**Poorvaja Sathasivam**

THEraPREDICT: AI Based Patient treatment tracker

Table of Contents

[1. Introduction 3](#_Toc169870881)

[1.1 Project Background 3](#_Toc169870882)

[1.2 Project Aims and Objectives 3](#_Toc169870883)

[2 Requirements Engineering 3](#_Toc169870884)

[2.1 PROBLEM Domain research 3](#_Toc169870885)

[2.1.1 Comparable Software System Review 3](#_Toc169870886)

[2.1.2 DEVELOPMENT Relevant Legislation 3](#_Toc169870887)

[2.2 Requirements Specification 3](#_Toc169870888)

[2.2.1 Problem Domain Description 3](#_Toc169870889)

[2.2.1.1 Existing Business Operation 3](#_Toc169870890)

[2.2.1.2 Summary of Existing Business Limitations 3](#_Toc169870891)

[2.2.2 Functional Requirements 3](#_Toc169870892)

[2.2.2.1 User Management 4](#_Toc169870893)

[2.2.2.2 Patient Management 4](#_Toc169870894)

[2.2.2.3 Treatment Log Management 4](#_Toc169870895)

[2.2.2.4 Predictive Analytics 4](#_Toc169870896)

[2.2.3 NON-FUNCTIONAL Requirements 4](#_Toc169870897)

[2.2.3.1 Performance 4](#_Toc169870898)

[2.2.3.2 Security 4](#_Toc169870899)

[2.2.3.3 Usability 4](#_Toc169870900)

[2.2.3.4 Reliability 4](#_Toc169870901)

[2.2.3.5 Maintainability 5](#_Toc169870902)

[2.2.3.6 Compliance 5](#_Toc169870903)

[2.2.3 Proposed Prototype Development Blocks/Priority of Development 5](#_Toc169870904)

[2.2.3.1 User interface 5](#_Toc169870905)

[3 System Analysis & Design (Software Development Block/Sprint 1) 6](#_Toc169870906)

[Sprint 1: Overview and Initial Setup 6](#_Toc169870907)

[Objectives 6](#_Toc169870908)

[Functional Components 6](#_Toc169870909)

[4 System Build and Technical Notes (Software Development Block/Sprint 1) 9](#_Toc169870910)

[5 System Testing & Evaluation Strategy (Software Development Block/Sprint 1) 9](#_Toc169870911)

[6 CONCLUSIONS 9](#_Toc169870912)

# 1. Introduction

## 1.1 Project Background

The AI-Based Patient Treatment Tracker is designed to assist healthcare professionals in managing and tracking the treatment progress of patients with alcohol and nicotine use disorders. By leveraging advanced deep learning models, this web application aims to predict treatment outcomes and provide personalized recommendations, thereby enhancing the effectiveness of treatment programs.

## 1.2 Project Aims and Objectives

The primary aim of this project is to develop a comprehensive web application that integrates AI functionalities to improve patient treatment outcomes. The specific objectives include:

* Providing a robust patient management system.
* Enabling detailed tracking and logging of treatment sessions.
* Utilizing deep learning for predictive analytics to support personalized treatment plans.
* Ensuring secure, scalable, and user-friendly application deployment.

# 2 Requirements Engineering

## 2.1 PROBLEM Domain research

### 2.1.1 Comparable Software System Review

Several existing systems track patient treatments and provide predictive analytics. These systems often lack integration with advanced AI models and personalized recommendations. This project aims to bridge this gap by incorporating deep learning for more accurate outcome predictions and tailored treatment advice.

### 2.1.2 DEVELOPMENT Relevant Legislation

This application must comply with healthcare regulations such as the Health Insurance Portability and Accountability Act (HIPAA) in the U.S. and the General Data Protection Regulation (GDPR) in the European Union to ensure the privacy and security of patient data.

## 2.2 Requirements Specification

### 2.2.1 Problem Domain Description

### 2.2.1.1 Existing Business Operation

### 2.2.1.2 Summary of Existing Business Limitations

## 2.2.2 Functional Requirements

### 2.2.2.1 User Management

* **Patient Registration**: Allow healthcare professionals to add new patients, capturing essential details like name, age, gender, and contact information.
* **User Authentication**: Secure login for administrators and healthcare professionals to access the system.

### 2.2.2.2 Patient Management

* **View/Edit Patient Information**: Provide functionalities to view and update patient details.
* **Patient History**: Display historical treatment data for each patient, allowing for comprehensive tracking of progress over time.

### 2.2.2.3 Treatment Log Management

* **Add Treatment Logs**: Record details of each treatment session, including date, session specifics, and progress notes.
* **View Treatment Logs**: Offer an interface to view all treatment logs for each patient.

### 2.2.2.4 Predictive Analytics

* **AI Predictions**: Utilize deep learning models to predict treatment outcomes based on patient data.
* **Personalized Recommendations**: Generate treatment recommendations based on predictions and patient history.

## 2.2.3 NON-FUNCTIONAL Requirements

### 2.2.3.1 Performance

* **Scalability**: The application should efficiently handle multiple users and a large number of patient records.
* **Response Time**: The system should provide real-time responses for data entry and retrieval operations.

### 2.2.3.2 Security

* **Data Encryption**: Ensure all patient data is encrypted during both storage and transmission.
* **Access Control**: Implement role-based access control to restrict access to sensitive information.

### 2.2.3.3 Usability

* **User-Friendly Interface**: Design an intuitive and easy-to-use interface for healthcare professionals.
* **Accessibility**: Ensure the application is accessible to users with disabilities.

### 2.2.3.4 Reliability

* **Data Backup**: Implement regular data backups to prevent data loss.
* **Error Handling**: Provide robust error handling and logging mechanisms.

### 2.2.3.5 Maintainability

* **Code Documentation**: Ensure the code is well-documented for future maintenance and updates.
* **Modular Design**: Follow a modular design approach to facilitate easy updates and feature additions.

### 2.2.3.6 Compliance

* **HIPAA Compliance**: Ensure the application meets HIPAA standards for handling patient health information.
* **GDPR Compliance**: Adhere to GDPR regulations for users in the European Union.

## 2.2.3 Proposed Prototype Development Blocks/Priority of Development

### 2.2.3.1 User interface

**Home Page**

* Overview of the application stating the mission and goals.
* Navigation to different web pages – depending on role-based log in.
* The web pages – Login, Patients, Treatment logs, Predictions, Contact.

**Dashboard Page**

* Main landing page after login. Displays role-specific dashboard with relevant information and navigations links.
* Displays user-specific data (appointments, tasks).
* Navigation links to manage users, appointments, profiles, etc.
* Role based content customizations.

**Manage Users Page**

* Admin-only page to manage users (CRUD operations).
* List all users with options to edit, delete, or add new users.
* Form for adding new users with role selection.
* Edit and delete functionalities.

**Manage Appointments Page**

* Healthcare professional's page to manage appointments.
* List all appointments with options to view details, update status, or cancel.
* Ability to add new appointments.
* View patient details associated with appointments.

**Profile Page**

* User profile page for viewing and updating personal information.
* Display user details (name, email, contact info).
* Form to update profile information.
* Change password functionality (if applicable).

**Logout Page**

* Log out the user and destroy the session.
* Clear session variables.
* Redirect to login page after logout.

**Login Page**

* User login page.
* Form for entering email and password.
* Authentication and session initiation upon successful login.
* Error message display for invalid login attempts.

**Register Page**

* User registration page.
* Form for entering registration details (first name, last name, email, password).
* Validation to ensure unique email and secure password storage (using password\_hash()).
* Redirect to login page upon successful registration.

**Deep learning model**

* Train a deep learning model using patient treatment data.
* Features used can include age, treatment duration, number of sessions attended, etc.
* Integrate the trained model into the application.
* Use the model to predict outcomes for new patients and generate recommendations.

# 3 System Analysis & Design (Software Development Block/Sprint 1)

## Sprint 1: Overview and Initial Setup

### Objectives

* Establish the core structure and navigation of the application.
* Implement user authentication and role-based access control.
* Create the foundation for patient management, treatment logs, and predictive analytics.
* Develop a deep learning model for predictive analytics.

### Functional Components

#### 1. Home Page

**Overview of the Application:** A brief introduction stating the mission and goals.

**Role-Based Navigation:** Different navigation options depending on the user's role (admin, healthcare professional, patient).

**Web Pages:** Links to Login, Patients, Treatment Logs, Predictions, Contact.

#### 2. Dashboard Page

**Main Landing Page:** Displays a role-specific dashboard with relevant information and navigation links.

**User-Specific Data:** Show appointments, tasks, and other relevant data for the logged-in user.

**Navigation Links**: Links to manage users, appointments, profiles, etc.

**Role-Based Customization:** Custom content and options based on user roles.

#### 3. Manage Users Page

**Admin-Only Access:** Page for admins to manage users.

**User List:** Display all users with options to edit, delete, or add new users.

**User Form:** Form for adding new users with role selection.

**Edit/Delete Functionalities**: Options to edit or delete existing users.

#### 4. Manage Appointments Page

**Healthcare Professional's Access:** Page for healthcare professionals to manage appointments.

**Appointment List:** Display all appointments with options to view details, update status, or cancel.

**Add Appointments:** Ability to add new appointments.

**Patient Details**: View patient details associated with appointments.

#### 5. Profile Page

**User Profile**: Page for viewing and updating personal information.

**User Details:** Display user details like name, email, and contact info.

**Update Form:** Form to update profile information.

**Change Password:** Functionality to change the password (if applicable).

#### 6. Logout Page

**Logout Functionality:** Log out the user and destroy the session.

**Clear Session Variables:** Ensure session variables are cleared.

**Redirect to Login Page:** Redirect users to the login page after logout.

#### 7. Login Page

**User Login:** Form for entering email and password.

**Authentication:** Verify credentials and initiate session upon successful login.

**Error Handling:** Display error messages for invalid login attempts.

#### 8. Register Page

**User Registration:** Form for entering registration details (first name, last name, email, password).

**Validation:** Ensure unique email and secure password storage (using password\_hash()).

**Post-Registration Redirect**: Redirect to the login page upon successful registration.

#### Deep Learning Model Integration

**Model Training:** Train a deep learning model using patient treatment data (age, treatment duration, number of sessions attended, etc.).

**Model Integration:** Integrate the trained model into the application to predict outcomes.

**Generate Recommendations:** Use the model to predict outcomes for new patients and generate personalized treatment recommendations.

##### Implementation Steps

* Set Up Development Environment
* Install V.je: Set up the environment with V.je.
* Project Directory and MySQL Workbench: Initialize the project directory and database using MySQL Workbench.

##### Develop Frontend

* HTML/CSS/JavaScript: Create the user interface using V.je.
* Navigation and Forms: Implement forms and navigation components.

##### Develop Backend

* V.je Scripts: Handle CRUD operations, user authentication, and session management.
* Role-Based Access Control: Implement role-specific access and content customization.

##### Integrate Deep Learning

* Develop and Train Model: Use Python to develop and train the deep learning model.
* Model Integration: Create an interface for running predictions and saving results.

##### Testing and Deployment

* Functionality Testing: Ensure all components work as expected.
* Performance Testing: Check the application's performance under load.
* Security Testing: Validate data encryption and access control mechanisms.
* Deployment: Deploy the application on a web server.

By the end of Sprint 1, the core functionality of the AI-Based Patient Treatment Tracker should be established, including user authentication, patient management, and initial predictive analytics integration. This provides a solid foundation for further development and refinement in subsequent sprints.

# 4 System Build and Technical Notes (Software Development Block/Sprint 1)

# 5 System Testing & Evaluation Strategy (Software Development Block/Sprint 1)

# 6 CONCLUSIONS