User Integration

Diego Iván Morales Gallardo

October 30, 2024

Contents

1 Introduction

In the field of psychology, managing patient data, including symptoms, prescriptions, and feedback, is crucial for effective treatment. The User Integration System is a comprehensive application that facilitates the management of user (patient) information and leverages artificial intelligence to assist psychologists in generating prescriptions and conducting diagnostic assessments.

This documentation aims to provide developers and users with detailed information on setting up, using, and understanding the underlying architecture of the system.

2 System Overview

The User Integration System consists of two main components:

- **Backend Server**: Handles data storage, API endpoints, and AI integrations.
- **Frontend Client**: Provides a user interface for interacting with the system.

2.1 Architecture

Backend:

- Built with Node.js and Express.js.
- Uses PostgreSQL for database management.
- Implements RESTful API endpoints for CRUD operations.
- Integrates OpenAI API for generating prescriptions.
- Uses Nearbyy API for retrieval-augmented generation in the chat interface.

Frontend:

- Developed using React.js.
- Provides components for user registration, dashboard, and detailed user views.
- Communicates with the backend via HTTP requests.

2.2 Technologies Used

- **Node.js**: Server-side JavaScript runtime.
- **Express.js**: Web framework for Node.js.
- **PostgreSQL**: Relational database management system.
- **React.js**: JavaScript library for building user interfaces.

- **OpenAI API**: Provides AI capabilities for generating prescriptions.
- **Nearbyy API**: Used for semantic search and context retrieval in the chat interface.
- **Axios**: Promise-based HTTP client.
- **Cors**: Middleware for enabling Cross-Origin Resource Sharing.
- **Dotenv**: Loads environment variables from a .env file.

3 Installation Instructions

This section outlines the steps required to set up and run the User Integration System on your local machine.

3.1 Prerequisites

- Node.js (version 14 or higher)
- npm or yarn package manager
- PostgreSQL database
- OpenAI API Key
- Nearbyy API Key

3.2 Backend Setup

1. **Clone the Repository**

```
git clone https://github.com/ivmg5/User-Integration.git
```

Listing 1: Cloning a Git Repository

2. **Navigate to the Backend Directory**

```
cd backend
```

Listing 2: Change Directory to Backend

3. **Install Dependencies**

```
npm install
```

4. **Configure Environment Variables**

Create a `.env` file in the backend directory with the following content:

```
OPENAI_API_KEY=your_openai_api_key
NEARBYY_API_KEY=your_nearbyy_api_key
```

Listing 3: Setting Environment Variables for API Keys

5. **Set Up the Database**

Configure the database connection in `config/db.js` if necessary. Ensure that PostgreSQL is running and a database named `catedraapi` exists.

6. **Run the Server**

```
node index.js
```

Listing 4: Run Node.js Application

3.3 Frontend Setup

1. **Navigate to the Frontend Directory**

```
cd frontend
```

Listing 5: Change Directory to Frontend

2. **Install Dependencies**

```
npm install
```

Listing 6: Install Node.js Dependencies

3. **Run the Frontend Application**

```
npm run dev
```

Listing 7: Run Development Server with npm

4. **Access the Application**
   Open your web browser and navigate to `http://localhost:5173`

4 Usage Instructions

This section describes how to use the application once it is up and running.

4.1 Running the Application

Ensure that both the backend server and the frontend application are running. The backend listens on port 3000, and the frontend runs on port 5173 by default.

4.2 Navigating the Application

4.2.1 Dashboard

Upon accessing the application, you are greeted with the Dashboard, which lists all registered users. The dashboard includes:

- A search bar to filter users by name.
- A ”Register” button to add new users.
- User cards displaying basic information (name and email).
- An option to delete a user (if no associated descriptions or feedbacks exist).

4.2.2 Registering a User

Clicking on the ”Register” button navigates to the registration form. The form requires the following information:

- Name
- Email
- Phone Number
- Address
- Birth Date
- Gender

Fill in the details and submit the form to create a new user.

4.2.3 User Details

Clicking on a user card in the dashboard navigates to the User Details page, which includes several sections:

Viewing User Information Displays all personal information about the user, including name, email, phone number, address, birth date, and gender.

Managing Descriptions and Prescriptions This section allows you to:

- View existing descriptions and prescriptions.
- Generate new prescriptions using AI.
- Save new prescriptions to the user’s record.
- Delete existing descriptions.

Generating Prescriptions To generate a prescription:

1. Enter the patient’s symptoms in the ”Description of Patient Symptoms” text area.
2. Click on ”Generate Prescription.” The system uses OpenAI’s API to generate a prescription based on the input.
3. Review the generated prescription in the ”Generated Prescription” text area.

4. Click on ”Save Prescription” to save it to the user’s record.

Managing Feedback The Feedback section allows you to:

- View existing feedback entries.
- Add new feedback.
- Delete existing feedback.

Interactive Chat The ”Interactive Chat: DSM-5 Diagnostic Criteria” section provides a chat interface where you can:

- Ask questions related to psychological diagnoses.
- Receive responses that consider relevant context retrieved via the Nearbyy API.

5 API Documentation

The backend provides several RESTful API endpoints for interacting with users, descriptions, feedbacks, and chat functionalities.

5.1 User Endpoints

- **GET /users**: Retrieves all users.
- **GET /users/:id**: Retrieves a user by ID.
- **POST /users**: Creates a new user.
- **PUT /users/:id**: Updates an existing user.
- **DELETE /users/:id**: Deletes a user.

5.2 Description Endpoints

- **GET /description/:userId**: Retrieves descriptions for a user.
- **POST /description/:userId**: Creates a new description for a user.
- **PUT /description/:id**: Updates a description.
- **DELETE /description/:id**: Deletes a description.

5.3 Feedback Endpoints

- **GET /feedback/:userId**: Retrieves feedback for a user.
- **POST /feedback/:userId**: Creates new feedback for a user.

- **PUT /feedback/:id**: Updates feedback.
- **DELETE /feedback/:id**: Deletes feedback.

5.4 Chat Endpoints

- **POST /chat**: Generates a response from the AI based on a prompt.
- **POST /chat/context**: Generates a response considering contextual information retrieved via semantic search.

6 Database Schema

The system uses a PostgreSQL database with the following tables:

6.1 Users Table

Column	Type	Description
id	SERIAL PRIMARY KEY	Unique identifier
name	VARCHAR	User's name
email	VARCHAR	User's email
phone$_{number}$	VARCHAR	User's phone number
address	VARCHAR	User's address
birth$_{date}$	DATE	User's birth date
gender	VARCHAR	User's gender

Table 1: User Database Schema

6.2 Description Table

Column	Type	Description
id	SERIAL PRIMARY KEY	Unique identifier
description	TEXT	Symptom description
prescription	TEXT	Generated prescription
user$_{id}$	INTEGER FOREIGN KEY	References Users(id)

Table 2: Medical Records Database Schema

6.3 Feedback Table

Column	Type	Description
id	SERIAL PRIMARY KEY	Unique identifier
feedback	TEXT	Feedback content
user$_id$	INTEGER FOREIGN KEY	References Users(id)

Table 3: User Feedback Database Schema

7 Future Enhancements

Potential improvements to the system include:

- **User Authentication**: Implementing secure login and role-based access control.
- **Data Encryption**: Enhancing data security by encrypting sensitive information.
- **Error Handling**: Improving error messages and handling to enhance user experience.
- **Scalability**: Optimizing the system for handling larger datasets and concurrent users.
- **Additional AI Models**: Incorporating more advanced AI models for better prescription generation.
- **Multilingual Support**: Extending the application to support multiple languages.

8 Conclusion

The User Integration System serves as a comprehensive tool for psychologists to manage patient data effectively while leveraging AI capabilities for prescription generation and diagnostic assistance. This documentation provides a detailed guide to setting up, using, and understanding the system, aiming to facilitate further development and usage.

9 References

- OpenAI API Documentation: `https://beta.openai.com/docs/`
- Nearbyy API Documentation: `https://docs.nearbyy.com/`
- Node.js Official Website: `https://nodejs.org/`

- Express.js Documentation: https://expressjs.com/
- React.js Documentation: https://reactjs.org/
- PostgreSQL Documentation: https://www.postgresql.org/docs/