Maintenance & Support Plan

Maintenance Strategy

The rental system for cars has a regular upkeep approach to provide sustained dependability and flexibility. Efficiency enhancement, improvements to the product, and frequent bug tracking are all part of this. Preventive (bug-fix patches), adaptable (environment modifications), and effective (improvements caused by user's feedback) upkeep tasks are divided into three categories. To keep track of all modifications and assure accuracy, a version-driven repo is kept up to date.

- The following changes are planned:
- Adding support for consecutive rentals to the reservation code
- Improving edge case correction processes
- Modules reworking for greater scaling

Versioning

Logical versions are used by the algorithm to explicitly express the purpose of alterations:

• Important version: Adds noteworthy improvements or fundamental modifications
• Minor version: includes improvements that are reversible.
• Patch version: Resolves minor problems or glitches

For instance, the first release, v1.0.0 Version 1.1.0: Included an adaptive price module; Version 1.1.1: Resolved an issue in the booking status updates

Backward Compatibility

The system has the following features to maintain confidentiality of information across updates:
Security tests to guarantee modular consistency; stable API endpoints to avoid interfering with CLI operations; and migrate scripts for changing structure of databases
In order to accommodate upcoming improvements without compromising present capabilities or user data, retrogression is given priority.

Project Management

SDLC Choice and Rationale

Following the Agile Software Development Life Cycle approach, the Car Renting System was developed. Agile was chosen because of its incremental structure, which facilitates initial evaluation of key features, modular adoption, and ongoing improvement. This method gave flexibility to integrate feedback and improvements while enabling the gradual release of services including registration of users, vehicle control, and reservation routines.
The modules that were the focus of each sprint were authorization for users, automobile supply, reservation logic and management functions. During the development cycle, agile concepts including continual integration, list sorting, and time-bound iterations been used.

Planning, Estimation, and Tracking

An abridged Kanban sheet was used for organizing the project and classifying jobs into "To Do," "Ongoing Development," and "Finalized." Each module was scheduled for execution within a one-week sprint, and estimates were calculated according to relationships and complexities.
A task checklists and software versioning changes were used to personally maintain progress, guaranteeing openness and reliability throughout the development process.

Risk Management and Mitigation

Few issues were notified during the planning phase, along with regarding mitigation strategies:

Risk	**Impact**	**Mitigation**
Incomplete feature implementation	Medium	Key improvements were given priority initially throughout the rush period.
Dependency conflicts	High	using simulated settings and requirements.txt
Data loss / corruption	Medium	Implemented validation and backup routines
CLI usability issues	Low	tested efficiency and improved instructions.
Time constraints	High	Effectively identified functionality and prevented perspective creep