# Non-Functional Requirements

This section outlines the **quality attributes**, **constraints**, and **expectations** that define how the **Syarti** system should perform. These are critical to ensure **reliability**, **accuracy**, **usability**, and **security** of the system, especially for **internal simulation**

## Performance Requirements

* The system is a **web-based responsive application** accessible on both desktop and mobile platforms.
* As this is a **front-end prototype**, the system *will not be subjected* to real-world load; hence, *no live* **concurrent** **users**.
* **Real-time fleet** operations (like *adding, updating, booking, and returning cars*) must occur without **noticeable delay**, prioritizing **accuracy** over speed.
* **Page** and **component interactions** (e.g., *car booking, form submissions*) are expected to respond **instantly (within 1 second)** in development/test environments.

## Safety Requirements

* **Auto-save functionality** *is expected* when handling data-entry forms (e.g., *booking, car info updates*).
* A **daily backup mechanism** *is expected* to be **simulated** as a critical **safety routine** for **data preservation**.
* The system *must be able to recover*from **crashes** or **power failures**, ensuring minimal **data loss** during operations.
* The system *must maintain* **data consistency** after unexpected **interruptions** (e.g., *transactions must be atomic* when **booking** or **returning** **cars**).

## Security Requirements

* **Basic authentication** required for all roles (**Admin***,* **Customer***,* **Accountant***,* **Legal****Verifier***,* **Fleet****Manager***,* **Pickup/Drop-off Handler**).
* **Password hashing** should be considered for *future implementation*.
* Sensitive data such as **payment information** and **driver license numbers** should be identified and marked for **future encryption**
* Each user role should have **clearly defined permissions**, such as:
  + ***Customers*:** browse, book, return cars.
  + ***Admin*:** full access.
  + ***Fleet******Manager*:** manage cars.
  + ***Legal******Verifier*:** verify documents.
  + ***Accountant*:** view transactions.
  + ***Pickup/Drop-off Handler*:** manage car handovers.

## Software Quality Attributes

### Reliability

* The system *should run without failures* during a complete **rental** **transaction** **cycle** (*book, return, record update*).
* All system features *should remain functional* after standard **user** **actions**.

### Maintainability

* The system **code** *should be* **modular** and **documented** to allow easy **future** **improvements**.
* Source **code** *should* *follow* *consistent* **naming** **conventions** and **comments** for readability.

### Scalability

* The system *is* *not* *intended* *to* *scale* for high **user** **traffic** but must support **logical** **extension** (e.g., *adding new user roles or views*).

### Portability

* The system *must* *be* *accessible* across **different** **devices** (*desktops, tablets, smartphones*) using modern browsers.

### Usability

* The system *should* *feature* a **clean**, **intuitive** **interface** for all **users**.
* **Users** *should be able to learn* **basic** **operations** (*booking, managing cars, reviewing status*) without external guidance.

### Business Rules

* **Users** *may rent* **multiple** **vehicles** simultaneously.
* **Late** **returns** *incur* additional **charges** based on *extended time*.
* **Users** with a **history** of **late** **returns** are *deprioritized* during **booking** **conflicts** (added to a *low-priority list*).
* There are *no rules on* **minimum** or **maximum** **rental** **durations**.