literature Review

This chapter provides a literature review that helps us understand the standards used in the past as well as those followed today. It also offers insight into existing business models and explains how they have evolved over time.

2.1 Alzheimer's Detection in Medical Imaging

MRI, PET and CT are the most common neuroimaging modalities used in studying and diagnosing Alzheimer's disease. MRI is non-invasive with high-resolution structural imaging and no ionizing radiation. PET imaging is very expensive, involves radioactive tracers, and gives only functional data. CT scan is cheap and quick with less soft tissue contrast, limiting its ability in detecting finer neurodegenerative patterns.

MRI is the best of these modalities to use in AI-based Alzheimer's detection systems due to its ability to provide intricate structural changes in the brain, high availability, and safety when used longitudinally [3].

2.1.1 MRI-Based Deep Learning Techniques

Deep learning automates the process of feature extraction from MRI scans. Preprocessing steps such as skull stripping, normalization, and resizing are applied, followed by CNN-based feature learning on 2D slice-based or 3D volumetric formats.

Recent reviews [4] indicate that 3D CNN architectures enhance volumetric representation learning and surpass conventional ML pipelines. Transfer learning (via pretrained models like VGG or ResNet) and data augmentation are typically used to overcome small dataset limitations, increase generalization, and minimize overfitting [5].

Figure 2.1 shows that the MRI images are preprocessed and then fed into a CNN model to extract and classify their features automatically.

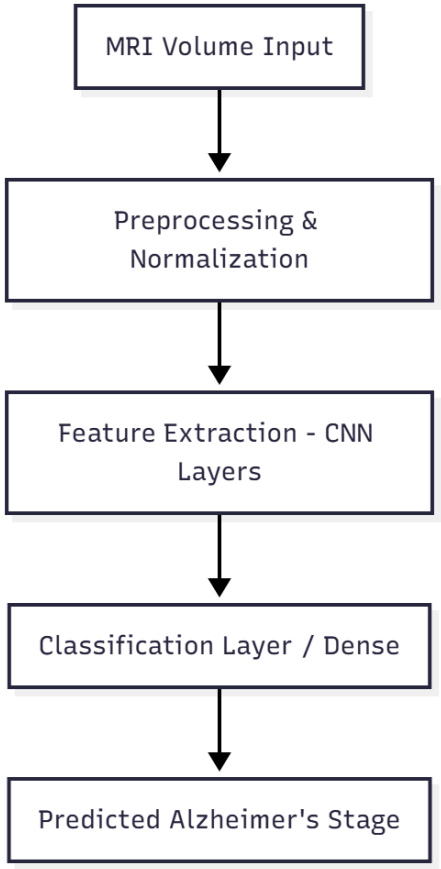

Figure 2.1: Standard MRI-to-CNN Classification Pipeline (Adapted from existing deep-learning studies)

2.1.2 CNN Architectures for Alzheimer's Detection (VGG, ResNet, EfficientNet)

In Alzheimer's Detection using MRI scans, several CNN models have been utilized. The simplicity of VGG16 and VGG19 is advantageous because of the nature of their deep architecture. ResNet-50 has been successfully used on the OASIS and ADNI datasets to categorise Clinical Dementia Rating (CDR) scores [6]. EfficientNet ensures accurate and efficient data with the help of compound scaling [7]. Hybrid CNN-Attention and Vision Transformers (ViT) have also recently achieved potential success in the area of improving feature localization and interpretability [8].

2.1.3 Binary vs. Multi-Class Alzheimer's Classification

Research may be divided into binary (Alzheimer's vs. healthy) and multi-class (Mild, Moderate, Very-Mild) classification. Theoretical accuracies of CNNs trained on ADNI to perform binary tasks are as high as 98% [9]. Multi-class models, although marginally lower (85-94%), are more clinically relevant since they project disease progression stages [10], [11], a feature of the MindCare system.

2.1.4 Performance and Accuracy Comparisons

Accuracies are reported to be different with the datasets, preprocessing and evaluation setup. Hybrid MRI models reached 96.19% on ADNI and 99.82% on NACC datasets [7]; a 3D CNN had 85.12 AUC on early-stage AD [11]. The differences in data size, overlaps of the tests and leakage do not allow direct comparison to be reliable.

2.1.5 Practical vs. Theoretical Performance Gap

High reported accuracies (>98%) may represent a state of perfection; overfitting and generalization errors are known to happen on unknown MRI scans [11]. The MindCare system, which was trained with the Kaggle Alzheimer's MRI data, was able to achieve a realistic, practical measure of performance of nearly 92-93% accuracy when predicting unseen local MRI data.

2.2 Cognitive Assessment and Behavioral Analysis

The early cognitive decline is tested through cognitive games that require the assessment of attention, memory, and problem-solving. Such interventions have been demonstrated to increase cognitive ability in older adults with mild impairment through meta-analysis [12], [13].

AI analyzes behavioral data including reaction time and accuracy to identify insidious cognitive impairment. This enables non-invasive, continuous, and personalized monitoring, combined with neuroimaging, as applied in MindCare [12], [13].

2.3 Comparative Study of Existing Solutions

Most recent systems utilize CNNs in cloud friendly (Flask, FastAPI, Streamlit) systems, with the ability to upload MRI images and provide real-time predictions. The majority of them are prototypes without authentication, databases, or multi-role dashboards [14].

Table 2.1 summarizes a comparative discussion of current solutions, their fundamental models, data sets and constraints.

Table 2.1: Comparative Study of Existing Solutions

System / Project	**Core Model**	**Dataset Used**	**Accuracy (%)**	**Features**	**Limitations / Reference**
Alzheimer's-Disease-Detection (GitHub)	CNN	ADNI	94	MRI classification only	Prototype script — no UI or database [6]
MRI-Alzheimer-Classifier	ResNet-50 + Grad-CAM	OASIS	96	Explainable predictions	Model-only implementation [6, 9]
Cloud MRI Prediction Demo	Custom CNN	ADNI	95	Web inference demo	Lacks authentication and role management [14]
Streamlit Image Predictor	Simple CNN	Local dataset	90	Upload + predict in browser	No persistent storage [14]
Cognitive Game Platforms	Behavioral AI	Game datasets	—	Game-based cognitive testing	No MRI integration [12, 13]
MindCare (Proposed)	ResNet-50 (Kaggle MRI)	Kaggle Dataset	91	MRI + Cognitive + Admin modules	Full-stack deployable solution [10, 11, 14]

MindCare is the first system to integrate MRI-based deep learning, cognitive analysis, and admin control in a deployable and user-friendly system.

2.4 Summary

This chapter has been a review of AI-based Alzheimer's detection and cognitive health assessment. MRI was described as the most appropriate modality among other scans such as PET and CT. CNNs rchitectures such as ResNet, VGG, and EfficientNet were described, with the focus on multi-class classification of the disease stage. As a complement to other tools, cognitive game-driven testing and AI behavioral monitoring were reviewed. Prior research gaps were evaluated and MindCare was introduced as a system that allows integrating MRI prediction, cognitive games, and analytics on the administration into a synthesized and deployable system.

Chapter 3

Requirement Specifications

System requirements include those configurations that a system needs in order to function properly and effectively. Failure to meet with these requirements may cause installation issues or performance issues. System requirements outline the features and behaviour of a system, including both functional and non-functional requirements.

3.1 Existing System

Existing systems designed to detect Alzheimer's are highly restricted to a single ML model or simple web prototypes. Models are mostly binary (Alzheimer's vs. normal) with no UI. The CNN-based MRI classification scripts that are open-source tend to offer no data storage, role management or mobile interfaces [3], [5]. There are also no interactive cognitive testing tools available that can be used to trace mental functioning other than MRI. Models with high accuracy (>98%) tend to be poor at real-world data because small high-quality training sets such as ADNI or OASIS are used [11]. These restrictions demonstrate the necessity of a realistic system that would involve a combination of AI-based MRI prediction, cognitive tests, progress tracking, and mobile accessibility.

3.2 Proposed System

MindCare is a mobile-based platform that integrates AI-based MRI prediction, cognitive assessment games, and an administrative monitoring platform.

- **User Module:** Upload MRI scans, receive stage forecasts, play interactive games, and track mental performance.
- **Admin Module:** Manage system, view user activity, track prediction and game histories, analyze performance, and suggest daily exercises.

- **Cognitive Games & Prevention Module:** Play interactive games which evaluate attention, memory and problem solving, offer adaptive preventive strategies based on MRI and cognitive outcomes.

3.3 Functional Requirements

The functional requirements outline the essence of behaviour and functionality of the MindCare system. Each requirement is established to make sure that the system achieves the intended role, i.e., to help the users in the early diagnosis of Alzheimer's disease as well as observing their cognitive ability over time.

3.3.1 User Authentication

The system ought to enable new users to sign up and existing users to log in through Firebase Authentication. The passwords are also securely stored and password recovery is via registered email addresses. Only the verified users access the MRI upload and cognitive modules.

3.3.2 MRI Upload and Prediction

MRI images are uploaded by users through their device storage or camera. The preprocessing and prediction of images are carried out through FastAPI and a ResNet50 model. The stage of predicted Alzheimer's (Non-Demented, Very Mild Demented, Mild Demented, Moderate Demented) and score on confidence are produced.

3.3.3 Cognitive Games

Four cognitive games are used to measure memory, attention span, reaction time and pattern recognition. The outcomes of the games are sent to Firebase and cross-analyzed with the MRI predictions to analyze the health of the brain, identify cognitive alterations, and offer individual preventive feedback.

3.3.4 Personalized Insights

The predictions of MRI and cognitive scores are added to create a general brain-health score. Users can be classified as low, medium, and high risk (MRI 60% Cognitive 40%), and lifestyle, memory training, and mental wellness routines are suggested.

3.3.5 History and Analytics

A user can view past MRI predictions and cognitive scores via an analytics dashboard which is a combination of MRI and cognitive data, displaying both brain-health patterns and risk level over time.

3.3.6 Admin Control

Admins can access user profiles, MRI and cognitive histories, monitor prediction frequency, examine overall performance and erase inactive/duplicate users. MRI results and user data cannot be changed by admins.

3.4 Non-Functional Requirements

3.4.1 Performance

Within 4–5 seconds of uploading an MRI image, the system must deliver predictions, and the response time must remain consistent when the system is operating under normal user load.

3.4.2 Security

The entire communication between the app and the backend should be encrypted by using HTTPS. Firebase Authentication provides secure logins and no sensitive data is stored locally in the device.

3.4.3 Reliability

The real-time synchronization of Firebase ensures that data is available offline even in instances of network disruption. When an upload fails, the system will automatically restart it until it succeeds.

3.4.4 Usability

The interface should be clean and user-friendly, suitable for the aged users who may not have any technical expertise. The font size and color scheme must be made reader-friendly.

3.4.5 Scalability

The backend architecture can easily be extended to additional servers or cloud applications as the number of users increases.

3.4.6 Maintainability

The system should have a modular architecture, where the FastAPI backend and React Native frontend can be updated independently, and the Firebase provides centralized data to improve the maintenance.

3.4.7 Portability

The application is developed for Android 8.0 and higher platform but with minor changes the subsequent versions can be made to work on the iOS environment.

3.5 Hardware Requirements

The hardware specifications that are necessary to deploy and test the proposed system are summarized in Table 3.1.

Table 3.1: Hardware requirements

Component	**Specification (Minimum)**
Processor	Intel Core i5 (or equivalent ARM)
RAM	8 GB
Storage	2 GB free space
GPU (optional)	NVIDIA GTX 1050 or higher for training (not for inference)
Mobile Device	Android smartphone (4 GB RAM, 64-bit processor)

3.6 Software Requirements

The software tools and frameworks used in the system development and deployment are described in Table 3.2.

Table 3.2: Software requirements

Software	**Purpose / Usage**
Python 3.10+	Model training, backend integration
TensorFlow / Keras	Deep learning model development
FastAPI	Web backend for model API
Firebase (Auth + Firestore)	Authentication and database
React Native (Expo CLI)	Mobile frontend development
VS Code / Android Studio	Development environment
Google Colab	Model training and experimentation

3.7 Use Cases

MindCare system involves two major actors: User and Admin. They all engage with the system with different modules which are connected in a way to fulfill particular goals.

- **User:** general users or patients that interact with the mobile-based application by signing in, uploading MRI scans, playing cognitive games, viewing past performance, getting insights, and tracking cognitive improvement.
- **Admin:** system overseers or authorized users who can view user activity, prediction trends, performance reports. However, the admin cannot change the MRI predictions or game results to keep data and system integral.

3.7.1 Use Case Diagram

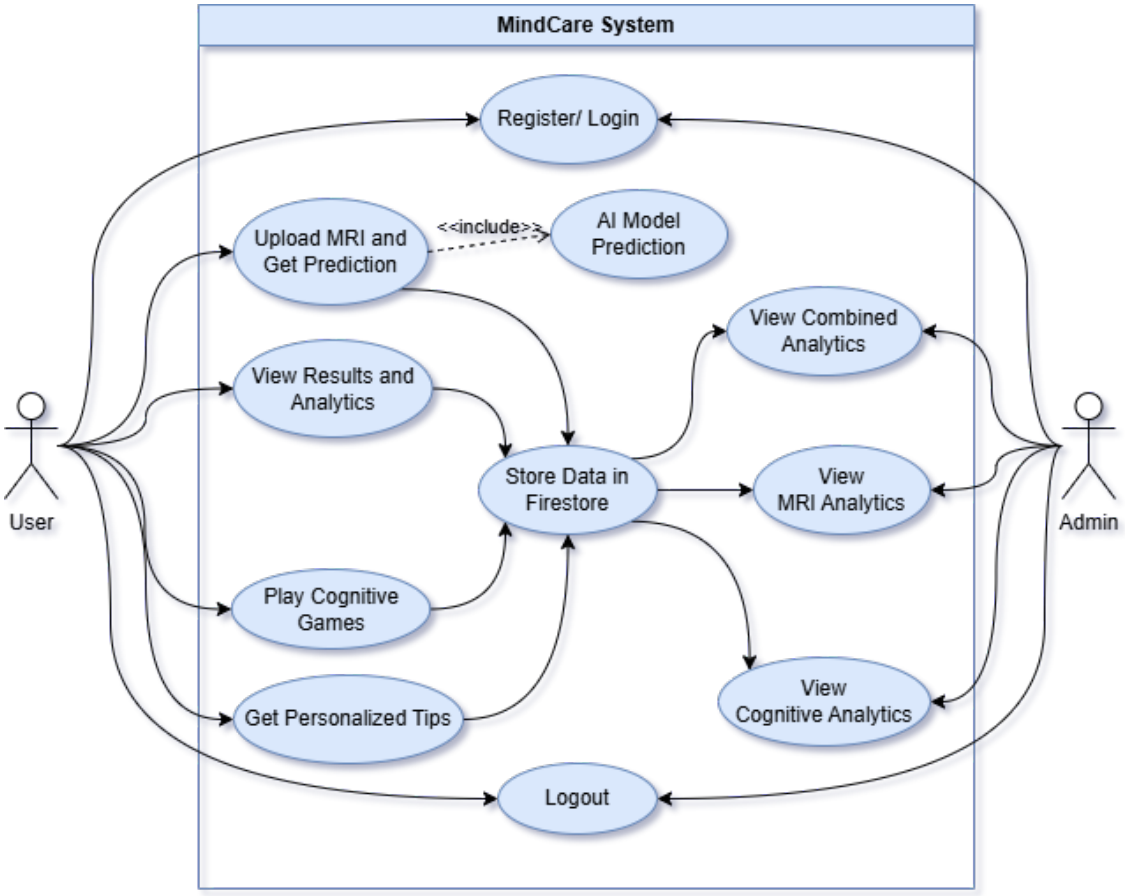

Figure 3.1: Use Case Diagram

As shown in Figure 3.1, the MindCare system specifies two main actors, namely User and Admin, that engage with different system modules depending on their roles and access permissions.

3.7.2 Use Case Descriptions

The high-level use cases of the MindCare system are summarized in Table 3.3 indicating the actors and their corresponding functionality in the system.

Table 3.3: Use case descriptions

Use Case ID	**Actor**	**Use Case Name**	**Description**
UC-01	User	User Authentication	The user registers or logs-in using email and password through Firebase Authentication.
UC-02	User	MRI Upload and Prediction	The user posts an MRI image that is processed on the backend using the trained ResNet50 model to determine the stage of Alzheimer's.
UC-03	User	Cognitive Game Interaction	The individual takes memory and reaction tests. Firebase is used to store game performance data to track progress.
UC-04	User	View Results and Progress	The user sees past MRI outcomes, game scores and progress of improvement in a customized dashboard.
UC-05	Admin	Monitor System analytics and Users	The admin keeps track of all the activity of all users, including MRI uploads and game participation, via the admin dashboard.
UC-06	User	View Personalized Insights & Recommendations	Uses MRI findings and cognitive-game scores to calculate a general score of brain-health and risk-level and shows individual tips and prevention measures.

3.7.3 Use Case Specification

In this subsection, each of the identified use cases has been specified in detail, describing the system behavior, actor interaction, preconditions, and expected outcomes. These defined requirements provide clarity in functional requirements and act as a guideline in system implementation and validation.

UC-01: User Authentication

The authentication process, comprising of user login, user registration and credential validation, is outlined in Table 3.4.

Table 3.4: UC-01: User Authentication

Use Case ID	UC-01
Use Case Name	User Authentication
Primary Actor	User
Trigger	User opens the app and chooses either “Login” or “Sign up.”
Description	Allows a new user to create an account or an existing user to log in with Firebase Authentication by means of a valid email and password.
Preconditions	The app is running and user is connected to the internet.
Postconditions	System authenticates credentials and provides access to the dashboard.
Main Flow	1. User opens the MindCare app. 2. Chooses to log in or register. 3. Enters email and password. 4. Firebase verifies credentials. 5. Dashboard loads on success.
Alternate Flow(s)	1. In case of invalid credentials, an error message is displayed. 2. In case of forgotten password, user is redirected to password recovery page.
Exceptions	Firebase service not reachable or device offline.
Frequency of Use	Frequent (every app access).
Priority	High

UC-02: MRI Upload and Prediction

The process of MRI upload and automated prediction with the help of the backend AI model is outlined in Table 3.5.

Table 3.5: UC-02: MRI Upload and Prediction

Use Case ID	UC-02
Use Case Name	MRI Upload and Prediction
Primary Actor	User
Trigger	User selects “Upload MRI” in the MRI prediction tab.
Description	Enables the user to upload an MRI scan, which is handled by the backend FastAPI and inputted into the AI model (ResNet50) to draw an inference on the stage of Alzheimer’s.
Preconditions	User logged in, has camera/gallery access, backend and model activated.
Postconditions	The confidence score and the predicted stage shown and saved in Firebase.
Main Flow	1. User clicks “Upload MRI.” 2. Selects or captures MRI image. 3. Backend receives and preprocesses image. 4. Model performs prediction. 5. Predicted stage appears on screen.
Alternate Flow(s)	1. If file format is invalid, then an error prompt is generated. 2. If an upload fails, then retry option is displayed.
Exceptions	Backend/API unavailable.
Frequency of Use	Moderate (occasional).
Priority	High

UC-03: Cognitive Game Interaction

Table 3.6 describes how users and cognitive games interact with each other, such as gameplay and storing the score.

Table 3.6: UC-03: Cognitive Game Interaction

Use Case ID	UC-03
Use Case Name	Cognitive Game Interaction
Primary Actor	User
Trigger	User chooses a game from the “Cognitive Games” tab menu.
Description	Allows the user to engage in interactive games that evaluate memory, attention as well as pattern recognition. The scores are saved in Firebase.
Preconditions	User logged in and connected to the internet.
Postconditions	Game results saved in Firebase for future analysis.
Main Flow	1. User selects a game. 2. System loads instructions. 3. User plays the game. 4. Scores are calculated and uploaded. 5. Confirmation message displayed.
Alternate Flow(s)	In case of network failure, the data is stored locally and synced later.
Exceptions	Game crashes or incomplete session.
Frequency of Use	Regular (daily/weekly).
Priority	Medium

UC-04: View Results and Progress

Table 3.7 shows how the users can review the historical results and performance trends.

Table 3.7: UC-04: View Results and Progress

Use Case ID	UC-04
Use Case Name	View Results and Progress
Primary Actor	User
Trigger	User taps on "Analytics" tab.
Description	Allows the user to view previous MRI results, game scores, and trend analytics.
Preconditions	User logged in; data exists in Firebase.
Postconditions	Results displayed successfully on screen.
Main Flow	1. User opens analytics page. 2. App fetches historical data from Firebase. 3. System renders trend charts and summaries. 4. User reviews insights.
Alternate Flow(s)	If no data is available, then "No results yet" message is displayed.
Exceptions	Network connection lost during fetch.
Frequency of Use	Frequent.
Priority	Medium

UC-05: Monitor System analytics and Users

Table 3.8 indicates the admin functions to monitor system activity and user analytics.

Table 3.8: UC-05: Monitor System Analytics and Users

Use Case ID	UC-05
Use Case Name	Monitor System analytics and Users
Primary Actor	Admin
Trigger	Admin logs into the admin dashboard.
Description	Allows the administrator to view user actions, MRI upload reports and the number of games played via a central dashboard.
Preconditions	Admin must be authenticated.
Postconditions	Dashboard reflects real time and historical data analytics.
Main Flow	1. Admin opens dashboard. 2. System retrieves user metrics. 3. Admin views summaries and statistics. 4. Logs out when done.
Alternate Flow(s)	In case of data retrieval error, retry option is displayed.
Exceptions	Database unavailable.
Frequency of Use	Occasional (weekly/monthly).
Priority	High

UC-06: View Personalized Insights & Recommendations

Table 3.9 explains how combined analysis is used to generate personalized insights and preventive recommendations.

Table 3.9: UC-06: View Personalized Insights

Use Case ID	UC-06
Use Case Name	View Personalized Insights & Recommendations
Primary Actor	User
Trigger	User opens the "Prevention Tips" tab.
Description	The system retrieves results of MRI and cognitive-games, calculates a joint score of brain-health by using weighted reasoning, classifies the user into a risk category and provides adaptive tips and suggestions.
Preconditions	There must be MRI and cognitive-game information of the logged-in user.
Postconditions	Combined score and recommendations displayed; revised risk level in the user analytics.
Main Flow	1. User opens Tips screen. 2. App fetches MRI and cognitive scores. 3. System computes weighted combined score. 4. Risk level determined. 5. Relevant tips displayed.
Alternate Flow(s)	If no data is present, it shows a message "Complete MRI and games to view insights."
Exceptions	Firebase fetch failure or network loss.
Frequency of Use	Regular (after each MRI upload or game session).
Priority	High

3.7.4 Use Case Flow

- The User begins by registering and logging in the mobile app.
- Once they have been authenticated, they can post MRI scans that are processed with the back-end AI model.
- The predicted stage is displayed, and the results are stored to be accessed later.
- The User can also play cognitive games. The results are automatically saved in Firestore.
- The User can then view combined MRI and cognitive analysis in the Insights screen, receiving adaptive prevention recommendations.
- The Admin monitors system usage through the admin dashboard, and can delete any idle or duplicate accounts through the User Management screen

This model offers a separation of roles, access control and a balanced, transparent workflow.

3.8 Summary

This chapter summarizes the functional and non-functional requirements, and general architecture of the proposed MindCare application. It started by comparing the currently available tools to detect Alzheimer's with the proposed solution and defining their shortcomings in interactivity, real-time monitoring, and accessibility. The MindCare system was subsequently specified with specific functional and non-functional requirements, including MRI upload, AI-driven prediction, cognitive gaming, joint brain-health analytics, personalized dynamic recommendation, and administrative ability including monitoring and restricted user pefrformance. Hardware and software requirement was described as well as scalability, usability, and security. Lastly, the use case models depicted user and admin interactions as well as workflows, which were transparent, practical, and efficient in the operation of the system. This chapter provided the foundation upon which the design and implementation, as discussed in the subsequent chapters, will be carried out.

Chapter 4

Design

The system design is the transformation of user requirements into a practical shape. In this chapter, the design approaches and constraints are discussed in detail. While designing our MindCare app, we tried our best to use the performance standards of the software industry and fulfil the needs of its user.

4.1 System Architecture

MindCare system architecture is built around a modular, cloud-connected architecture which enables the React Native mobile client, FastAPI backend, TensorFlow/Keras ResNet-50 model and Firebase to communicate in real time. This design ensures scalability, maintenance, and secure data processing.

The architecture of the system aims to achieve accuracy and reliability through the use of a well-tuned ResNet-50 model to classify stages of the Alzheimer disease. The Firebase cloud services provide scalability, therefore supporting multiple users at a time. The separation of concerns is ensured by the isolation of the user interface, backend logic, AI inference and database layers. Firestore listeners are used to support real-time synchronization to deliver immediate analytics updates. Firebase Authentication and encrypted communication based on HTTPS are implemented to provide security.

The general system architecture is presented in Figure 4.1, showing the interaction between the mobile application, backend services, AI model, and Firebase cloud elements.

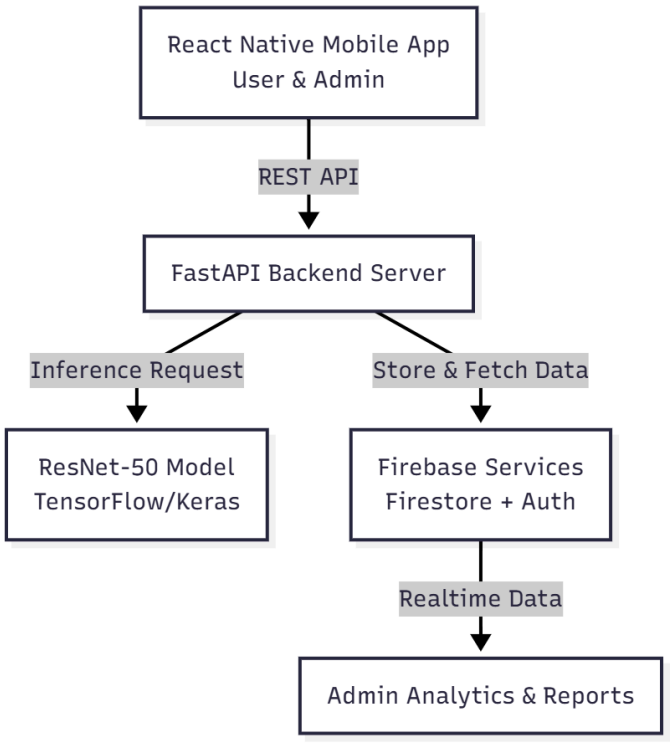

Figure 4.1: System Architecture Diagram

4.1.1 System Workflow

The mobile client interacts with the FastAPI backend via the REST API calls. A user uploads a MRI which is sent to FastAPI backend server for preprocessing and then to ResNet-50 model to perform inference. The predicted stage along with the confidence score are returned and stored in Firebase Firestore. Firebase also processes cognitive-game data, user activity and analytics. Aggregated data is displayed in the admin dashboard.

Figure 4.2 illustrates the end to end operational flow of the system starting with MRI upload all the way to the analytics visualization.

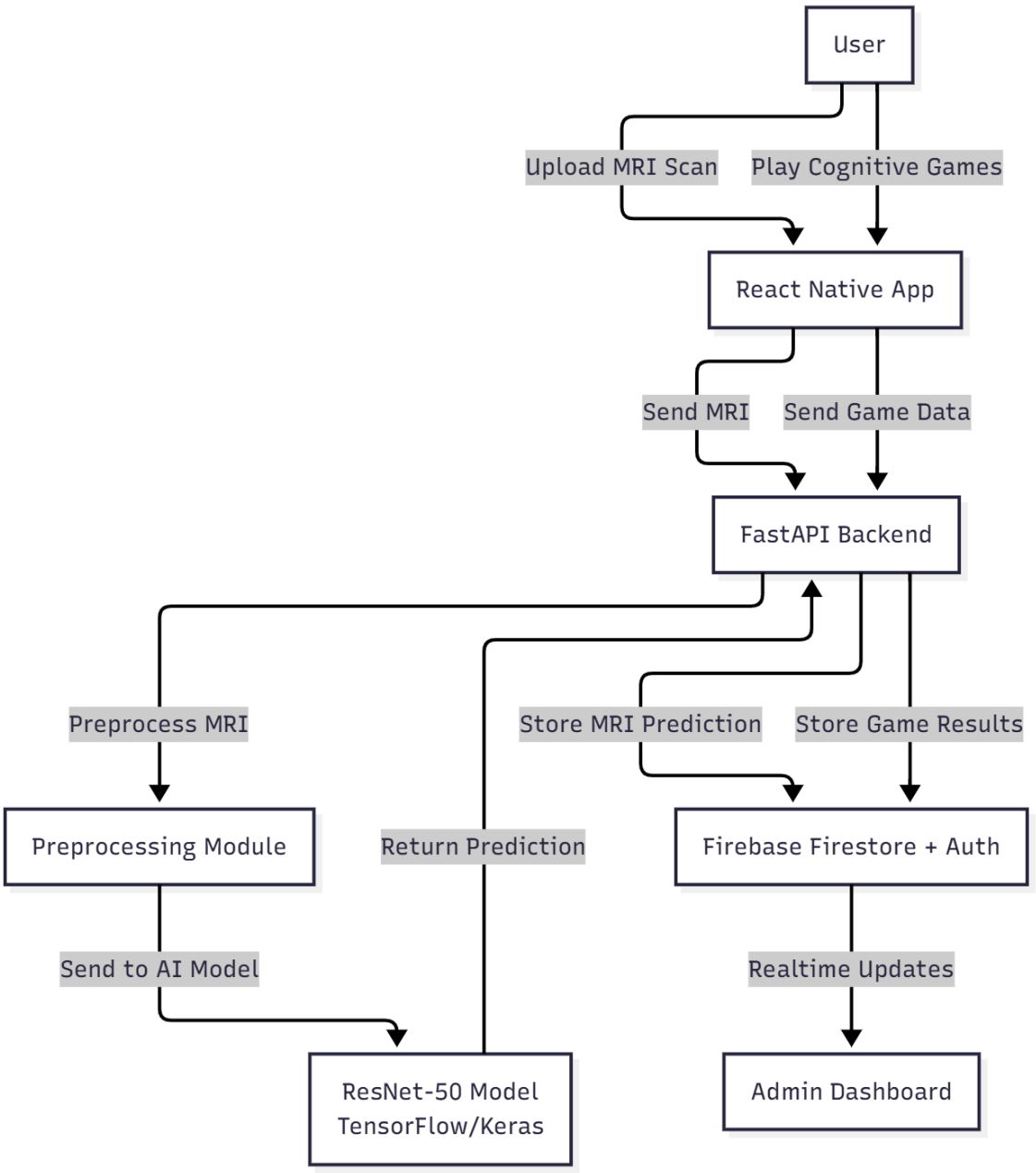

Figure 4.2: System Workflow Diagram

4.1.2 System Components

The following subsection outlines the key functional components of the MindCare system and elaborates on their respective roles within the entire architecture. Each component is independent, and works together with the rest to ensure smooth data flow and system reliability.

1. **Frontend Module**
   The frontend is built in React Native that provides user and admin dashboards. User dashboard displays MRI analysis, cognitive games, personalized tips, and analytics whereas, Admin dashboard displays user management, mri and cognitive analytics and system reports. The interface is accessibile with responsive layouts and uniform navigation.

2. **Backend Module**
   The backend is built in FastAPI (python) that specifies API endpoints, loads the trained ResNet-50 model on the start-up, performs predictions and gives formatted JSON response. It provides synchronized data sharing with Firebase.

3. **Machine Learning Model**
   The ResNet-50 model (TensorFlow/Keras) can classify MRI scans into these four classes (Non-Demented, Very Mild Demented, Mild Demented, and Moderate Demented). Before inference, an input MRI is resized to 128x128 pixels. The model is stored at alz-backend/model/model.h5.

4. **Database and Cloud Services**
   Firebase Firestore as NoSQL database contains user profiles, MRI predictions, cognitive-game sessions, and analytics information whereas, Firebase Auth is used to secure logins, and maintain real time synchronization.

4.2 Design Constraints

To achieve a tradeoff between performance, cost and scalability, several design choices have been undertaken. Firebase has been chosen because it is easy to integrate and has real-time synchronization. Local inference makes deployment easier but throughput is reduced. ResNet-50 was selected as it has high accuracy and fast inference time, whereas more deep networks like ResNet-101 were discarded because of complexity and unavailability of huge datasets. Mobile-first strategy was emphasized based on the assumption that users would mainly use smartphones to access the system. FastAPI was chosen for better performance and asynchronous handling. It also works perfectly with the TensorFlow/Keras model for MRI inference. React Native was chosen for faster development, strong library support, easier maintenance, and smooth Firebase integration.

Table 4.1 summarizes the major design constraints that affect the performance, scalability, security and usability.

Table 4.1: Design Constraints

Constraint	**Points / Description**
Hardware and Performance	1. Backend inference requires at least 8 GB RAM and Intel i5 (or equivalent). 2. React Native app performs well on Android 8.0+ with minimum 2 GB RAM. 3. Each MRI prediction takes ∼4–5 seconds. 4. Asynchronous processing prevents frontend freezing.
Software	1. Backend Framework: FastAPI (Python 3.10+). 2. ML Framework: TensorFlow 2.x with Keras API. 3. Frontend Framework: React Native (TypeScript). 4. Database & Cloud Services: Firebase (Firestore + Auth). 5. All modules must be compatible for seamless operation.
Resource Limitations	1. Free-tier Firebase limits daily reads/writes. 2. Local inference constrains prediction speed. 3. Large-scale deployment requires GPU or cloud server.
Security	1. Firebase Authentication used for email-based login. 2. All API calls use HTTPS. 3. Firestore rules enforce role-based access. 4. Only MRI prediction metadata stored; sensitive medical data not saved.
Usability and Interface	1. App supports portrait orientation and mid-range mobile resolutions. 2. Firebase-dependent operations require stable internet. 3. Color schemes and font sizes optimized for readability. 4. Full WCAG compliance pending.

4.3 Design Methodology

This section describes the development strategy utilized in the development of the MindCare system based on iterative design, modular implementation and continuous assessment across the entire development lifecycle.

4.3.1 Agile Methodology

MindCare System is based on an Agile, iterative approach, which enables incremental development and constant testing. The modules (frontend, backend, AI model, and database) were created as independent components allowing rapid development prototyping, early integration, and an easy refinement process, which relies on testing and user feedback. This design provides scalability, flexibility and maintainability during development.

Figure 4.3 shows the cyclical process of planning, implementation, testing, refinement and integration used during the development of MindCare System.

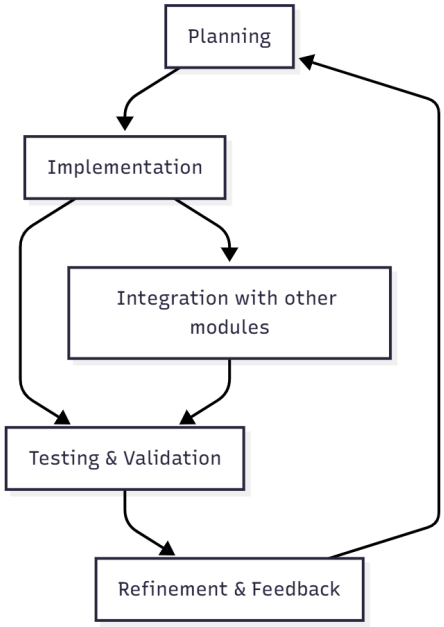

Figure 4.3: MindCare System Development Cycle

4.4 High-Level Design

High-level design reflects the system in a logical perspective, showing how significant components relate to each other to achieve the system goals.

4.4.1 Conceptual / Logical View

MindCare System consists of these three main modules: User, Admin, and Shared Cloud Services modules. All the modules communicate with each other using REST APIs and Firebase SDKs.

- The User Module deals with MRI uploads, cognitive games, personalized tips and personal analytics.
- The Admin Module handles user administration, system usage, and analyzes compiled system data.
- The Shared Cloud Services layer supports the two modules, as it deals with authentication, database storage and real-time updates.

The conceptual component view of the MindCare system as depicted in Figure 4.4 shows the connection between the user module, the admin module, and the shared cloud services.

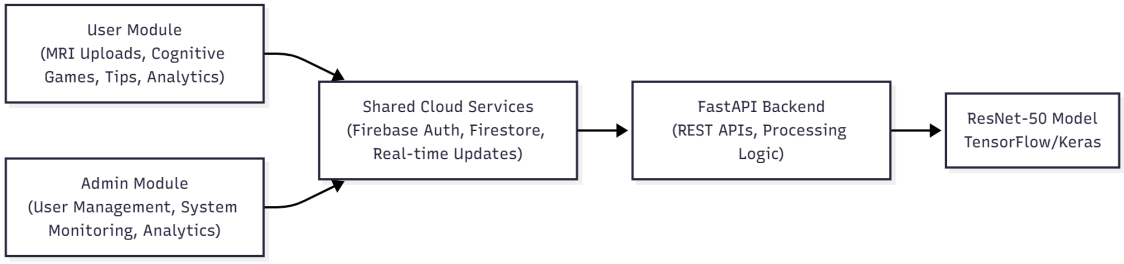

Figure 4.4: Component Diagram

4.4.2 Process View

The process view explains the steps involved in the interaction between the mobile app, backend, AI model, and Firebase during real-time performance.

1. The Firebase Authentication allows the user to authenticate.
2. After a successful login, an MRI image can be uploaded in the app which is sent to the FastAPI server where it is preprocessed and predicted.
3. The backend will call the ResNet-50 model, which will provide the stage of Alzheimer and confidence score and stored in Firestore.
4. The interface requests the latest information and shows it to the user in real time.

Figure 4.5 shows the chain of interactions between the mobile application, backend services, the AI model, and Firebase.

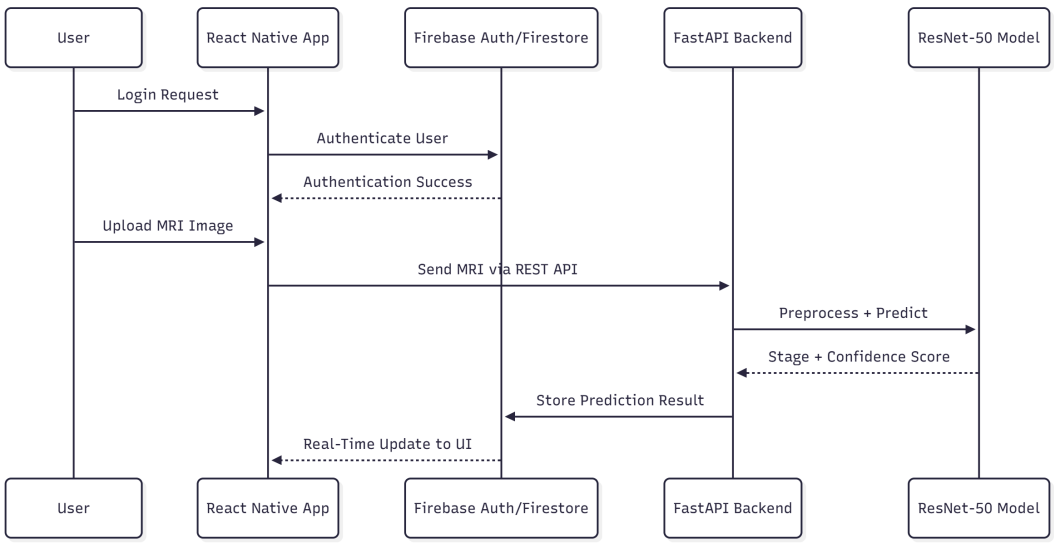

Figure 4.5: Process View Diagram

4.4.3 Physical View

The physical perspective provides an overview of how the system is being configured across the devices and servers. The system applies the hybrid architecture of a local backend server and cloud-based Firebase services.

Deployment Environment:

- **Frontend:** Deploys on an Android device via React Native.
- **Backend:** Runs locally using FastAPI.
- **Model File:** Loaded from model/model.h5 during backend initialization.
- **Database and Storage:** Firebase cloud services

Figure 4.6 below reflects the physical deployment perspective of the MindCare system, where the system components are distributed between the mobile device, a local backend server, and Firebase cloud-based services.

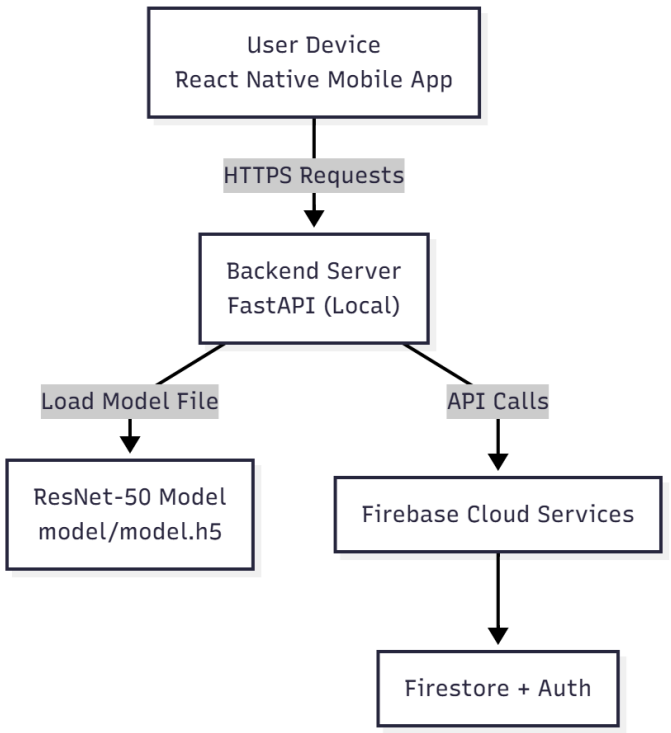

Figure 4.6: Physical/Deployment Diagram

4.4.4 Module View

The system is broken down into structured modules so that it is maintainable and clear. Each of the modules represents a major functional area in the architecture.

1. **User Module:** Deals with authentication, MRI upload, access to games, and personal analytics. Core screens include:
    - UserDashboardScreen.tsx
    - MRIPredictionScreen.tsx
    - CognitiveGamesScreen.tsx
    - UserAnalyticsScreen.tsx
    - PersonalizedTipsScreen.tsx
2. **Admin Module:** Enables the administrator to track users, analyze MRI results and system reports. Core screens include:
    - AdminDashboardScreen.tsx
    - MRIAnalyticsScreen.tsx
    - CognitiveAnalyticsScreen.tsx
    - UserManagementScreen.tsx
    - SystemReportsScreen.tsx
3. **Backend Module:** Calls API requests, makes predictions based on the ResNet-50 model and communicates with Firebase to store and retrieve data. Files include:
    - app.py
    - routes/predict.py
    - model/model.h5
4. **Database Module:** Consists of Firestore collections:
    - users
    - mri_predictions
    - game_sessions
    - user_cognitive_profiles
    - cognitive_assessments
5. **AI Model Module:** Consists of the TensorFlow/Keras-implemented ResNet-50 architecture to perform MRI classification. It gives the label prediction and probability scores.

4.4.5 Security View

- **Authentication and Authorization:** Firebase Authentication, Firestore rules
- **Data Transmission Security:** HTTPS encryption
- **Access Control:** Firestore security rules
- **Model Security:** Model file stored locally
- **Data Privacy:** Only prediction metadata stored

4.5 Low-Level Design

The low level design explains how the MindCare system is internally structured and how the various components process data, communicate with others and provide reliable behaviour. It is concerned with the backend logic, end user interactions, database management and information flow between the user input and the final analytics and recommendations.

4.5.1 Detailed Design of Major Components

1. **Backend Component – FastAPI and Model Integration:** The backend is developed in FastAPI in Python that receives the requests sent by the mobile app, loads the ResNet-50 model, makes MRI predictions, and sends the results in JSON format. The working can be summarized as follows: FastApi server accepts an image file via a POST request in the predict endpoint. The image is temporarily saved in some local directory then preprocessed by a route predict.py preprocessing function that resizes and normalizes the image to 128128 pixels. The loaded ResNet-50 model takes the preprocessed image and infers it. The probabilities of the output are transformed into a confidence score and predicted class label. The output is sent in a JSON format and stored in Firebase at the same time.

   Figure 4.7 shows the backend MRI prediction flow.

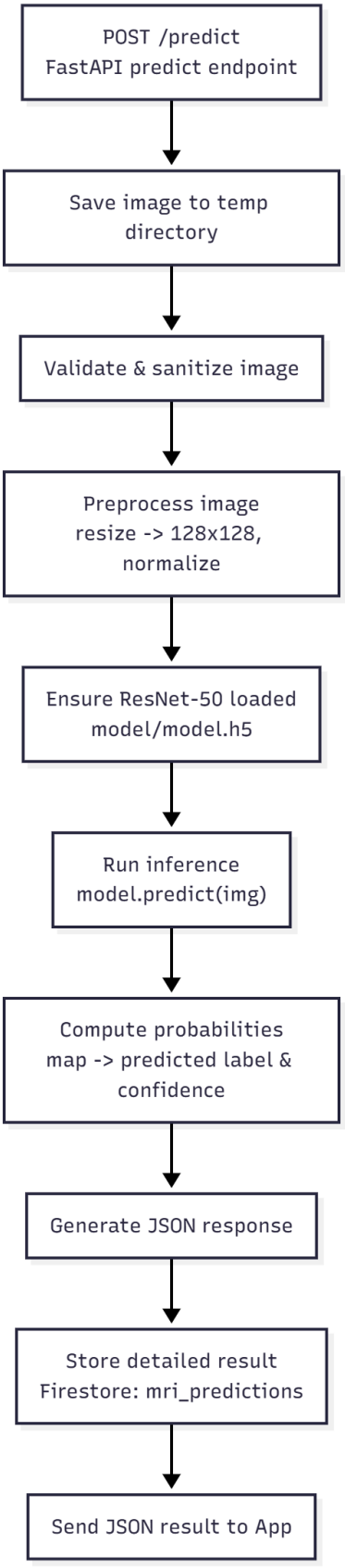

Figure 4.7: Backend MRI Prediction Flow

2. **Frontend Component – React Native Application:** The mobile application handles all user and admin interactions. All screen components are written in TypeScript, and have a component-based maintainability structure. The MRIPredictionScreen.tsx component deals with MRI upload, prediction request, and display of results. The CognitiveGamesScreen.tsx component consists of logic of cognitive games like Memory Match, Reaction Timer, Spatial Recognition and Pattern Recognition. The UserAnalyticsScreen.tsx component loads and visualizes the performance of the user based on Firebase queries. The PersonalizedTipsScreen.tsx module obtains adaptive cognitive health and lifestyle recommendations on the basis of combined MRI and game analytics. Admin Dashboard is based on MRIAnalyticsScreen.tsx and CognitiveAnalyticsScreen.tsx that aggregate system-wide data.

   Figure 4.8 shows the frontend data handling structure.

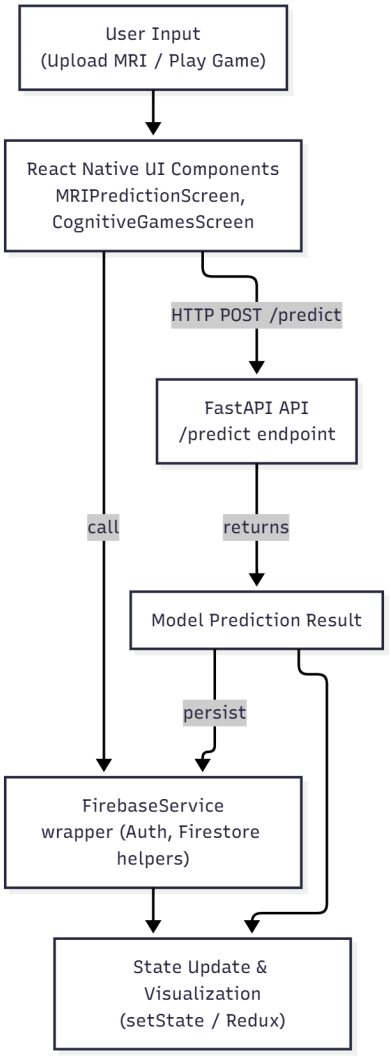

Figure 4.8: Frontend Data Handling Structure

3. **Firebase Component – Database and Auth Integration:** Firebase offers NoSQL Firestore database, Authentication and Cloud storage to manage all persistent data. Each collection is a major element of the system. users stores scan summaries, registration and authentication. mri_predictions: stores detailed MRI prediction outcomes, including predicted stage, confidence, probabilities, inference time, and metadata. game_sessions: monitors cognitive game activity, including scores, accuracy, duration, and timestamps. cognitive_assessments: makes cumulative cognitive reports at least 5 game sessions, storing domain-wise scores, overall scores, trends, and recommendations. user_cognitive_profiles: retains life cognitive profiles, summarizing MRI and cognitive game insights, strengths, areas for improvement, and historical data.

   Figure 4.9 shows the Firebase integration flow.

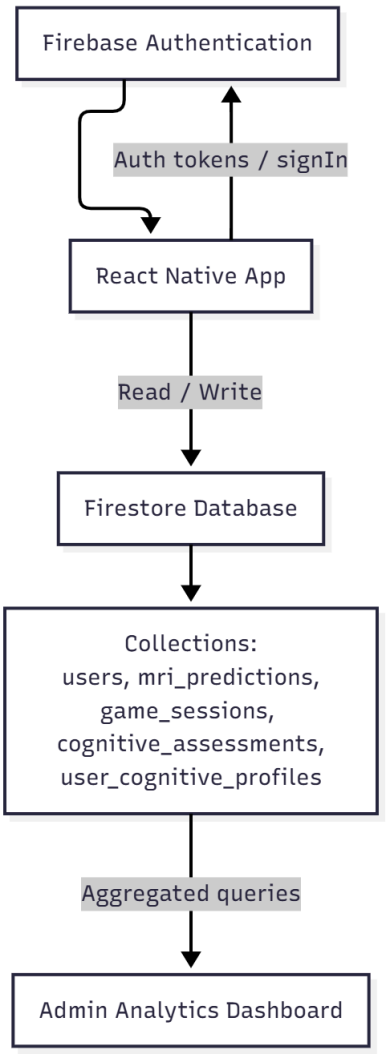

Figure 4.9: Firebase Integration Flow

4.5.2 UML Class Diagram

Figure 4.10 the UML class diagram of the system containing relationships and responsibilities of key classes.

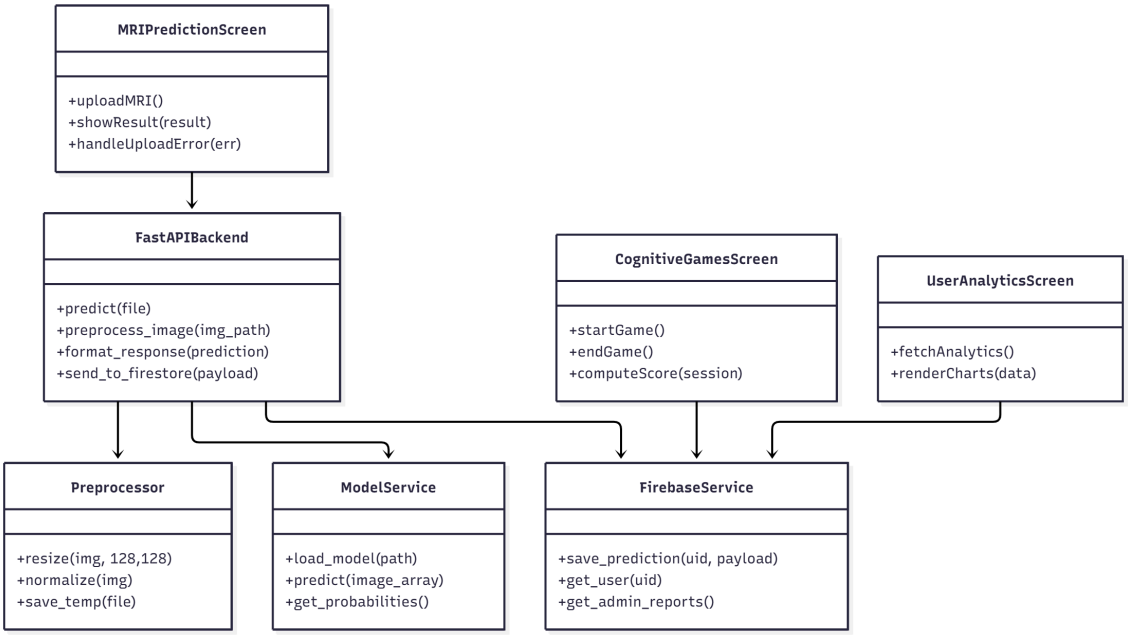

Figure 4.10: UML Class Diagram

4.5.3 Sequence Diagram

Figure 4.11 MRI prediction sequence diagram illustrating the sequential interaction between mobile frontend, backend and database.

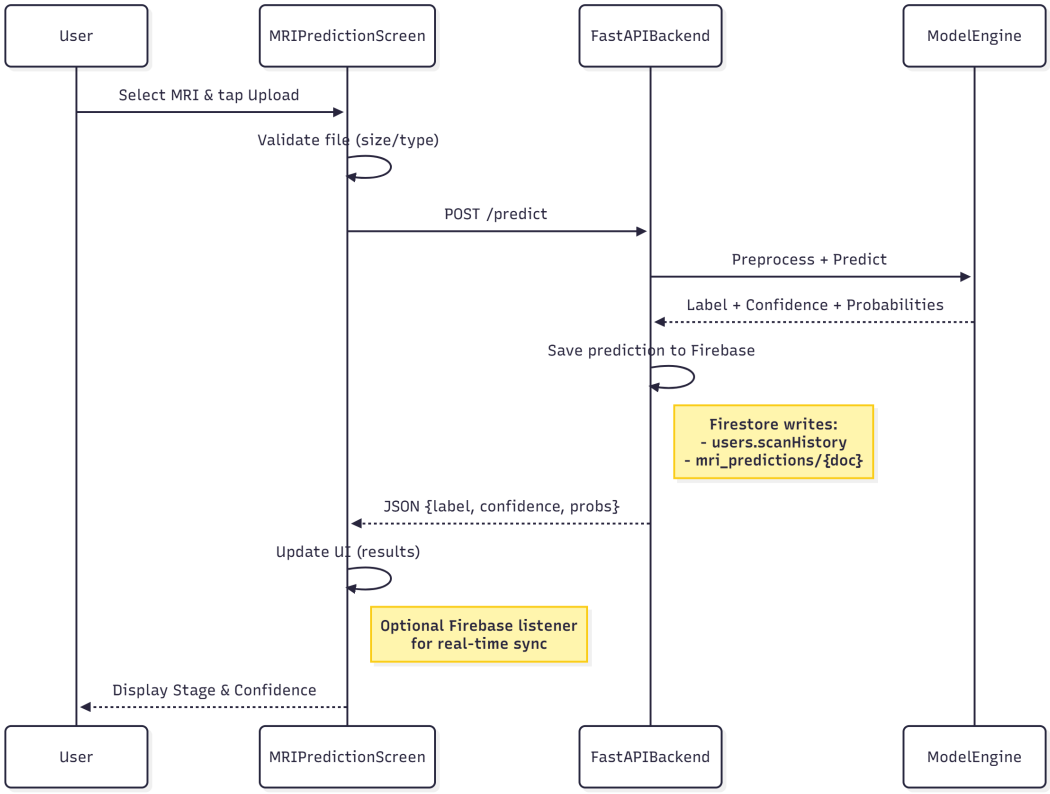

Figure 4.11: MRI Prediction Sequence Diagram

4.5.4 Flowcharts for Core Functions

The following flowcharts demonstrate the internal logic of the core functionalities of the system in steps. They offer a clear visual representation of the data processing and decision-making process in each of the main modules.

1. MRI Prediction Workflow:
   Figure 4.12 the MRI Prediction Workflow.

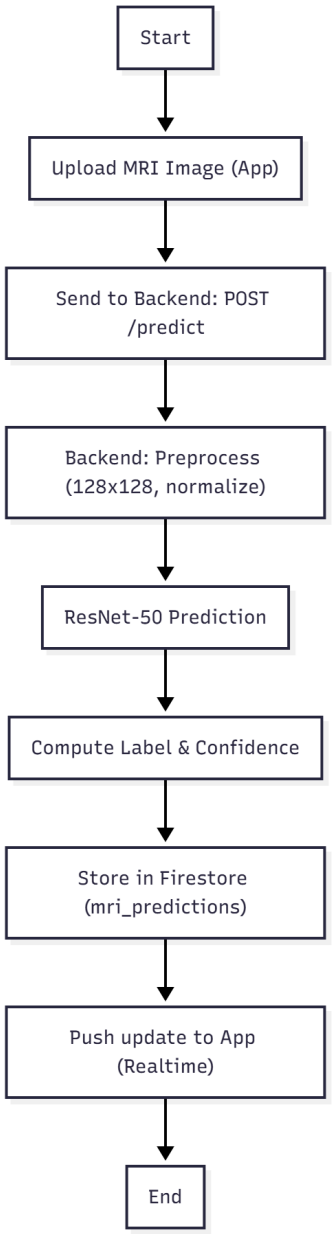

Figure 4.12: MRI Prediction Flowchart

2. Cognitive Game Score Computation:
   Figure 4.13 the Cognitive Game Score Computation flowchart.

Figure 4.13: Cognitive Game Score Flowchart

3. User Analytics Aggregation:
   Lastly, the User Analytics Aggregation process is represented in Figure 4.14.

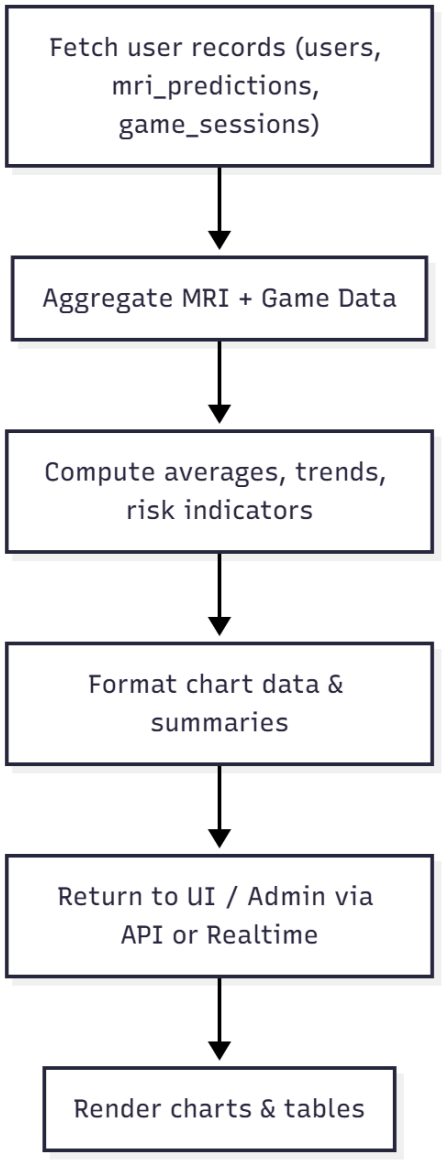

Figure 4.14: User Analytics Aggregation Flowchart

All of these low-level designs explain the internal behavior of the system, e.g. how the system generates predictions and displays analytics, as well as how each component effectively and predictably relates to another in the specified architecture.

4.6 AI Model Design

This section outlines dataset description, preprocessing pipeline, model architecture, and training strategy of the Alzheimer MRI classification model incorporated in the MindCare

system.

4.6.1 Dataset Description

The publicly available Alzheimer MRI dataset used to train the model was obtained from Kaggle. The data include T1-weighted brain MRI images that are divided into four classes of clinical interest, namely Non Demented, Very-Mild Demented, Mild Demented, and Moderate Demented. The dataset contains about 3,200 Non-Demented, 2,240 Very-Mild-Demented, 896 Mild-Demented and 64 Moderate-Demented images [15].

Such distribution implies the inherent class imbalance regularly present in medical data, where more advanced cases of the disease are less represented. To maintain uniform evaluation, we split the dataset into three sets which are training, validation, and testing sets in an 80-10-10 % split. The detailed distribution charts along with representative samples of each class are provided in Appendix A.

4.6.2 Data Preprocessing

All MRI images are processed through a standardized preprocessing pipeline prior to training in order to have consistency and model optimization. The images are first resized to 128 x 128 pixels and then scaled to the 0-1 range. This normalization aids in stabilizing learning through keeping the value scales equal amongst samples.

Data augmentation is used to enhance generalization, including steps like random rotation, zoom and shear transformations of the training set. These processes subject the model to different spatial patterns and less overfitting. Because the dataset is imbalanced, class weights are calculated and used in training such that the minority classes, especially the Moderate-Demented category, have sufficient representation in the loss function. Automated splitting utilities are eventually used to divide the dataset into organized Train/Validation/Test directories to make the dataset reproducible.

4.6.3 Model Architecture

A transfer learning method was implemented based on the ResNet50 architecture that has been pretrained on ImageNet because ResNet50 performs well in visual recognition tasks and also has residual connections which enable deep networks to have consistent training. The convolutional feature extraction layers of ResNet50 were only kept with a custom classification head added to the network to adapt it to four-class Alzheimer classification.

Figure 4.15 (created, as part of the development, with the plot_model utility) shows the entire model architecture, both ResNet50 backbone and custom classification head.

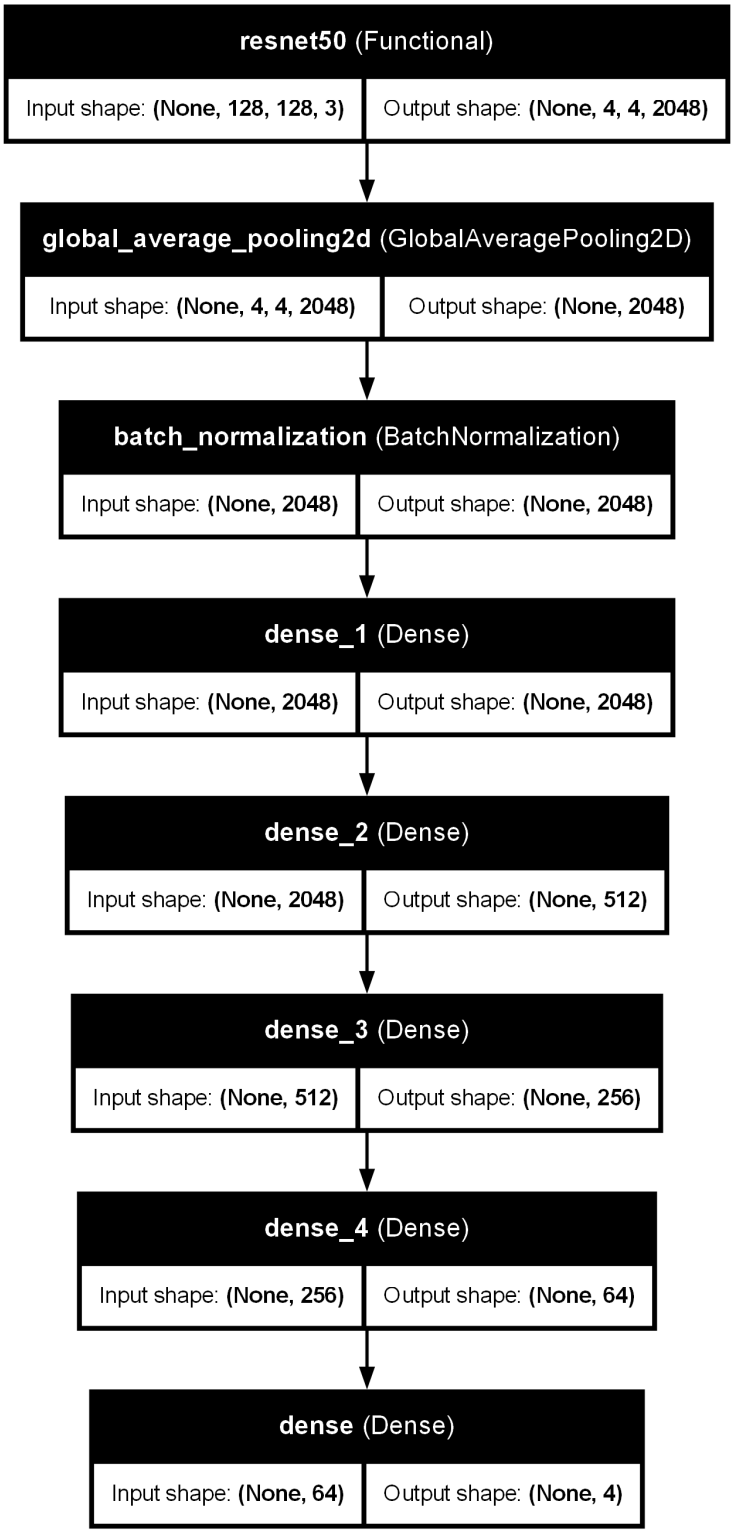

Figure 4.15: ResNet50-Based Alzheimer Classification Model

4.6.4 Training Strategy

The model is trained using Adam optimizer with a loss function of categorical cross-entropy. The batch size is 64 and several epochs are applied with EarlyStopping to avoid overfitting. The optimal model weights have been stored in a ModelCheckpoint callback to use during deployment.

Accuracy, precision, recall, and AUC on both training as well as validation set are used to monitor performance. The model shows consistent learning behavior during training. References on all metric curves and extended evaluation figures are provided in Appendix A.

4.7 Database Design

MindCare system has Firebase Firestore, a cloud-based no-SQL database where all user and system data is stored. Firestore provides real time synchronization, scalability, and secure authentication using Firebase Auth. Data is organized in collections and documents as opposed to the traditional tables to ensure clear separation of entities, minimal redundancy and rapid retrieval by users and administrators.

4.7.1 Entity Relationship (ER) Diagram

Though Firestore is not relational, a relational between collections is represented in terms of an ER model in Figure 4.16.

Main Entities:

1. users
2. mri_predictions
3. game_sessions
4. cognitive_assessments
5. user_cognitive_profiles

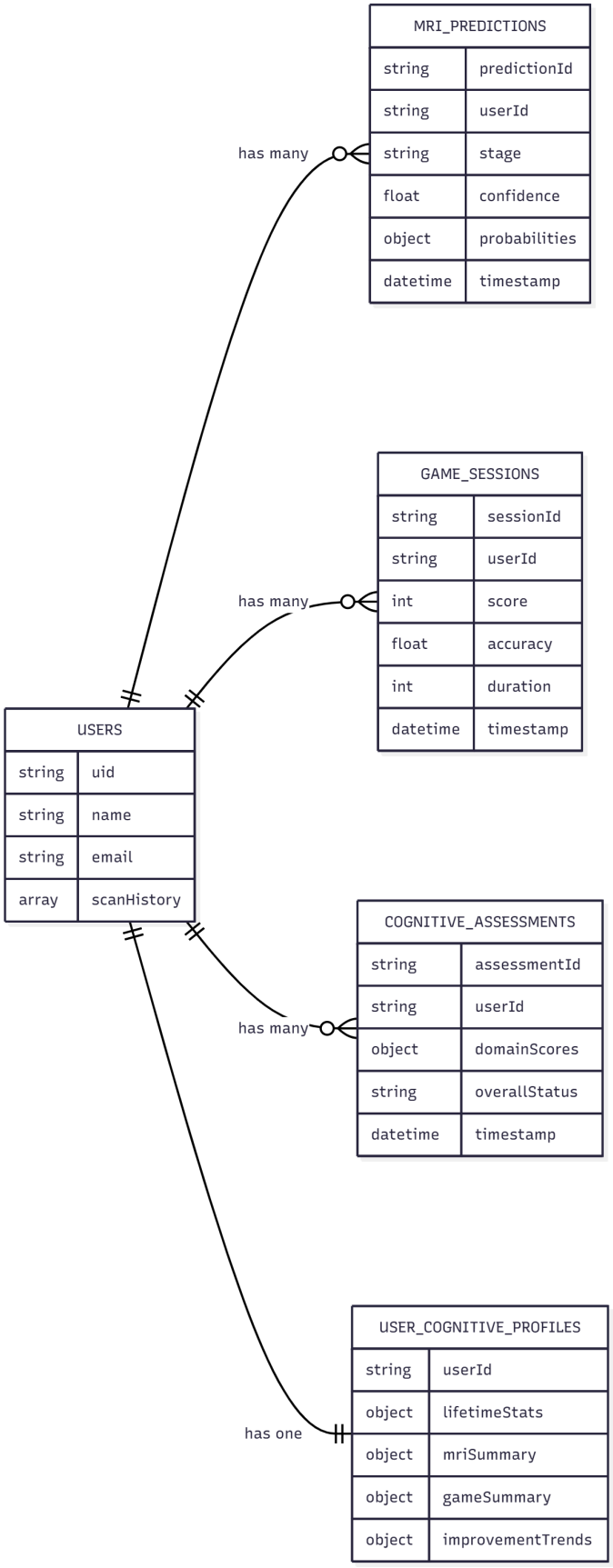

Figure 4.16: Entity Relationship Diagram (ERD)

4.7.2 Entity Descriptions

1. **users Collection**
   This collection stores scan summaries, registration and authentication. Whenever a new user is registered or first logs in, a new user document is created automatically. After each MRI prediction, scanHistory array is updated.

2. **mri_predictions Collection**
   This collection stores detailed MRI prediction outcomes. When a user uploads the MRI through the mobile application, a record is generated. The prediction is completed in the FastAPI back-end and the outcome is returned to the app where it is saved using the firebaseService.savePredictionResult() method.

3. **game_sessions Collection**
   This collection records all movements of cognitive game sessions with performance information like score, accuracy and time taken to complete the game.

4. **cognitive_assessments Collection**
   This collection makes cumulative cognitive reports at least 5 game sessions. It is created by the generateCognitiveAssessment () function once there is enough recent game play data.

5. **user_cognitive_profiles Collection**
   This collection retains a long-term cognitive profile of a user, which is a combination of MRI-based predictions and data on behavioral games to demonstrate overall cognitive well-being and improvement. This collection is updated each time a user has played a game or when a new cognitive assessment is created.

The logical relationships between these entities are shown in Figure 4.17.

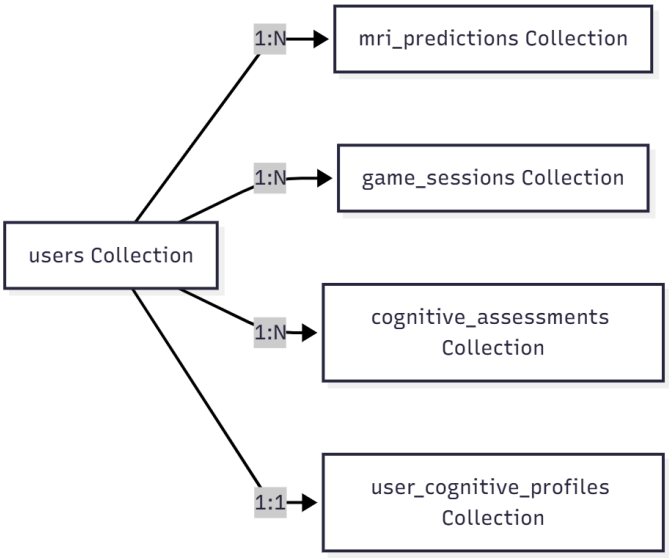

Figure 4.17: Logical Relationships

4.7.3 Data Flow with Firebase

Data is dynamically transferred between the frontend, backend and Firebase in real time. This data flow is shown in Figure 4.18.

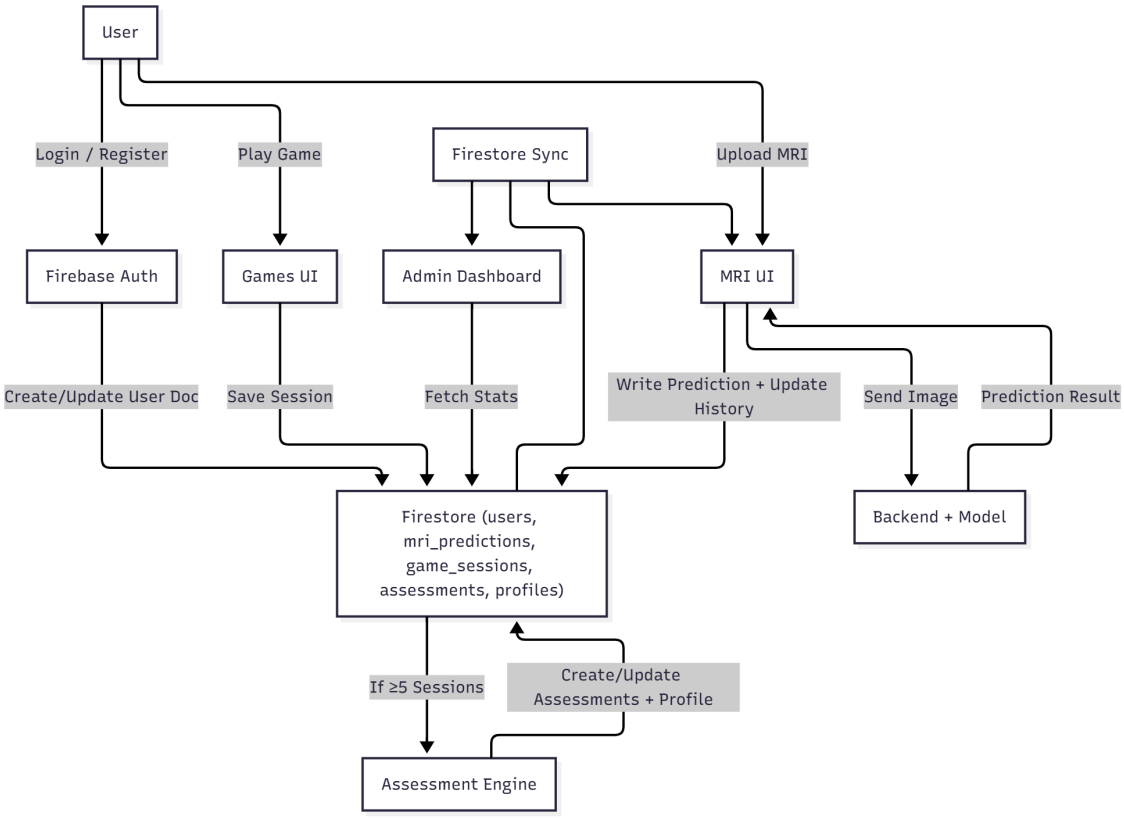

Figure 4.18: Firebase Data Flow

4.7.4 Data Validation and Security Rules

Firestore provides data integrity with the help of validation and Firebase Security Rules.

Validation within the Application Layer:

- Any user data is verified before it is written to Firestore.
- Firebase Authentication allows users to write/read only if they are authenticated.
- Admin privileges are checked in the app code, through hard-coded email checks.

Database-Level Rules and Access Control:

- The testing rules have been liberalized, whereas the production rules mandate role-based access.
- Secure communication is guaranteed by HTTPS.
- MRI images are not stored permanently, only prediction metadata is stored.

4.8 User Interface Design

The app has a mobile-first, user-focused design, which allows easy navigation and fast understanding. The color scheme has been applied with blue colors depicting trust, green color depicting positive actions and the neutral whites/grays to make the text readable. The major objectives of the UI design are

- **Clarity:** To display data including MRI outcomes and analytics in a readable and comprehensible format.
- **Accessibility:** To guarantee usability to all users including those with mild cognitive limitations.
- **Consistency:** To have a consistent design throughout all screens and modules.
- **Interactivity:** To deliver interactive experiences with cognitive games.
- **Responsiveness:** To provide a smooth navigation and flexibility on Android devices.

4.8.1 Navigation Design

The application navigation is logical for both user and administration module and is shown in Figure 4.19.

- **User Flow:** Splash – Onboarding – Login/Signup – User Dashboard – MRI Analysis – Cognitive Games (Memory Match, Reaction Timer, Spatial Navigation, Pattern Recognition) – Personalized Tips – Analytics.

- **Admin Flow:** Login – Admin Dashboard – User Management – MRI Analytics – Cognitive Analytics – System Reports

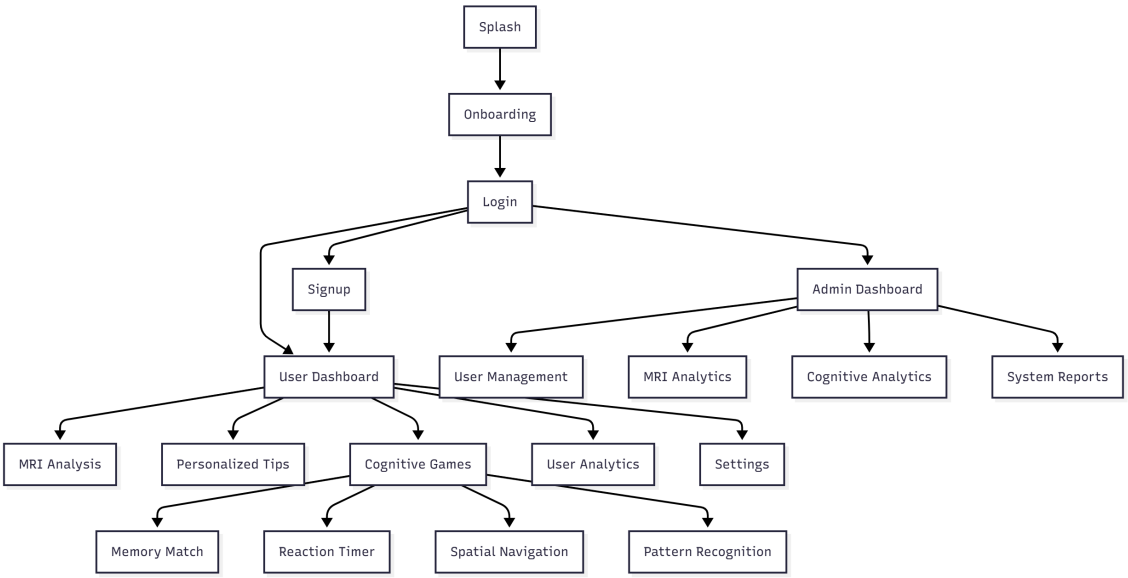

Figure 4.19: App Navigation Flow

4.8.2 Wireframes of Key Screens

The section provides the wireframes of the user interface and the administrator interface to demonstrate the appearance, navigation, and workflow of the MindCare application.

(A) User Module Screens

The user module wireframes are summarized in Table 4.2, detailing the purpose and functionality of each of the major screens in the mobile application.

Table 4.2: User Module Screens

Figure No.	Screen	Description
Figure 4.20	Splash Screen	This is the first screen which is displayed when the app is launched, showing the app logo and app title for about 3 seconds, and initializing Firebase services.
Figure 4.21	Onboarding Screen	This screen consists of a set of 3 short auto-scrolling slides that explain key app features and a "Let's get started" button, leading to login screen.
Figure 4.22	Login Screen	In this screen, current users can enter their email and password through Firebase Auth and gives new users a Forgot Password and Signup option.
Figure 4.23	Signup Screen	New users can create their accounts throuh this screen.
Figure 4.24	Forgot Password Screen	Users can reset their password using their registered email to which Firebase sends a reset-link.
Figure 4.25	User Dashboard	It is the primary screen and contains bottom navigation tabs of Home, MRI, Games, Tips, and Analytics. It shows welcome message, app features and quick access cards.
Figure 4.26	Settings Screen	This screen enables users to edit their profile, change password, and log out. Firebase is used to manage all changes.
Figure 4.27	MRI Analysis Screen	This screen allows users to upload MRI and display the predicted Alzheimer's stage and confidence level which are fetched from FastAPI backend in real-time.
Figure 4.28	Cognitive Games Screen	This screen displays total games played, saved games, 4 cognitive games with icons, progress indicators, and start buttons, and games' information.
Figure 4.29	Reaction Timer Game	The screen shows a reaction-based game in which users are asked to tap when the visual indicator changes to green from red, which will produce time-based performance scores.
Figure 4.30	Spatial Navigation Game	In this screen, a grid-based memory game is displayed in which individuals memorise the positions of stars in the grid and attempt to recreate them within time constraint.
Figure 4.31	Memory Match Game	In this screen, there is a traditional card-matching game in which users match pairs to test the short-term memory generating time and attempt-based score in performance.
Figure 4.32	Pattern Recognition Game	The screen displays MCQs of visual patterns which users are required to memorise and imitate as a measure of attention and recognition capacity.
Figure 4.33	Personalized Tips Screen	This screen displays personalized health tips based on recent MRI scan and game which is fetched from Firebase.
Figure 4.34	User Analytics Screen	This screen displays the charts and graphs of prediction history and game performance.

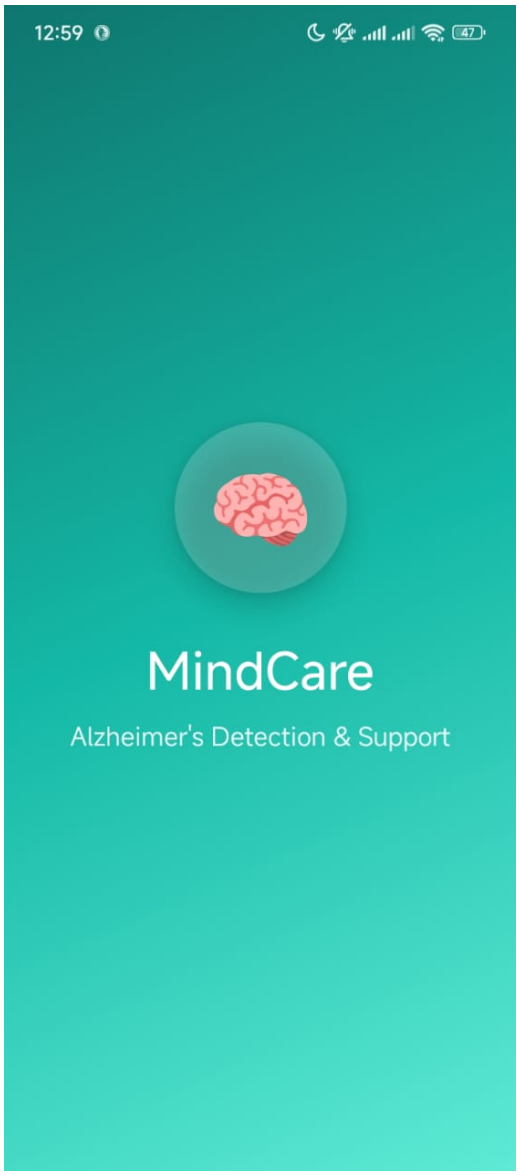

Figure 4.20: Splash Screen

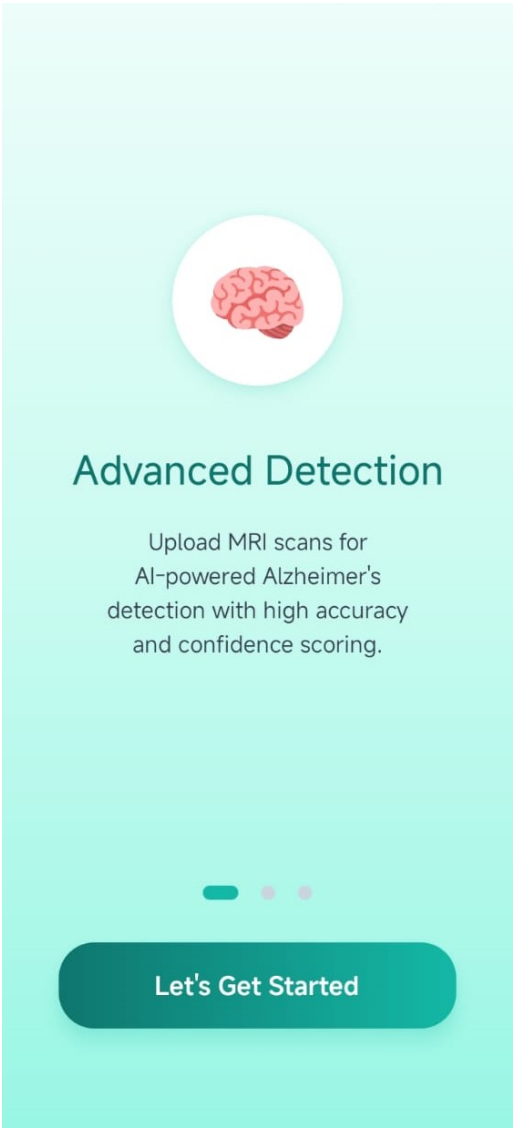

Figure 4.21: Onboarding Screen

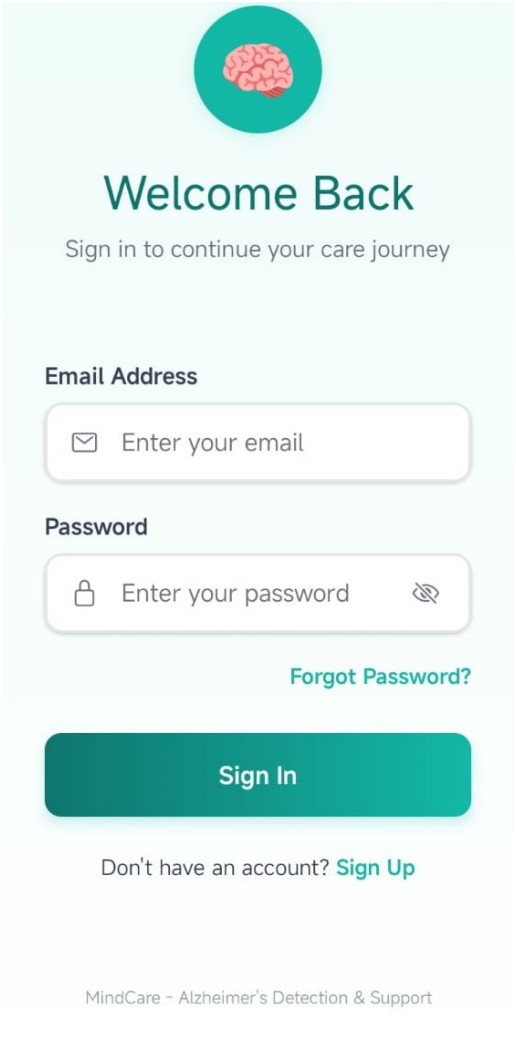

Figure 4.22: Login Screen

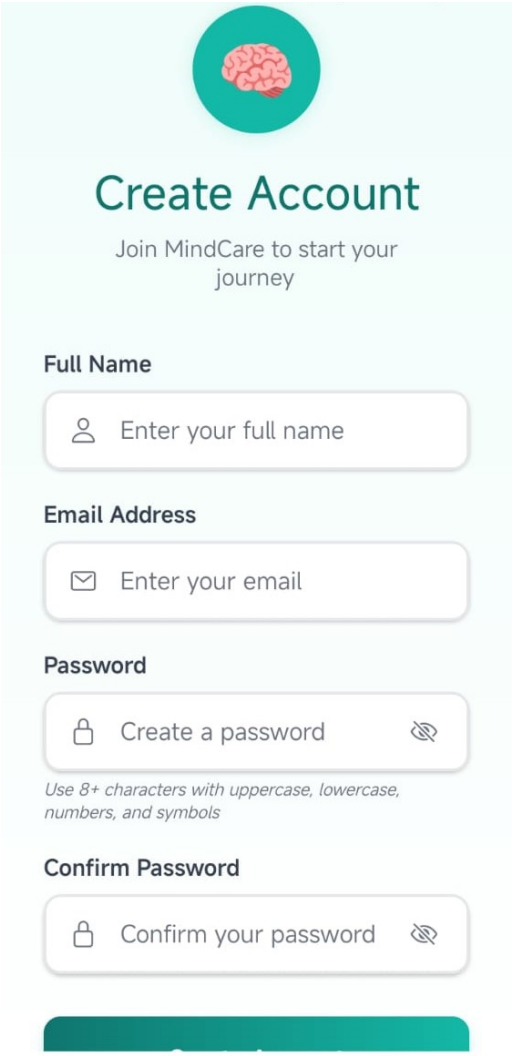

Figure 4.23: Signup Screen

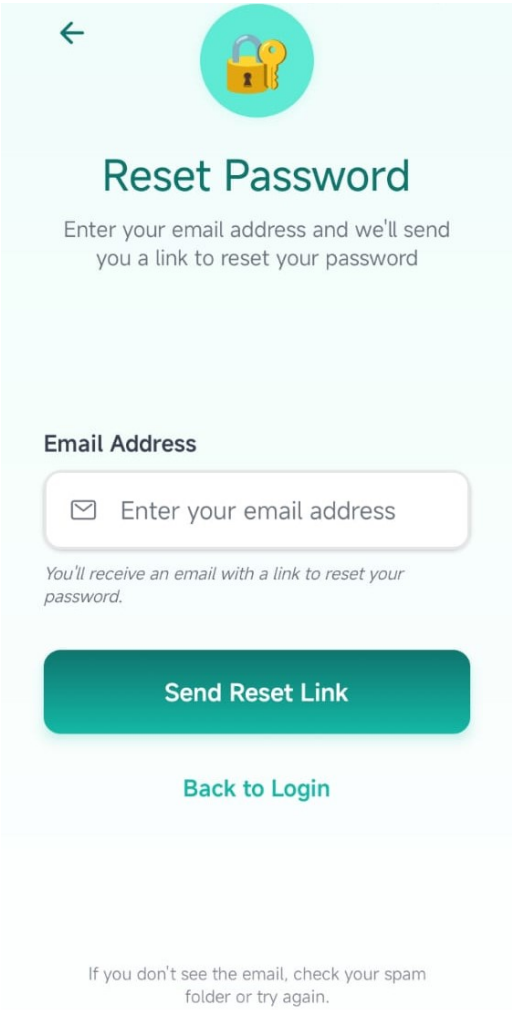

Figure 4.24: Forgot Password Screen

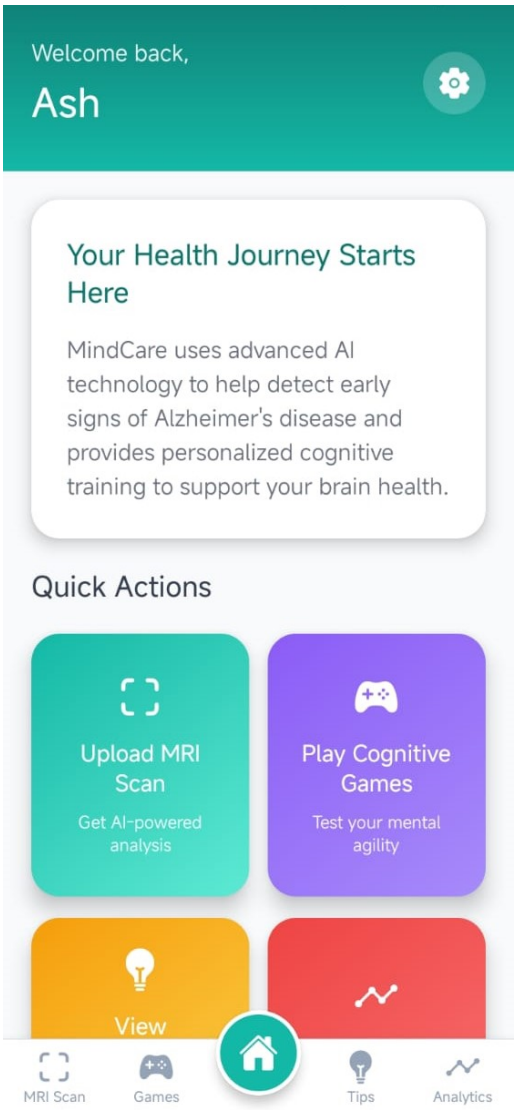

Figure 4.25: User Dashboard

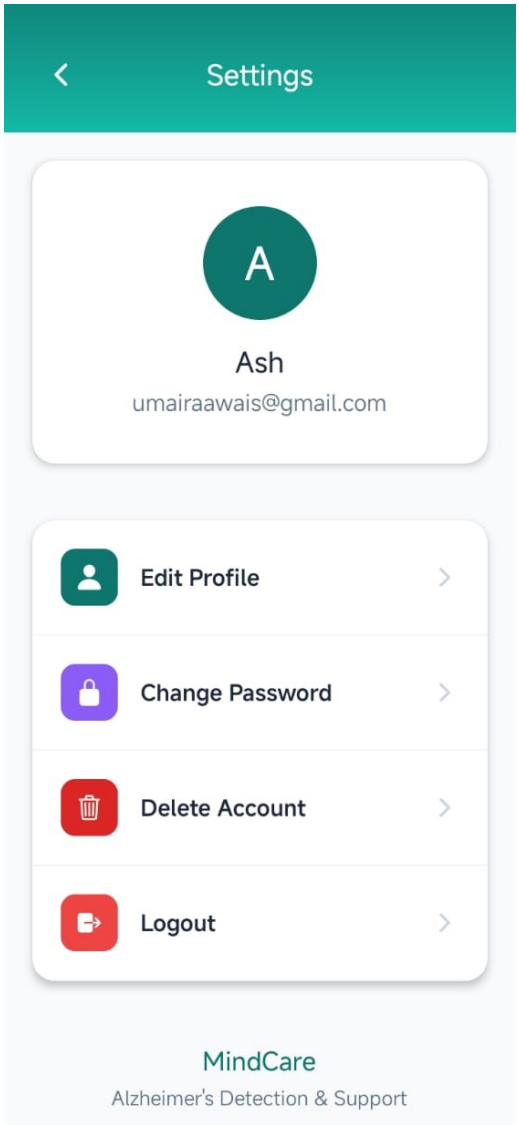

Figure 4.26: Settings Screen

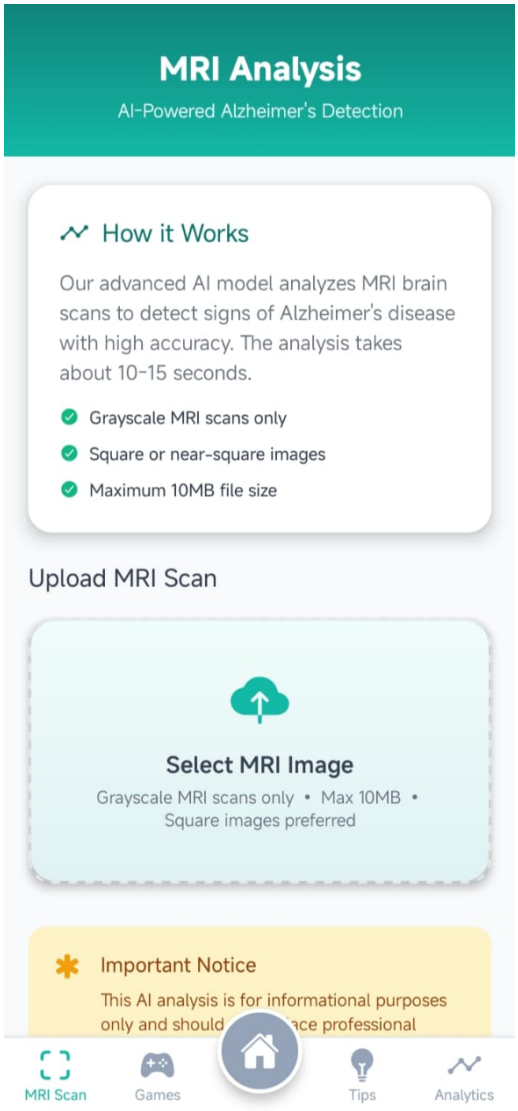

Figure 4.27: MRI Analysis Screen

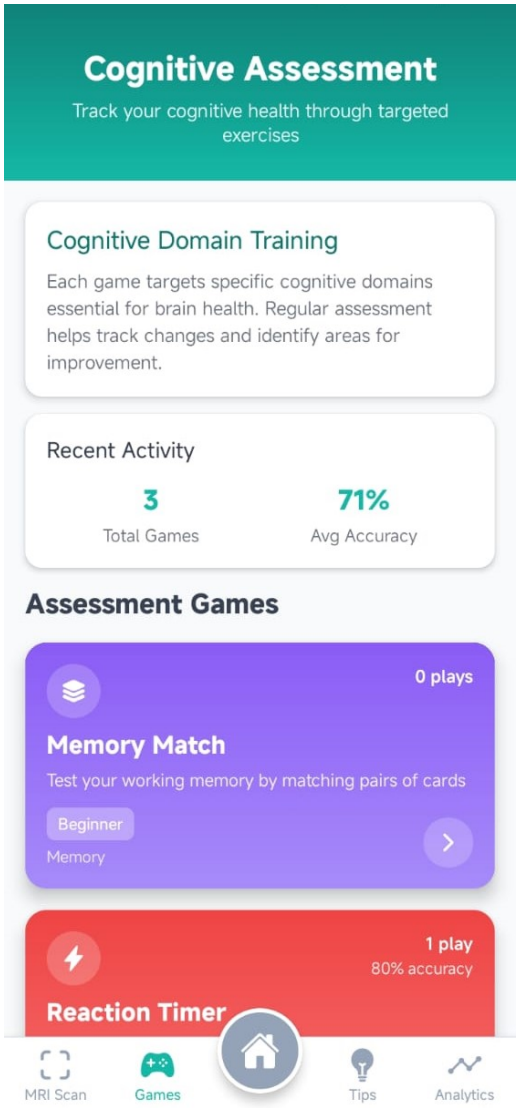

Figure 4.28: Cognitive Games Screen

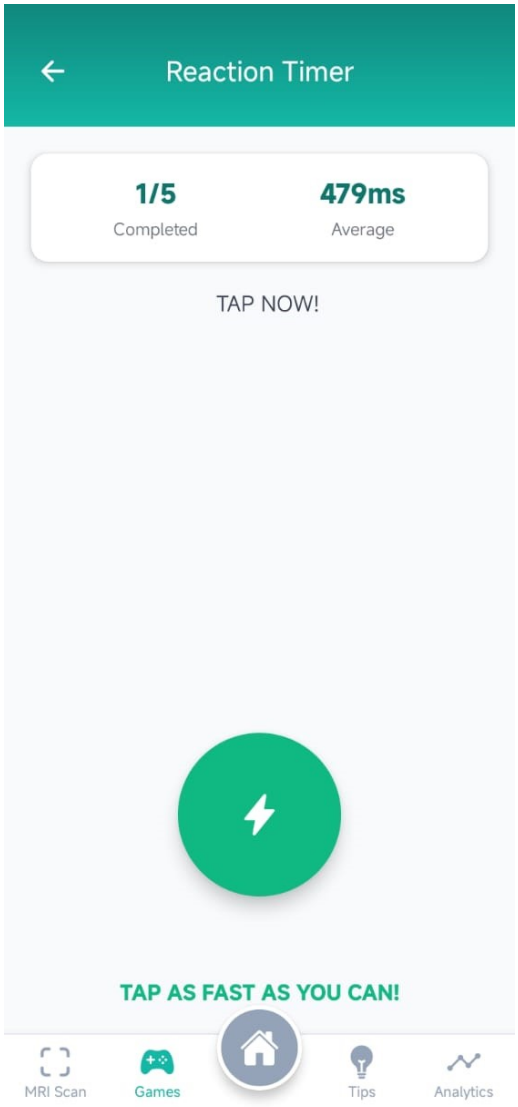

Figure 4.29: Reaction Timer Game

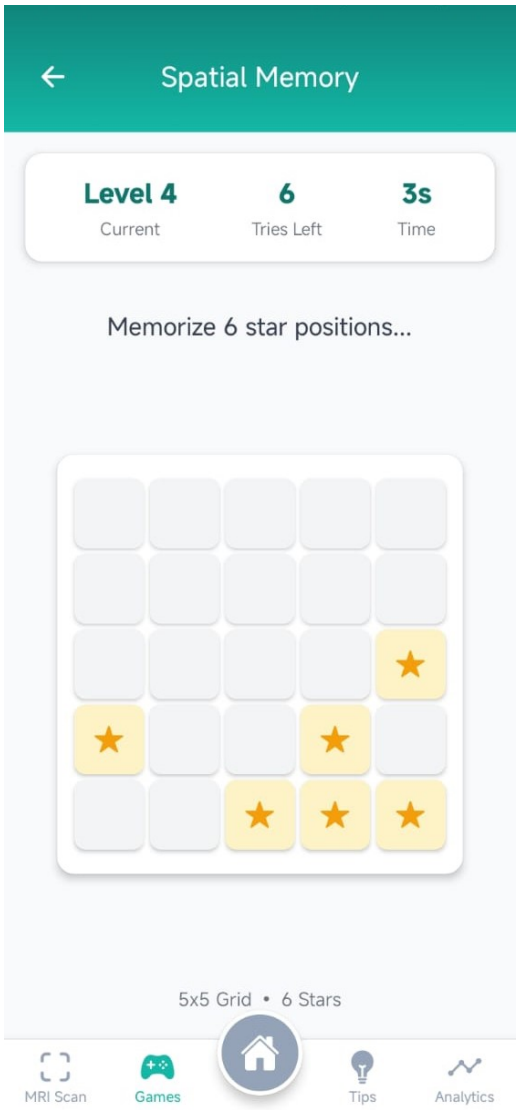

Figure 4.30: Spatial Navigation Game

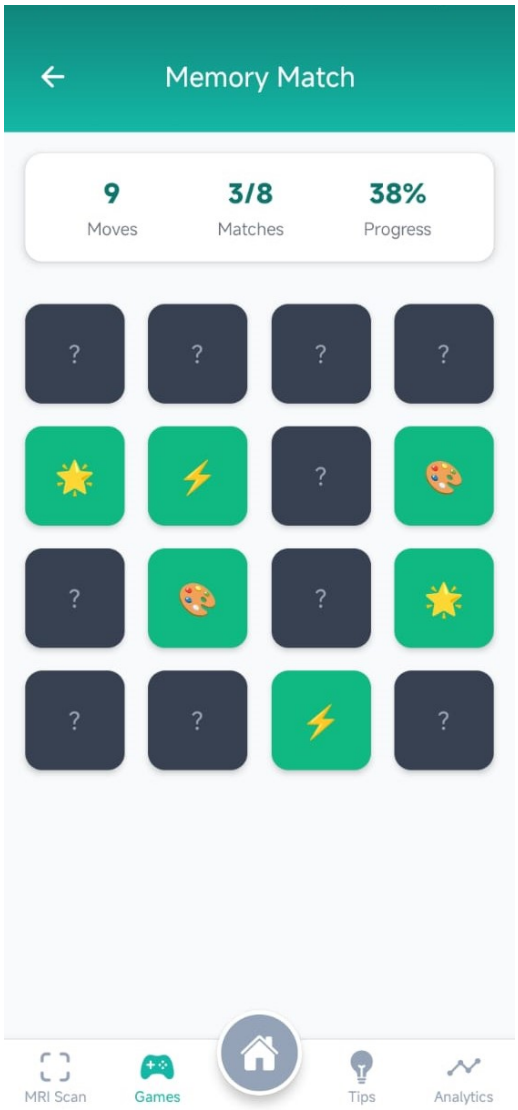

Figure 4.31: Memory Match Game

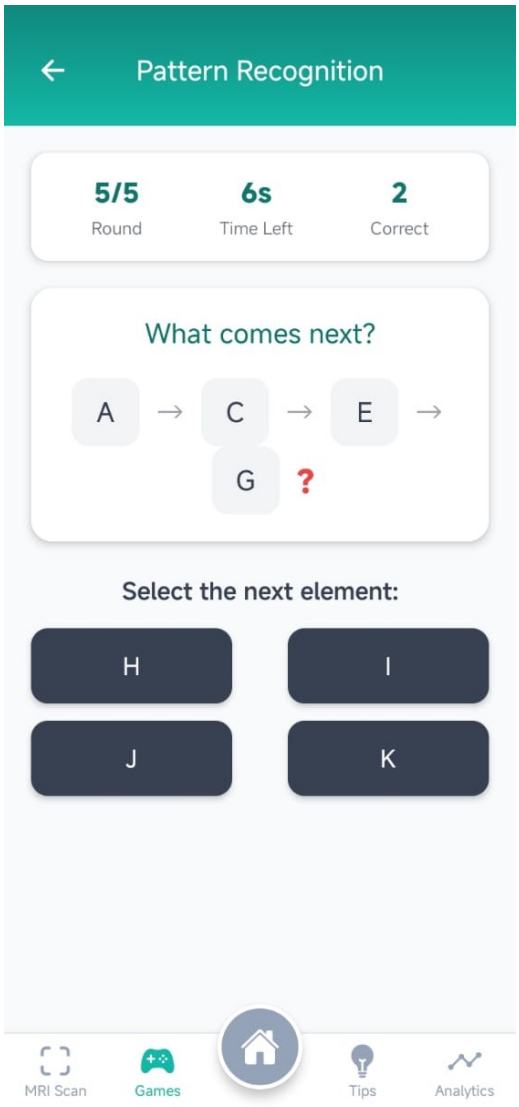

Figure 4.32: Pattern Recognition Game

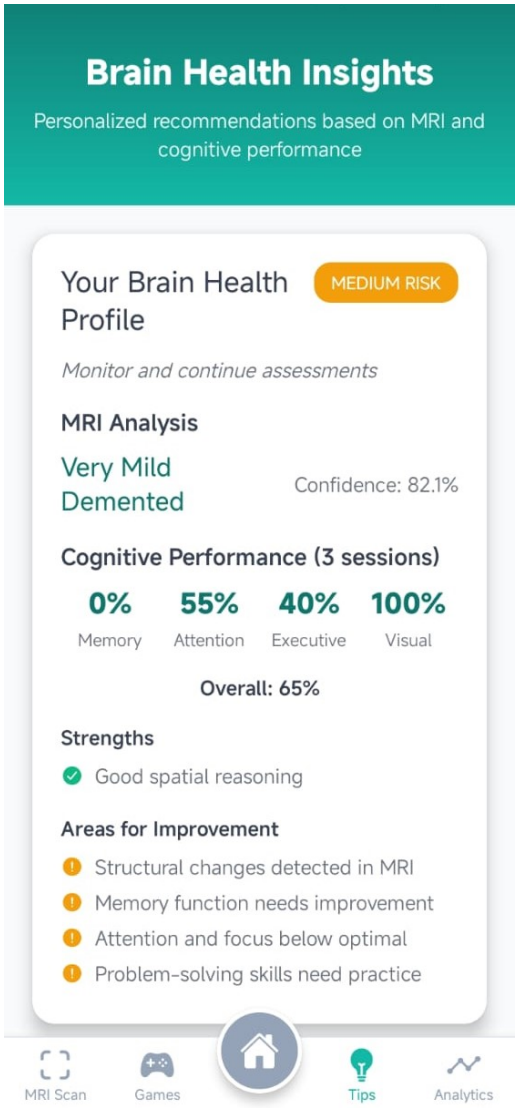

Figure 4.33: Personalized Tips Screen

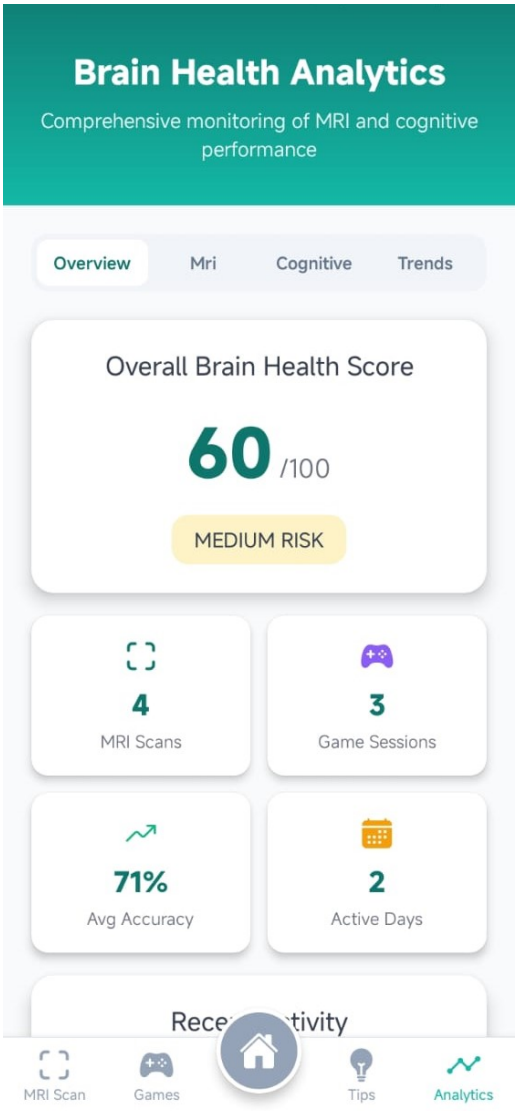

Figure 4.34: User Analytics Screen

(B) Admin Module Screens

Table 4.3 provides a summary of the wireframes of the admin module, including the screens where system monitoring, user management, and analytics are performed.

Table 4.3: Admin Module Screens

Figure No.	**Screen**	**Description**
Figure 4.35	Admin Dashboard Screen	This screen gives a summary of the system statistics such as total users, total predictions, and average confidence.
Figure 4.36	User Management Screen	This screen shows each user details (UID, name, email, scan and game data) and allows admin to delete user data.
Figure 4.37	MRI Analytics Screen	This screen displays combined MRI prediction data like stage distributions using bar and pie charts.
Figure 4.38	Cognitive Analytics Screen	This screen displays aggregate game scores, such as average accuracy, completion rates and the number of games played.
Figure 4.39	System Reports Screen	This screen generates summaries of system performance, predictions and cognitive activity.

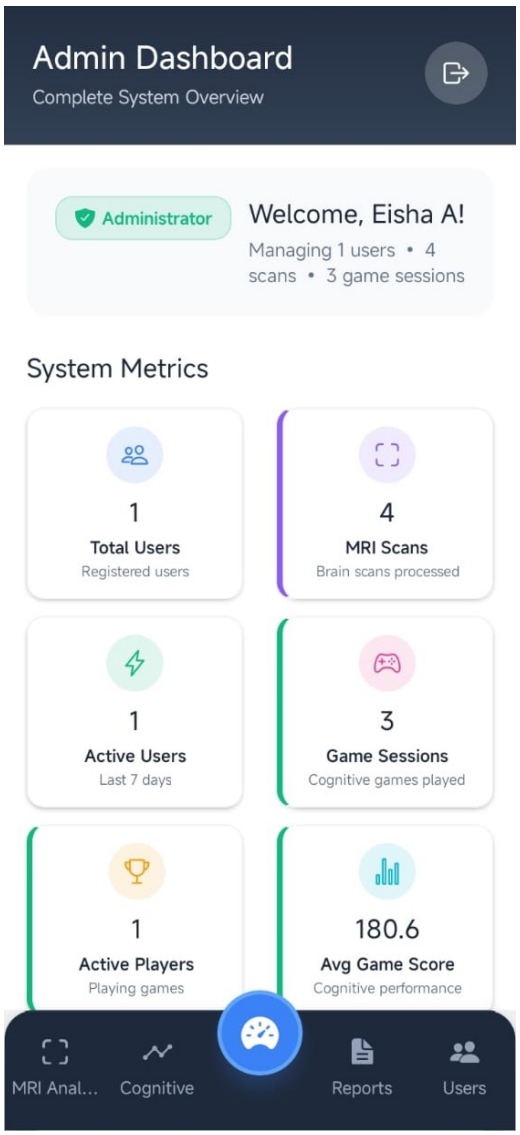

Figure 4.35: Admin Dashboard Screen

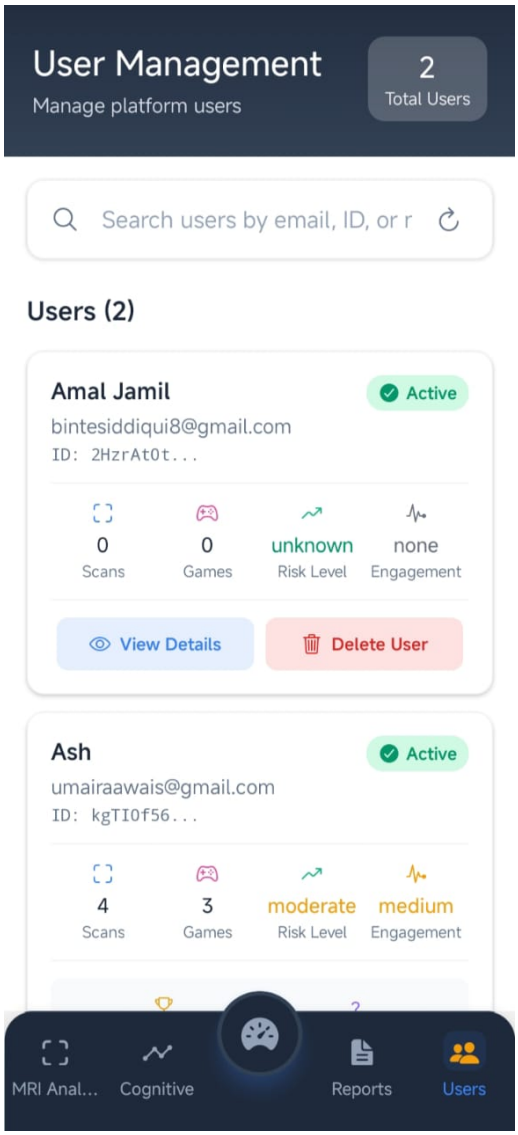

Figure 4.36: User Management Screen

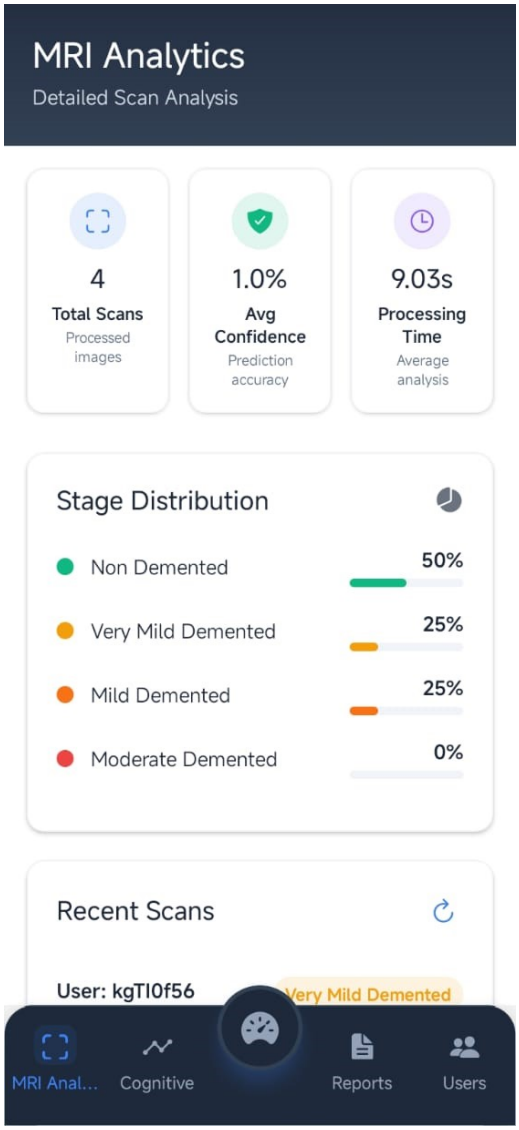

Figure 4.37: MRI Analytics Screen

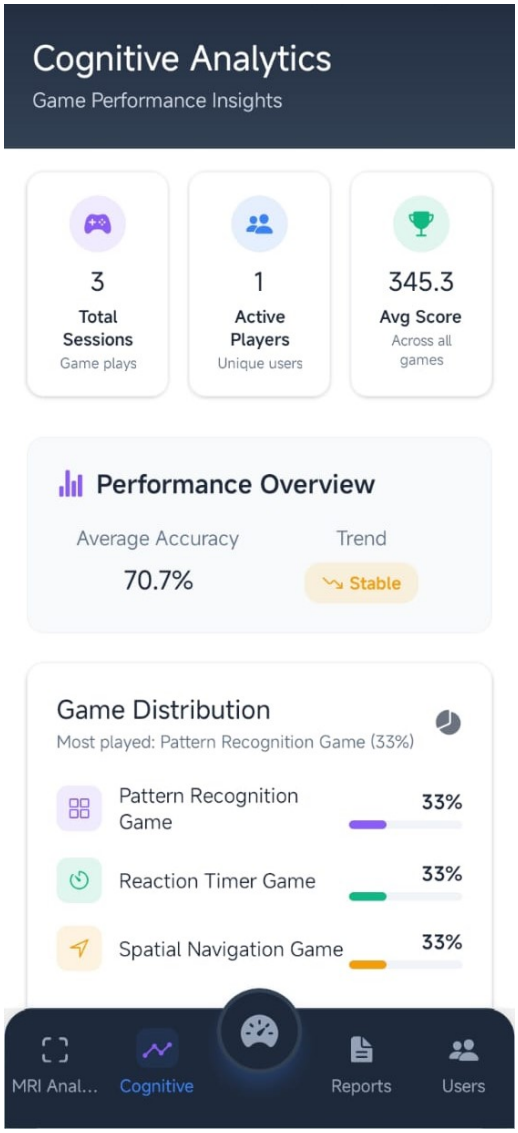

Figure 4.38: Cognitive Analytics Screen

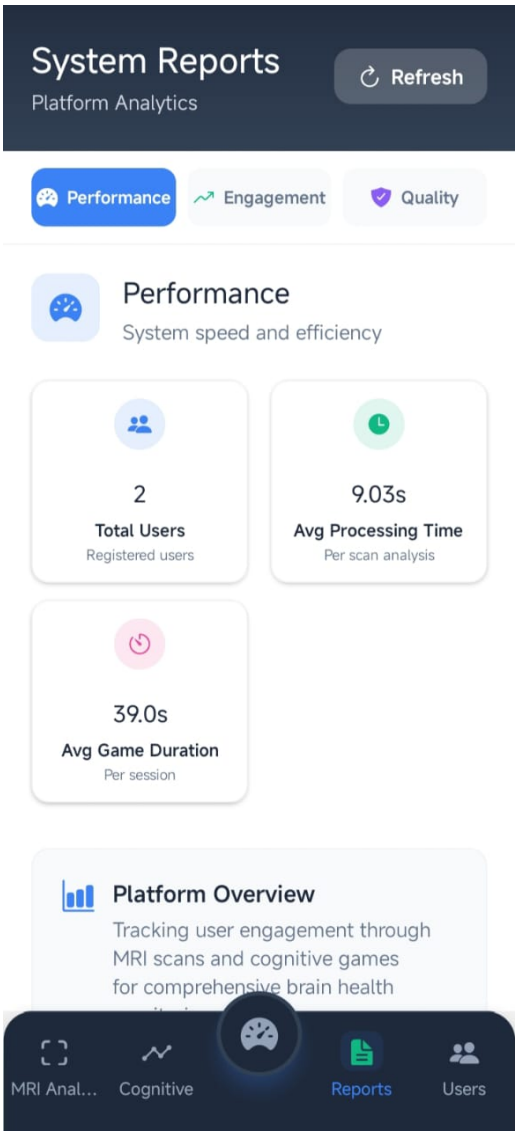

Figure 4.39: System Reports Screen

4.9 External Interfaces

4.9.1 FastAPI API Interface

FastAPI backend serves as an intermediary linking the mobile app and ResNet50 model. The MRI images pass through RESTful endpoints, where the preprocessing and inference operations are implemented and finally the predicted stage of Alzheimer's and the confidence score are sent back. Secure HTTPS is used in all communication.

4.9.2 Firebase Cloud Interface

Firebase offers authentication and data storage as well as real-time syncing. Firebase Authentication is used to log in and sign up into the app, whereas Firestore is used to store user profile, MRI results, cognitive scores, and analytics. Local processing of MRI images is done through the backend, and they are not stored in Firebase Storage.

4.9.3 TensorFlow/Keras Model Interface

The ResNet50 model (TensorFlow/Keras) model is built into the FastAPI backend. The backend uses the pre-trained weights and performs inference on every uploaded MRI and gives structured JSON responses.

4.9.4 Device-Level Interfaces

React Native libraries allow access to device hardware including storage and camera. Examples such as expo-image-picker and react-native-fs can be used to pick or capture MRI images, whereas device sensors and timers are used to score games and analytics.

4.9.5 Network and Security Interface

All the communication between the app, backend, and Firebase is encrypted with the help of HTTPS. Request authentication will be done by firebase tokens, and FastAPI CORS configurations will limit access to the API to authorised domains, thereby providing safe and reliable data flow.

4.10 Summary

This chapter summarizes the architecture and design of MindCare System. Firebase collections, high-level and low-level design structures were defined to demonstrate the way the user's MRI and cognitive data are arranged and secured. The overall UI design choices are consistent with the objectives defined earlier. The next chapter describes the system implementation based on the design discussed in this chapter.

Chapter 5

System Implementation

This chapter describes the process of the system development and includes the tools, technologies, and techniques to transform the design into a solution. The chapter also shows the way various components of the system were coordinated to produce the desired functionality.

5.1 System Architecture

The MindCare System has a local-cloud hybrid structure that integrates a locally-hosted FastAPI back-end, a TensorFlow/Keras ResNet50 MRI prediction model, and the cloud-hosted Firebase Firestore and Firebase Authentication. The frontend is developed using React Native that is connected to the backend and Firebase using secure HTTPS. This framework guarantees rapid inference, real-time coordination, as well as the modular separation of concerns.

5.1.1 Communication Between Components

The system elements interact via Firebase SDKs and REST APIs. Once an MRI scan has been uploaded, the React Native app transmits it to the FastAPI server for preprocessing and ResnNet50 model inference. The output of prediction is delivered in the form of JSON to the app and stored in Firestore. The real-time listeners automatically update UI whenever new data is availed thus keeping consistency between local and cloud data.

5.1.2 Application Access and Database Security

The system provides security on the following levels:

- **Authentication & Authorization:** Firebase Auth has access control; administrators are given higher levels of permissions.
- **HTTPS Transmission:** Every request is carried out in encrypted channels.
- **Firestore Rules:** Loose when developing, role-based access when in production.
- **Local Storage:** MRI images are not stored, just prediction results are stored.

5.1.3 Tools, Technologies, and Development Environment

The tools, frameworks, and technologies employed in the development and implementation of the MindCare system are listed in Table 5.1.

Table 5.1: Tools and Technologies Used

Category	**Tool / Framework**	**Purpose**
Languages	Python, TypeScript	Backend & Mobile development
Backend	FastAPI	REST API for prediction and data flow
AI Framework	TensorFlow/Keras	MRI classification
Mobile Framework	React Native	Cross-platform mobile app
Database	Firebase Firestore	Real-time storage
Authentication	Firebase Auth	Secure login & roles
IDE	VS Code	Development & debugging
Testing	Physical Android Device	Real-device validation

5.2 Implementation of Major Modules

This subsection indicates how the modules of the system interact with each other during the execution process, indicating how data will be passed between the mobile application, the backend services, the AI model, and Firebase.

5.2.1 Frontend (React Native)

The frontend incorporates screens of MRI analysis, games, analytics, settings, and administration. Firestore listeners and modular TypeScript components are used to make sure that it is updated in real-time. Testing was done on physical Android devices to test accuracy.

5.2.2 Backend (FastAPI Server)

MRI upload and prediction endpoints are offered by FastAPI. Conducts resizing, normalization, RGB conversion and inference. Produces a response in the form of a JSON, and logs predictions in Firestore.

5.2.3 AI Model (ResNet50 + TensorFlow/Keras)

The AI model does four-class classification of Alzheimer. Output contains the predicted class and index of class, probabilities and confidence. Model is easily upgradable.

5.2.4 Firebase Integration

Firebase is used to manage authentication, cloud storage and real time syncing. The collections consist of users, mri_predictions, game_sessions, cognitive_assessments, and user_cognitive_profiles Real-time listeners make sure that the analytics dashboards do not need to be manually refreshed.

5.2.5 Admin Dashboard

This is exclusively available to admins and it displays system-wide metrics: prediction distribution, game activity and user analytics. React-native-chart-kit is used in data visualization to provide clean and interactive charts.

5.2.6 Data Synchronization Flow

Figure 5.1 the end-to-end data processing and synchronization of the frontend, backend, AI model, and Firebase services.

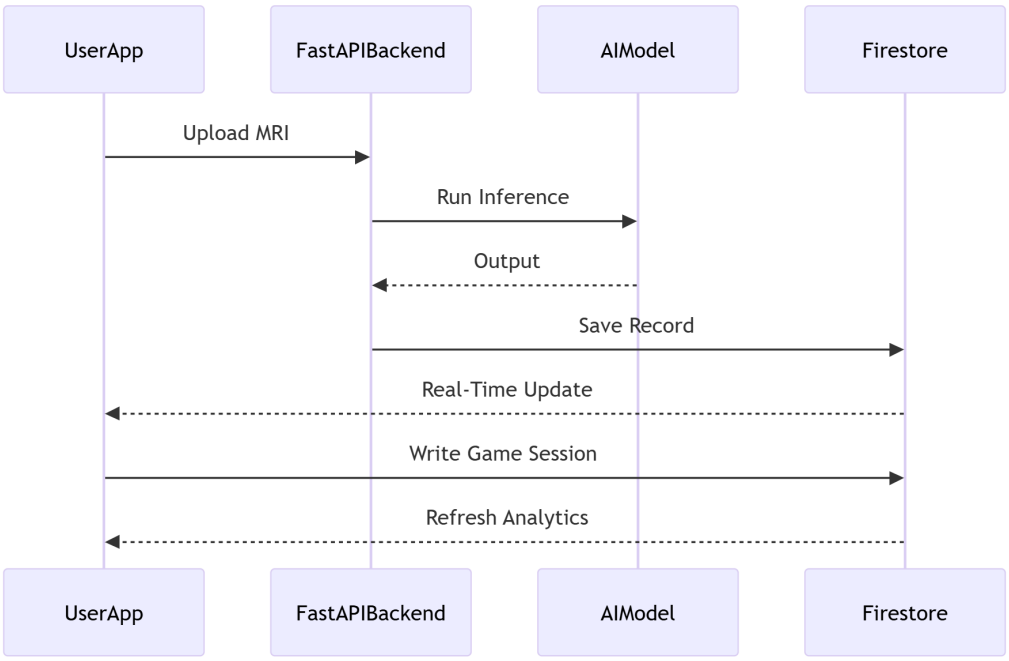

Figure 5.1: Data Processing & Synchronization Flow

5.3 Prepocessing Logic and Algorithms

This part outlines the pre-processing procedures and algorithms used on both MRI images and cognitive game data prior to prediction, analysis and storage.

5.3.1 MRI Preprocessing and Prediction

The uploaded MRI image is resized and normalized and expanded to the model input. The ResNet50 model estimates the probability of Alzheimer's stages but the stage with the highest confidence score is the output stage. The metadata (stage, confidence, timestamp) is the only data that is stored in Firestore.

5.3.2 Cognitive Game Logic

The cognitive games are executed locally on the device to assess spatial recognition, attention and memory. game_sessions store scores, accuracy and duration along with UID of the user. The design is modular in nature and does not require any changes on the backend to add or replace games.

5.3.3 Analytics Aggregation Logic

user_cognitive_profiles combines MRI and games. Statistics like average score on a game, number of sessions, confidence in the last prediction and newest stage of Alzheimer's are computed. Analytics is presented to the users and the admins through real-time updates and react-native-chart-kit charts.

5.4 Deployment and Configuration

The backend run via uvicorn and React Native app runs on a physical Android device through wireless ADB.

```
uvicorn app:app --host 0.0.0.0 --port 8000 --reload.
npx react-native start --host <local-IP>
npx react-native run-android.
```

A debug APK was developed using Gradle. Backend dependencies were configured through requirements.txt and mobile app dependencies were configured through npm. Google-services.json was used to load the firebase credentials. Successful prediction flow, game logging, real-time analytics and secure communication were established through end-to-end validation.

5.5 Summary

This chapter documents the deployment of MindCare System, the frontend, the backend, the AI model, the Firebase integration, processing logic, and the deployment process. The testing of the real devices proves that the system works with high security, efficiency and in real time. Its modular nature makes it easy to implement improvements in the future, i.e. new cognitive games or better AI models.

Chapter 6

System Testing and Evaluation

System testing shows that MindCare System is a complete integrated product where all modules are functioning as per the requirements and that the system as a whole is working reliably. It uses a black box, scenario-based methodology, where the entire workflow (user authentication, MRI prediction, cognitive gameplay, and analytics retrieval) is tested. The core goal of system testing is to verify that everything works correctly, detect usability and stability problems, and assess the extent to which the system can combine React Native, FastAPI and Firebase into a smooth user experience.

6.1 Testing Environment

Table 6.1 the hardware, software, and tools with which the system was tested in real and simulated conditions.

Table 6.1: Testing Environment

Category	**Details**
Hardware Setup	Windows 10 (Intel Core i5), 8 GB RAM, Android Emulator (Pixel 6 – API 30–33), Real Android Device (Android 12)
Software Setup	React Native (Expo), FastAPI (Python 3.10), Firebase Authentication & Firestore, VS Code, Postman, Firebase Console
Debugging Tools	React Native Chrome DevTools Debugger (opened through Metro using Press J / Open Debugger), Chrome/Expo Debugger
Test Data	Real MRI dataset (four Alzheimer's categories), cognitive game session data, multiple Firebase user accounts, known MRI images for verification

6.2 Functional Testing

In this section, the accuracy of every system operation is assessed by comparing the anticipated behavior to the actual operation within real usage parameters.

6.2.1 User Authentication Module

This module is tested to provide secure user access, acceptable credentials validation, and acceptable error handling.

TS-01: Successful Login with Valid Credentials

Table 6.2 the successful user login with valid credentials.

Table 6.2: TS-01: Successful Login with Valid Credentials

Test Scenario ID	TS-01	**Test Case ID**	TC-01
Test Case Description	To verify that a user can successfully log in by using valid credentials	**Test Priority**	High
Pre-Requisite	User account exists in Firebase Authentication.	**Post-Requisite**	User is redirected to dashboard screen.
Test Execution Steps:			

S.No	**Action**	**Inputs**	**Expected Output**	**Actual Output**	**Test Environment**	**Result**	**Test Comments**
1.	Open login screen	—	Login page loads successfully	As expected	Android, Firebase Auth	Pass	UI loads correctly
2.	Enter valid credentials	Email, Password	Email shown in field but password is hidden.	As expected	Android	Pass	—
3.	Tap ‘login’ button	—	Correct credentials accepted, redirected to dashboard	As expected	Android	Pass	User gets authenticated
Overall Test Result						**Pass**	

TS-02: Login Attempt with Invalid Credentials

Table 6.3 proves system behavior in the case of invalid login credentials.

Table 6.3: TS-02: Login Attempt with Invalid Credentials

Test Scenario ID	TS-02	**Test Case ID**	TC-02
Test Case Description	To verify that the system doesn't allow login attempts using invalid credentials	**Test Priority**	High
Pre-Requisite	Wrong credentials	**Post-Requisite**	Error is displayed and access is not granted.
Test Execution Steps:			

S.No	**Action**	**Inputs**	**Expected Output**	**Actual Output**	**Test Environment**	**Result**	**Test Comments**
1.	Open login screen	—	Login page loads successfully	As expected	Android, Firebase Auth	Pass	UI loads correctly
2.	Enter invalid credentials	Invalid email or invalid password	Email shown in field but password is hidden.	As expected	Android	Pass	—
3.	Tap 'login' button	—	"Invalid Credentials" message shown and fields are cleared	As expected	Android	Pass	Proper error handled
Overall Test Result						**Pass**	

TS-03: Password Reset Functionality

Table 6.4 shows that the password reset mechanism was working correctly.

Table 6.4: TS-03: Password Reset Functionality

Test Scenario ID	TS-03	**Test Case ID**	TC-03
Test Case Description	To verify that the password reset functionality works as intended.	**Test Priority**	Medium
Pre-Requisite	User account exists in Firebase Authentication.	**Post-Requisite**	Reset link is sent via email.
Test Execution Steps:			

S.No	**Action**	**Inputs**	**Expected Output**	**Actual Output**	**Test Environment**	**Result**	**Test Comments**
1.	Open login screen	—	Login page loads successfully	As expected	Android	Pass	UI loads correctly
2.	Tap 'Forgot Password' button	—	Redirected to password reset screen	As expected	Android, Firebase Auth	Pass	—
3.	Enter registered email	Valid email	A reset link is sent to the email	As expected	Android, Firebase Auth	Pass	New password set
Overall Test Result						**Pass**	

6.2.2 MRI Prediction Module

This module is challenged to confirm MRI upload processing, accuracy of predictions, error detection, and storage of results in a secure manner.

TS-04: Valid MRI Upload and Prediction Processing

Successful MRI upload and prediction processing is validated in Table 6.5.

Table 6.5: TS-04: Valid MRI Upload and Prediction Processing

Test Scenario ID	TS-04	**Test Case ID**	TC-04
Test Case Description	To verify and validate the process of uploading MRI image and prediction	**Test Priority**	High
Pre-Requisite	Backend API is running, valid `.jpg` MRI image	**Post-Requisite**	Prediction result is stored in Firestore in JSON format.

Test Execution Steps:

S.No	**Action**	**Inputs**	**Expected Output**	**Actual Output**	**Test Environment**	**Result**	**Test Comments**
1.	Open MRI Prediction Screen	—	UI loads successfully	As expected	Android, FastAPI	Pass	—
2.	Upload valid MRI image	Image file	AI prediction is displayed and stored in Firestore	As expected	Android, Firebase firestore	Pass	Verified Firestore save
Overall Test Result						**Pass**	

TS-05: System Handling of Invalid MRI Image Input

Table 6.6 measures how the system copes with the invalid MRI image inputs.

Table 6.6: TS-05: System Handling of Invalid MRI Image Input

Test Scenario ID	TS-05	**Test Case ID**	TC-05
Test Case Description	To verify the handling of invalid MRI image	**Test Priority**	High
Pre-Requisite	Invalid image selected	**Post-Requisite**	Error message is shown and no prediction is made
Test Execution Steps:			

S.No	**Action**	**Inputs**	**Expected Output**	**Actual Output**	**Test Environment**	**Result**	**Test Comments**
1.	Open MRI Prediction Screen	—	UI loads successfully	As expected	Android, FastAPI	Pass	—
2.	Upload invalid MRI image	Invalid image file	“Invalid File” alert	As expected	Android, FastAPI	Pass	Error detected correctly
Overall Test Result						**Pass**	

TS-06: Firestore Storage After Successful Prediction

Table 6.7 confirms that the results of prediction are properly stored in Firestore.

Table 6.7: TS-06: Firestore Storage After Successful Prediction

Test Scenario ID	TS-06	**Test Case ID**	TC-06
Test Case Description	To verify that after prediction, results are stored in Firebase Firestore collection	**Test Priority**	High
Pre-Requisite	Successful prediction done	**Post-Requisite**	Result saved in `mri_predictions` collection
Test Execution Steps:			

S.No	**Action**	**Inputs**	**Expected Output**	**Actual Output**	**Test Environment**	**Result**	**Test Comments**
1.	Open MRI Prediction Screen	—	UI loads successfully	As expected	Android, FastAPI	Pass	—
2.	Perform valid prediction	Image file	A new entry appears in collection	As expected	Android, Firebase Firestore	Pass	Record saved and verified
Overall Test Result						**Pass**	

TS-07: Verification of Correct Prediction Output

Table 6.8 ascertains that the AI model makes correct prediction of known MRI inputs.

Table 6.8: TS-07: Verification of Correct Prediction Output

Test Scenario ID	TS-07	**Test Case ID**	TC-07
Test Case Description	To verify whether the prediction made is correct or not	**Test Priority**	High
Pre-Requisite	Output is known for selected MRI image	**Post-Requisite**	Label matches expected output
Test Execution Steps:			

S.No	**Action**	**Inputs**	**Expected Output**	**Actual Output**	**Test Environment**	**Result**	**Test Comments**
1.	Open MRI Prediction Screen	—	UI loads successfully	As expected	Android, FastAPI	Pass	—
2.	Upload known MRI image	Mild Demented MRI image file	Output "Mild Demented" with confidence score	As expected	Android, FastAPI	Pass	AI model is accurate
Overall Test Result						**Pass**	

6.2.3 Cognitive Games Module

This module is tested to check the game logic and behavior of user interaction along with persistence of session data.

TS-08: Reaction Timer Game Functionality

The functionality of the reaction timer cognitive game is proven in Table 6.9.

Table 6.9: TS-08: Reaction Timer Game Functionality

Test Scenario ID	TS-08	**Test Case ID**	TC-08
Test Case Description	To verify the functionality of reaction timer game	**Test Priority**	Medium
Pre-Requisite	User is logged in	**Post-Requisite**	Score is saved in Firestore collection
Test Execution Steps:			

S.No	**Action**	**Inputs**	**Expected Output**	**Actual Output**	**Test Environment**	**Result**	**Test Comments**
1.	Open Reaction Timer game from Cognitive Games screen	—	Menu loads successfully	As expected	Android	Pass	—
2.	Start game	Tap Start button	Timer runs, score recorded	As expected	Android, Firebase	Pass	Game logic is correct
Overall Test Result						**Pass**	

TS-09: Memory Match Game Functionality

Table 6.10 confirms proper implementation of memory match game.

Table 6.10: TS-09: Memory Match Game Functionality

Test Scenario ID	TS-09	**Test Case ID**	TC-09
Test Case Description	To verify the functionality of memory match game	**Test Priority**	Medium
Pre-Requisite	User is logged in	**Post-Requisite**	Score is saved in Firestore collection
Test Execution Steps:			

S.No	**Action**	**Inputs**	**Expected Output**	**Actual Output**	**Test Environment**	**Result**	**Test Comments**
1.	Open Memory Match game from Cognitive Games screen	—	Menu loads successfully	As expected	Android	Pass	—
2.	Start game	Click cards	Score and accuracy is calculated	As expected	Android, Firebase	Pass	Game logic is correct
Overall Test Result						**Pass**	

TS-10: Pattern Recognition Game Functionality

Table 6.11 confirms the pattern recognition game logic and scoring.

Table 6.11: TS-10: Pattern Recognition Game Functionality

Test Scenario ID	TS-10	**Test Case ID**	TC-10
Test Case Description	To verify the functionality of pattern recognition game	**Test Priority**	Medium
Pre-Requisite	User is logged in	**Post-Requisite**	Score is saved in Firestore collection
Test Execution Steps:			

S.No	**Action**	**Inputs**	**Expected Output**	**Actual Output**	**Test Environment**	**Result**	**Test Comments**
1.	Open Pattern Recognition game from Cognitive Games screen	—	Menu loads successfully	As expected	Android	Pass	—
2.	Start game	Choose options	Score is calculated	As expected	Android, Firebase	Pass	Game logic is correct
Overall Test Result						**Pass**	

TS-11: Spatial Navigation Game Functionality

Table 6.12 checks proper gameplay behavior of the spatial navigation game.

Table 6.12: TS-11: Spatial Navigation Game Functionality

Test Scenario ID	TS-11	**Test Case ID**	TC-11
Test Case Description	To verify the functionality of spatial navigation game	**Test Priority**	Medium
Pre-Requisite	User is logged in	**Post-Requisite**	Score is saved in Firestore collection
Test Execution Steps:			

S.No	**Action**	**Inputs**	**Expected Output**	**Actual Output**	**Test Environment**	**Result**	**Test Comments**
1.	Open Spatial Navigation game from Cognitive Games screen	—	Menu loads successfully	As expected	Android	Pass	—
2.	Start game	Choose cards	Score is calculated after each level	As expected	Android, Firebase	Pass	Game logic is correct
Overall Test Result						**Pass**	

TS-12: Game Session Storage in Firestore

Table 6.13 validates the fact that the completed game sessions are recorded and displayed in the UI.

Table 6.13: TS-12: Game Session Storage in Firestore

Test Scenario ID	TS-12	**Test Case ID**	TC-12
Test Case Description	To verify the game session storage and UI updates.	**Test Priority**	High
Pre-Requisite	Game played	**Post-Requisite**	Session scores stored in Firestore and visible in Cognitive Games screen

Test Execution Steps:

S.No	**Action**	**Inputs**	**Expected Output**	**Actual Output**	**Test Environment**	**Result**	**Test Comments**
1.	Finish any game	—	Game session visible in Firestore and in Cognitive Games screen	As expected	Android, Firestore	Pass	—
Overall Test Result						**Pass**	

TS-13: UI Behaviour During Gameplay (Pause/Resume/Exit)

Table 6.14 measures UI behavior when pausing, resuming and leaving the game.

Table 6.14: TS-13: UI Behaviour During Gameplay (Pause/Resume/Exit)

Test Scenario ID	TS-13	**Test Case ID**	TC-13
Test Case Description	To verify the UI behavior during gameplay like pause, resume, exit.	**Test Priority**	High
Pre-Requisite	Game being played	**Post-Requisite**	Session saved and visible in Cognitive Games screen
Test Execution Steps:			

S.No	**Action**	**Inputs**	**Expected Output**	**Actual Output**	**Test Environment**	**Result**	**Test Comments**
1.	Click on back navigation button to pause, resume or exit	Click on back navigation button	Session saved and visible in Cognitive Games screen, can be resumed.	As expected	Android	Pass	Scores are not saved in FIrestore unless gameplay completion
Overall Test Result						**Pass**	

6.2.4 Analytics Module

This module is tested to ensure proper aggregation and real time visualization of user and system analytics.

TS-14: User Analytics Updates After Predictions and Games

Table 6.15 confirms the updates of real time user analytics after games and predictions.

Table 6.15: TS-14: User Analytics Updates After Predictions and Games

Test Scenario ID	TS-14	**Test Case ID**	TC-14
Test Case Description	To verify the user analytics updates after predictions, games etc.	**Test Priority**	High
Pre-Requisite	Completion of prediction/game	**Post-Requisite**	Stats updated in dashboard
Test Execution Steps:			

S.No	**Action**	**Inputs**	**Expected Output**	**Actual Output**	**Test Environment**	**Result**	**Test Comments**
1.	Open User Analytics screen	—	Updated info, graphs visible	As expected	Android, Firestore	Pass	—
Overall Test Result						**Pass**	

TS-15: Admin Analytics and Dashboard Metrics Retrieval

Valid retrieval and visualization of the analytics at the admin level is confirmed in Table 6.16.

Table 6.16: TS-15: Admin Analytics and Dashboard Metrics Retrieval

Test Scenario ID	TS-15	**Test Case ID**	TC-15
Test Case Description	To verify the analytics updates on admin side.	**Test Priority**	High
Pre-Requisite	Data in Firestore	**Post-Requisite**	Metrics fetched and showed
Test Execution Steps:			

S.No	**Action**	**Inputs**	**Expected Output**	**Actual Output**	**Test Environment**	**Result**	**Test Comments**
1.	Open Admin screens i.e., MRI Analytics, Cognitive Analytics, User Management, System Analytics screens.	—	Shows totals, average accuracy, average scores, active users, scans etc.	As expected	Android, Firebase	Pass	Verified in realtime
Overall Test Result						**Pass**	

The results of the functional testing indicate that all the core modules of the system perform as intended when subjected to expected use conditions. The accuracy, reliability, and data consistency of authentication, MRI prediction, cognitive games, and analytics features were verified. The successful completion of all test cases can be used to testify that the system is stable, responsive, and can be deployed in the real world.

6.3 Non-Functional Testing

The non-functional testing was performed to test the usability, functionality, security, compatibility, and the general stability of the system. The objective of the non-functional testing is to ensure that the application not only works properly, but also provides a comfortable, secure and consistent usage experience on a variety of devices and under varying conditions of use.

6.3.1 Usability Testing

Usability testing ensures that a new user can use the system easily with no difficulty. There was no guidance given to test users on how to use the MRI screen, gameplay screens, and analytics panels. The general workflow was user-friendly, the buttons were properly marked, and the feedback was responsive. The level of usability was significant since users could accomplish prediction and gameplay tasks without any confusion.

6.3.2 GUI Testing

GUI testing was done on layout responsiveness, scaling, and pictorial consistency between emulators and real Android devices. Everything was rendered correctly, there were no overlapping elements or clipped parts. Font, spacing, and contrast color were consistent across devices, and screen to screen transitions worked well.

6.3.3 Performance Testing

Performance testing was done by monitoring API response times, loading behaviour and gameplay fluidity. FastAPI prediction endpoint was able to return results in less than one second, Firestore read/write operations took a matter of a few hundred milliseconds, and the cognitive games did not experience frame drops. The system exhibited overall smooth behaviour even in back-to-back operations.

6.3.4 Compatibility Testing

Multiple levels of android API and screen sizes were tested to ensure compatibility. The MindCare application was reliable on Android 10+ (including Android 14, 15) and the interface was well adjusted to small, medium, and large screens. No compatibility problems and device-specific crashes were identified.

6.3.5 Load and Stress Testing

Load testing was done to simulate multiple MRI prediction and continuous gameplay to confirm stability throughout extended use. The system was able to perform with consistency and reliability on repeated predictions and save game results correctly even with repeated rapid submissions. The Firebase and backend services were able to service multiple requests simultaneously without degradation.

6.3.6 Exception Handling Testing

The exception cases, like invalid credentials, corrupted MRI images, and the no-internet cases were deliberately invoked. The system delivered relevant user-friendly responses, avoided crashes and maintained data integrity. The upload of invalid MRI images resulted in an obvious error notification, and all Firebase-related errors (auth failures, missing permissions) were properly managed.

6.3.7 Security Testing

Security checking was concerned with access control and Firestore rule enforcement. Unauthenticated users could not access prediction, analytics or game play screens. Firestore rules made sure that users were only allowed to read and write their data. Privilege escalation was not permitted since the admin-only screens were fully restricted.

6.3.8 Installation Testing

The installation of the APK was also tested in the emulator and real devices. Installation was successful and Firebase configuration initiated correctly on the initial launch. Relevant permissions were obtained at the right time, and there were no crashes or unsuccessful installations in the process of testing.

6.4 Summary

This chapter explained the testing, validation, verification and evaluation of MindCare System. The prediction accuracy and responsive game modules and reliable data reflection on the dashboard were validated with quantitative results. The qualitative results indicated that the interface is user-friendly and the information is well displayed. Although small performance degradation is experienced during peak load, and database querying is still not extensive, the system still provides a fast user-friendly experience. The hybrid local-cloud system has practical advantages compared to the standard cloud-only systems [3], and further refinements can be made in terms of wider model support, better concurrency management, and more powerful analytics.

Chapter 7

Conclusions

The MindCare System project was intended to develop and deploy an integrated mobile-based solution to detect and support Alzheimer's disease early using MRI-based prediction, cognitive testing, and user analytics. The chapter highlights the key findings, conclusions, limitations, and possible improvements.

7.1 Summary

The MindCare System manages to provide a fully operational React Native application with secure user authentication, real-time MRI image recognition via a FastAPI back-end, cognitive games to evaluate memory and attention, and analytics based on the recorded user activity on Firebase Firestore. The modules are all interconnected in a sequential manner, allowing users to upload MRI scans, take part in cognitive activities, and track progress in one platform.

The project shows that real-time classification of MRI can be technically implemented on mobile devices using lightweight deep learning models using FastAPI. The cognitive games are effective in capturing the interaction-based data with the potential to track the behavioural trends. React Native, Firebase, and Python API integration was found to be dependable and scalable in mobile health applications.

7.2 System Limitations

The current system would need an active internet connection to perform the authentication, data synchronization, and predictions which restricts its use in offline settings. The app is developed specifically on Android and is only tested on a limited number of device sizes.

Furthermore, the MRI model is also accurate, but a larger and more varied dataset would be helpful in increasing generalization.

7.3 Future Enhancements

7.3.1 Platform Expansion

The system can be made available to a larger pool of users by introducing iOS support and optimizing the interface to meet tablet users.

7.3.2 Offline Accessibility

On-device prediction (TensorFlow Lite or PyTorch Mobile) would permit MRI predictions without being connected to the internet.

7.3.3 Enhanced Cognitive Games

The system can be enhanced with the addition of more scientifically validated cognitive tests to better monitor the performance of the user and identify subtle cognitive changes.

7.3.4 Advanced Analytics

Automation of progress reports, trend analysis, and clinician-ready summaries can make the system more useful in the long-term monitoring.

7.3.5 Addition of Better Models

It could be possible to use more advanced architectures or retraining the model with larger and diverse MRI datasets so as to improve overall accuracy of classification and prediction.

7.3.6 Medical Professional Interface

The inclusion of a specific clinician portal may allow doctors access to patient histories and track progression and issue informed recommendations.

Appendix A

User Manual

This appendix outlines the additional content of MindCare Alzheimer's Disease Detection System. It contains elaborate AI model assessment metrics, sample pictures and the entire project folder layout.

A.1 Sample MRI Images

Figure A.1 below shows sample MRI images from all four Alzheimer's stages used during model training: MildDemented, ModerateDemented, NonDemented, VeryMildDemented.

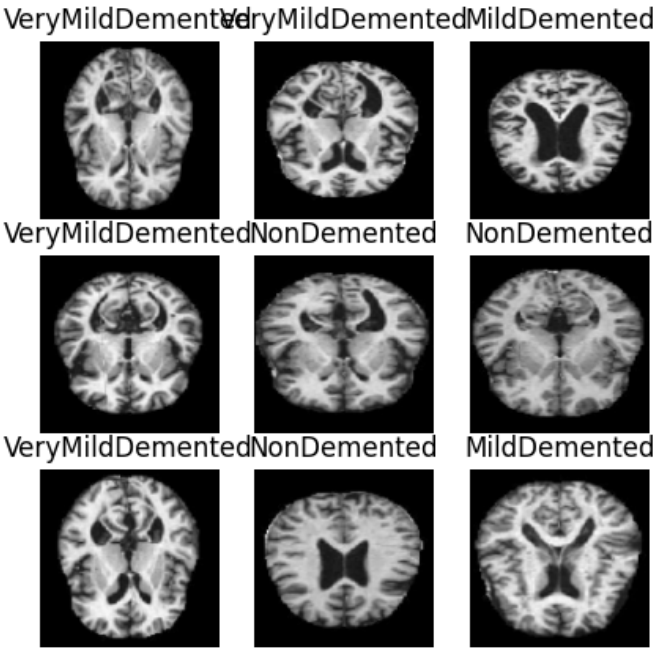

Figure A.1: Sample MRI Images for Each Class

A.2 Data Imbalance Visualization

The dataset is class-imbalanced with the large sample sizes of Non Demented and Very-Mild Demented over the moderate sample size of Moderate Demented. To address this, class weights were computed to be used during training. The distribution can be summarized in the following bar chart figure A.2.

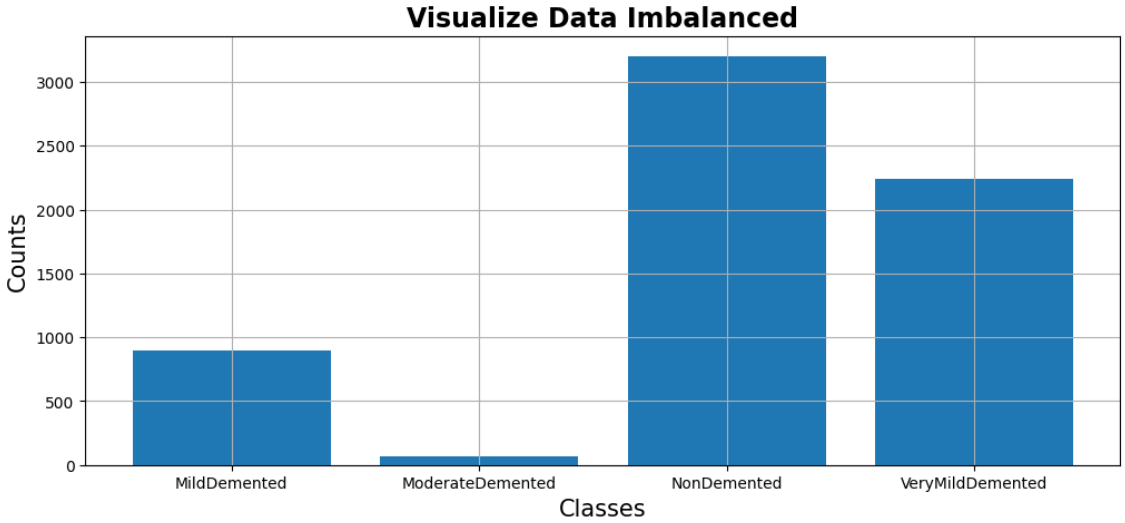

Figure A.2: Class-wise distribution of MRI images (Imbalanced dataset)

Class Weights: Class weights, applied to the imbalanced dataset, are shown in Table A.1.

Table A.1: Class Weights

Class	**Weight**
MildDemented	1.79
ModerateDemented	25.0
NonDemented	0.5
VeryMildDemented	0.71

A.3 AI Model Evaluation Metrics

On the test dataset, Table A.2 shows results of final model.
Test Set Performance:

Table A.2: Test Set Performance

Metric	**Value**
Accuracy	0.9128
Loss	0.3049
AUC	0.9843
Precision	0.9156
Recall	0.9128

Classification Report (Test Set): The classification report on the test set is given in Table A.3.

Table A.3: Classification Report

Class	**Precision**	**Recall**	**F1-Score**	**Support**
MildDemented	0.92	0.89	0.91	91
ModerateDemented	1.00	0.86	0.92	7
NonDemented	0.92	0.94	0.93	320
VeryMildDemented	0.90	0.88	0.89	224

A.4 Test Set Confusion Matrix

The trained ResNet50 model reached 91.2% accuracy of the test, and the confusion matrix is demonstrated in Figure A.3.

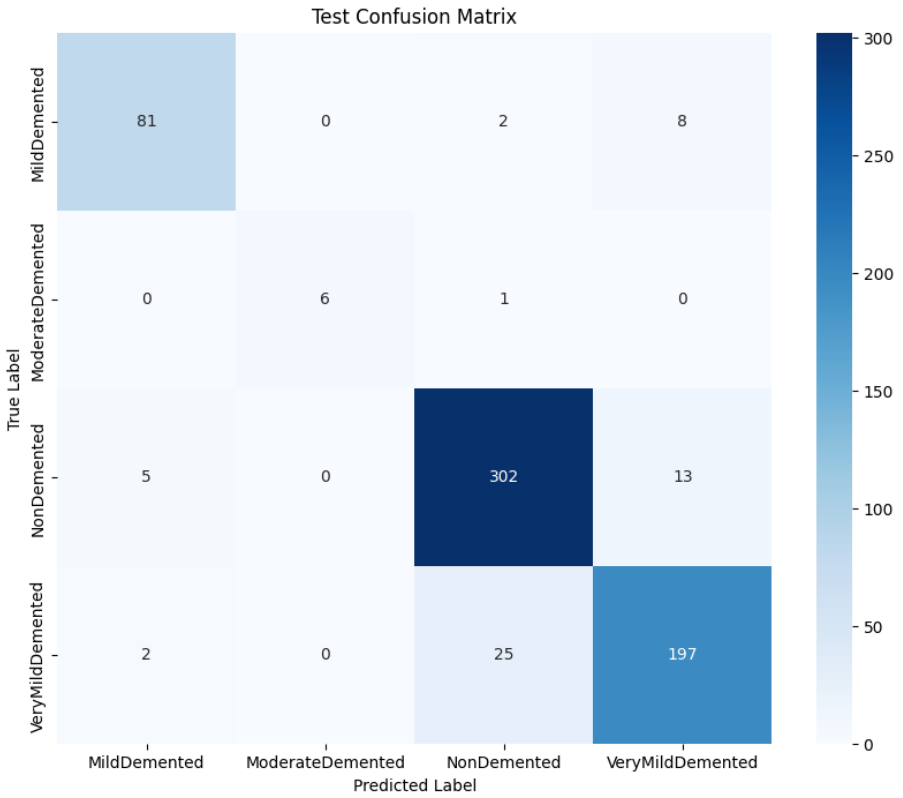

Figure A.3: Test Set Confusion Matrix

Notes:

- Chapter 4 and chapter 5 explain the backend execution, frontend startup commands and Firebase collections.
- This appendix offers further information on reproduction and evaluation of the project.

References

[1] Alzheimer's Association. 2023 alzheimer's disease facts and figures. *Alzheimer's & Dementia*, 19(4):1598–1695, 2023. Cited on p. 1.

[2] World Health Organization. Dementia fact sheet. https://www.who.int/news-room/fact-sheets/detail/dementia, 2023. Accessed 2025. Cited on p. 1.

[3] E. M. Mohammed, A. M. Fakhrudeen, and O. Y. Alani. Detection of alzheimer's disease using deep learning models: A systematic literature review. *Informatics in Medicine Unlocked*, 50:101551, 2024. Cited on pp. 6, 10, and 84.

[4] J. Wen, E. Thibeau-Sutre, M. Diaz-Melo, J. Samper-González, A. Routier, S. Bottani, D. Dormont, S. Durrleman, N. Burgos, O. Colliot, Alzheimer's Disease Neuroimaging Initiative, Australian Imaging Biomarkers, and Lifestyle Flagship Study of Ageing. Convolutional neural networks for classification of alzheimer's disease: Overview and reproducible evaluation. *Medical Image Analysis*, 63:101694, 2020. Cited on p. 6.

[5] S. Basaia, F. Agosta, L. Wagner, E. Canu, G. Magnani, R. Santangelo, M. Filippi, and Alzheimer's Disease Neuroimaging Initiative. Automated classification of alzheimer's disease and mild cognitive impairment using a single mri and deep neural networks. *NeuroImage: Clinical*, 21:101645, 2019. Cited on pp. 6 and 10.

[6] M. Odusami, R. Maskeliūnas, and R. Damaševičius. An intelligent system for early recognition of alzheimer's disease using neuroimaging. *Sensors*, 22(3):740, 2022. Cited on pp. 7 and 9.

[7] M. Mujahid, A. Rehman, T. Alam, F. S. Alamri, S. M. Fati, and T. Saba. An efficient ensemble approach for alzheimer's disease detection using an adaptive synthetic technique and deep learning. *Diagnostics*, 13(15):2489, 2023. Cited on pp. 7 and 8.

[8] H. Shin, S. Jeon, Y. Seol, S. Kim, and D. Kang. Vision transformer approach for classification of alzheimer's disease using 18f-florbetaben brain images. *Applied Sciences*, 13(6):3453, 2023. Cited on p. 7.

[9] S. Korolev, A. Safiullin, M. Belyaev, and Y. Dodonova. Residual and plain convolutional neural networks for 3d brain mri classification. In *IEEE 14th International Symposium on Biomedical Imaging (ISBI)*, pages 835–838, Melbourne, Australia, 2017. Cited on pp. 7 and 9.

[10] M. U. Ali, S. J. Hussain, M. Khalid, M. Farrash, H. F. M. Lahza, and A. Zafar. Mri-driven alzheimer's disease diagnosis using deep network fusion and optimal selection of features. *Bioengineering*, 11(11):1076, 2024. Cited on pp. 7 and 9.

[11] S. Liu, A. V. Masurkar, H. Rusinek, J. Chen, B. Zhang, W. Zhu, C. Fernandez-Granda, and N. Razavian. Generalizable deep learning model for early alzheimer's disease detection from structural mris. *Scientific Reports*, 12(1):17106, 2022. Cited on pp. 7, 8, 9, and 10.

[12] G. Ben-Sadoun, V. Manera, J. Alvarez, G. Sacco, and P. Robert. Recommendations for the design of serious games in neurodegenerative diseases. *Frontiers in Aging Neuroscience*, 10:13, 2018. Cited on pp. 8 and 9.

[13] C. Muscio, P. Tiraboschi, U. P. Guerra, C. A. Defanti, and G. B. Frisoni. Clinical trial design of serious gaming in mild cognitive impairment. *Frontiers in Aging Neuroscience*, 7:26, 2015. Cited on pp. 8 and 9.

[14] Sai Teja Nuka. Cloud-based ai systems for real-time medical imaging analysis and diagnostics. *International Journal of Advanced Research in Computer and Communication Engineering*, 11(12), 2022. Cited on pp. 8 and 9.

[15] Marco Pinamonti. Alzheimer mri 4 classes dataset. https://www.kaggle.com/datasets/marcopinamonti/alzheimer-mri-4-classes-dataset, 2021. Accessed 2025. Cited on p. 40.

16% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.

Filtered from the Report

- Bibliography
- Quoted Text
- Cited Text

Match Groups

212 Not Cited or Quoted 16%
Matches with neither in-text citation nor quotation marks

0 Missing Quotations 0%
Matches that are still very similar to source material

0 Missing Citation 0%
Matches that have quotation marks, but no in-text citation

0 Cited and Quoted 0%
Matches with in-text citation present, but no quotation marks

Top Sources

7% Internet sources
2% Publications
16% Submitted works (Student Papers)