Software Requirement Specification (SRS)

For RentaGo – Car Renting System (AI POWERED)

---

1. Introduction

1.1 Purpose

The purpose of this document is to define the software requirements for **RentaGo**, a car renting system that simplifies the process of renting cars for customers while allowing administrators to manage cars, pricing, and user accounts efficiently. This SRS serves as a reference for developers, testers, and stakeholders throughout the system's lifecycle.

1.2 Scope

RentaGo is a software solution designed to:

- Allow **customers** to register, log in, browse cars, view details, and rent vehicles.
- Enable **administrators** to manage cars, set rental pricing, approve/reject user accounts, and monitor transactions.
- Provide **AI-powered suggestions** to recommend suitable cars based on customer needs (budget, distance, trip type).

The system will initially be implemented as a **CLI-based application** but is designed for scalability into **web and mobile platforms**.

1.3 Definitions, Acronyms, and Abbreviations

- **CLI** – Command Line Interface
- **AI** – Artificial Intelligence
- **Admin** – Administrator of the system
- **User** – Customer who rents cars

1.4 References

- IEEE Standard 830-1998: Recommended Practice for Software Requirements Specifications
- OpenAI / Hugging Face free-tier APIs / OpenRouter / Internal Basic AI implementation for AI integration

---

2. Overall Description

2.1 Product Perspective

RentaGo is a **standalone system** that interacts with a database for persistent storage and may utilize external AI APIs for recommendation features. It is modular and designed to scale to support additional features such as online payments and mobile app support.

2.2 Product Functions

- **Customer Functions:** Registration, login/logout, browsing cars, viewing details, renting cars, and viewing rental history.
- **Admin Functions:** Car management, rental price management, user account approval, transaction monitoring, and reporting.
- **AI Module:** Car recommendations based on user inputs such as trip type, budget, and distance.

2.3 User Classes and Characteristics

- **End Users (Customers):** Individuals seeking to rent cars. Expected to have minimal technical knowledge.
- **Administrators:** Staff managing cars and customer accounts. Require full control and reporting capabilities.

2.4 Operating Environment

- **Platform:** Windows/Linux/Mac (CLI-based)
- **Programming Language:** Java / Python
- **Database:** SQLite/MySQL
- **AI Integration:** Free-tier APIs (e.g., OpenAI, Hugging Face)
- **Hardware Requirements:** Standard computer with at least 4 GB RAM, 1 GHz processor

2.5 Design and Implementation Constraints

- Initial release limited to CLI interface.
- Internet required for AI module.
- Must use lightweight, free database for storage (e.g., SQLite).

2.6 Assumptions and Dependencies

- Users will provide valid information during registration.
- AI recommendations depend on availability of external APIs.
- System assumes reliable internet access for cloud features.

---

3. System Features and Requirements

3.1 Functional Requirements

User Module

- Register account with unique username and password.
- Login/Logout with authentication.
- Browse and search available cars.
- View car details (model, type, price per day/hour).
- Rent cars for specific duration.
- View rental history.
- Receive AI-based recommendations.

Admin Module

- Add/update/delete car records.
- Set and update rental prices.
- Approve/reject customer registrations.
- View all rental transactions.
- Generate basic reports (e.g., most rented car, earnings report).

3.2 Non-Functional Requirements

- **Performance:** The system should handle at least 100 concurrent transactions.
- **Security:** Passwords stored in encrypted format. Admin access restricted.
- **Usability:** CLI commands must be simple and intuitive.

- **Reliability:** Data must be preserved in case of unexpected shutdowns.
- **Portability:** System should run on multiple OS platforms.
- **Scalability:** Should allow migration to web/mobile platforms in the future.

---

4. External Interface Requirements

4.1 User Interfaces

- CLI-based menus for navigation (text-based).
- Clear error messages and success confirmations.

4.2 Hardware Interfaces

- Local machine with keyboard input and display output.

4.3 Software Interfaces

- Database: SQLite/MySQL
- AI API: OpenAI/Hugging Face free-tier

---

5. Other Requirements

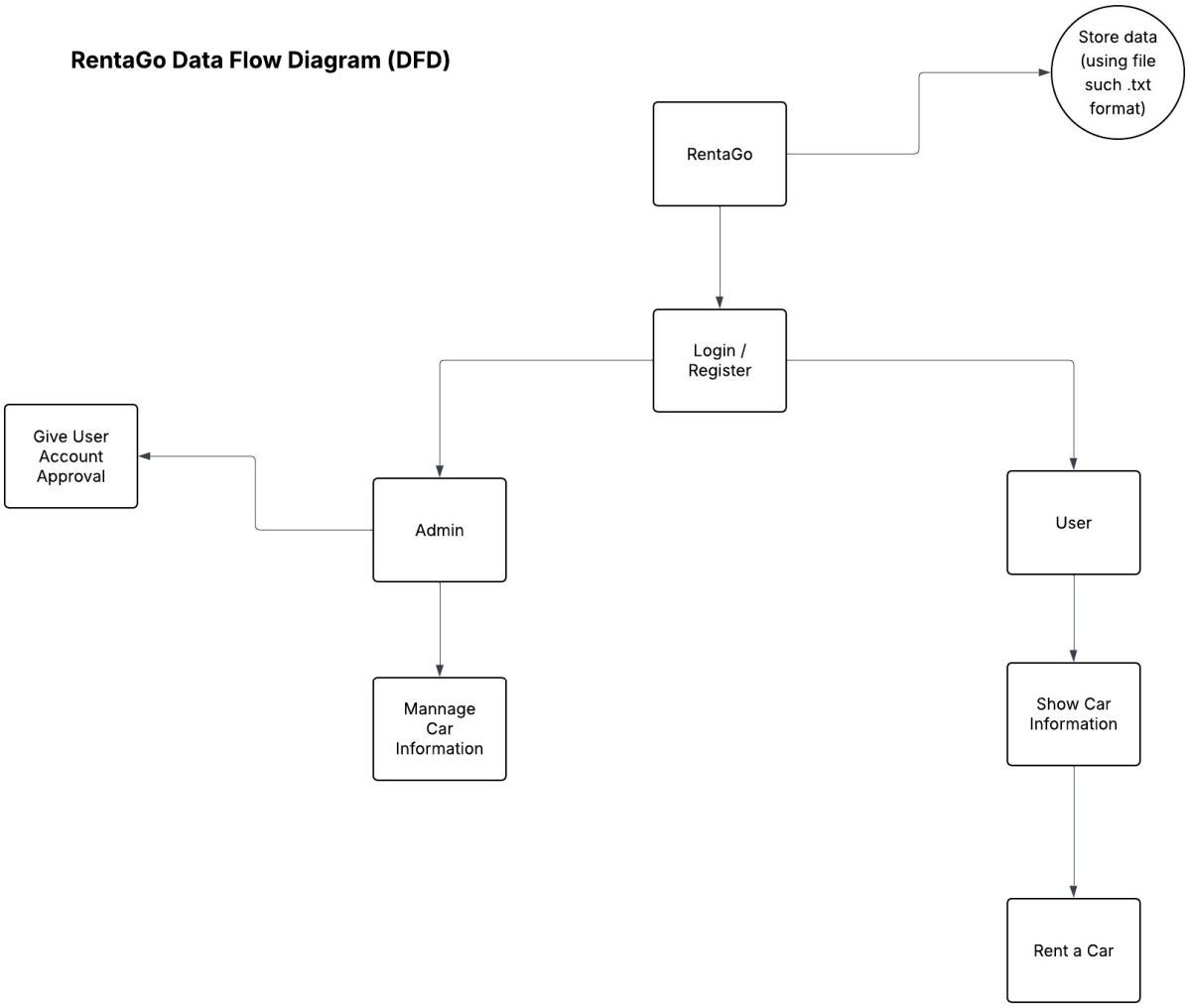
- Data backup should be automated daily.
- Logs of transactions should be maintained for auditing.

---

6. Future Enhancements

- Web and mobile application support.
- Online payment gateway integration.
- Real-time car availability and GPS tracking.
- Dynamic AI-driven pricing strategies.

RentaGo Data Flow Diagram (DFD)



RentaGo Use Case Diagram