AI-Powered University Information System
User Manual

Jasmine Morales Tiana Luu Kira Ma Linda Tien

December 2025

Purpose

This user manual provides an overview of the AI-Powered University Information System, including its purpose, goals, required setup steps, and instructions for accessing both the web and backend components. Outlining the system's functional objectives, the manual is designed for developers, administrators, and end users who wish to install, test, or extend the system.

Jira Project Link

Jira Project URL: `https://jasminemiamorales.atlassian.net/jira/software/projects/FP/summary`

Formal Objective Breakdown

The primary objective is to implement the foundation for a centralized AI-powered university information system. This includes:

- Developing a user-friendly web interface for students and faculty to access course, schedule, and faculty information.
- Implementing secure authentication and role-based access for different user groups (students, faculty, administrators).
- Creating a basic display within the web application to allow users to view course catalogs, schedules, and faculty directories.
- Setting up networking components to communicate between the front-end client, back-end server, and database.
- Defining API endpoints on the backend to serve course data, faculty information, and AI-driven recommendations.

Why This System Is Needed

Universities often rely on multiple disconnected systems for course information, faculty directories, and academic planning. This fragmentation creates unnecessary barriers for students and staff and slows down administrative processes. A unified, AI-enhanced information system addresses these challenges by:

- Centralizing all essential academic resources into one platform.
- Reducing confusion and time lost switching between incompatible systems.
- Supporting academic planning through personalized recommendations.
- Improving data consistency across departments.
- Enabling long-term scalability for future automation and analytics tools.

Goals

The AI-Powered University Information System aims to modernize how students and faculty access academic information by transitioning from fragmented systems to a unified, intelligent platform. This digital transformation offers several key benefits:

- **Improved Efficiency:** Centralized access reduces the time spent searching across multiple university systems.
- **Personalized Recommendations:** AI-driven suggestions help students select courses and resources tailored to their academic goals.
- **Enhanced Accessibility:** Responsive design and accessibility features ensure usability across devices and for all users.
- **Real-time Data Access:** Synchronization with university databases ensures up-to-date course schedules and faculty information.
- **Better Decision-Making:** Integrated analytics and recommendations provide actionable insights for students and administrators.

System Requirements

To run or develop the system, ensure the following tools are installed:

- **Node.js and npm** (latest LTS version recommended)
- **React or Angular** framework for the frontend
- **Code Editor:** Visual Studio Code, IntelliJ, or similar
- **Backend Environment:** Node.js or Java (Spring Boot)
- **Database:** PostgreSQL or Microsoft SQL Server
- **Git:** Required for cloning and maintaining project versions

Ensure no other applications are running on ports 3000 (frontend) or 8080 (backend).

How to Download and Access

0.1 Web Application

To launch the front-end server, you will need Node.js and npm installed. A JavaScript framework such as React or Angular is required, along with a code editor (e.g., Visual Studio Code).

1. Open the `university-info-system/client` folder.
2. Run the command: `npm install` followed by `npm start`.
3. The home page can then be accessed at `http://localhost:3000`.

To launch the back-end server, you will need Node.js (or Java if using Spring Boot), a suitable IDE (Visual Studio Code, Eclipse, or IntelliJ), and the required dependencies.

1. Open the `university-info-system/server` folder.
2. Install dependencies: `npm install`.
3. Run the server with: `npm run dev`.
4. The backend server will listen on port 8080 in development mode. Ensure no other processes are using this port.

The backend connects to a PostgreSQL or Microsoft SQL Server database. Ensure the database service is running and accessible.

0.2 Mobile Application (Optional Future Work)

If a mobile application is developed, Android Studio and Java/Kotlin will be required.

1. Import the `university-info-system/mobile` folder as a new project in Android Studio.
2. The Gradle build system will automatically download dependencies.
3. A phone running Android 8.0 (Oreo) or higher is recommended. The emulator can be used, but a physical device is preferred for testing.
4. Connect the device to the same network as the backend server.
5. Build and install the APK via Android Studio: Go to **Build ¿ Build APK**, then install the APK on the device.

Note: After any updates to the database schema, application data may need to be cleared to avoid crashes. This can be done via the Application Manager on the device.