Rental Property Management System

**Index**

* **Project Overview**
  + Description of the problem being solved.
  + Objectives and scope of the project.
  + Target users and stakeholders.
  + Overview of the proposed solution.
* **Functional Requirements**
  + Detailed functional requirements of the system.
  + Use cases and user stories covering key features.
  + Non-functional requirements.
  + Prioritized list of requirements.
* **System Architecture**
  + High-level architecture diagram.
  + Description of key components and technologies.
  + Database schema and descriptions.
  + File structure and descriptions.
* **UML Diagrams**
  + Use case diagrams.
  + Class diagrams.
  + Sequence diagrams for critical user workflows.
  + State chart diagrams.
* **Implementation**
  + Source code.
  + User interface.
  + Testing.
  + Installation and deployment instructions.
  + Conclusion.
  + Summary.
  + Potential enhancements and future work.
* **Acknowledgements**
* **References**

**Description of the problem**

The management of rental properties presents a complex set of difficulties that needs to be addressed efficiently and effectively. These challenges can span across different aspects of the system, including but not limited to:

1. **Property Management:** The task of managing all the different properties with a vast variety of attributes like location, size, type, or status can be overwhelming. This also includes keeping track of property maintenance certificates and tenant information within each property.
2. **Rent Collection:**  The process of collecting rent from each property in a timely manner can tedious and time-consuming. This is further complicated by late payments and tracking payment history.
3. **Communication:** Effective communication between tenants and landlords is very crucial for smooth operation. However, this can be complicated due to the lack of a centralised system to send notifications, updates, and other important information.
4. **Handling Complaints:** Addressing maintenance request effectively and in a timely manner is essential for tenant satisfaction. Having said that, tracking and managing these complaints from different properties can be a daunting task.
5. **Tedious Paperwork:** Renting a property is subject to many laws and regulations. So, Landlord or management team is often required to keep different documents regarding their properties and tenants. Storing these documents, safe and secure is a task on its own.

**Objectives and Scope**

The primary objective of this project is to develop and deploy a reliable system, which can handle all the different operational difficulties of landlords or management agencies.

This project aims to provide a seamless system for managing different properties in a user-friendly style to improve the efficiency and efficacy of the process. Tenants will be provided with a robust system to make rent payments, provide documentation, and to request maintenance. Additionally, the ability to have centralised control on properties, tenants and employees can be implemented depending on the users needs. The capability to raise, track and manage maintenance requests will be added.

Likewise, a reliable system will be put in place to handle the collection and storage of different documents form tenants and certifications of the property, in a secure location. A robust system to generate contracts and receipts will be provided as well.

Overall, the aim of the project is to completely digitalise the process and to improve the communication between landlords and tenants.

As the project is developed as part of TCA 2 of my module, so the project is to be produced in one month. The main objective is to produce a software with the basic functionality implemented.

**Target users and Stakeholders**

The primary users of this application are property owners or property management firms who are looking for a more reliable, lightweight, and cheaper alternative to the current option. The stakeholders involved are some rental management firms who wants to fully digitalise the process. The application will then be launched for others to use for a fee.

**Overview of the proposed solution.**

The aim is to develop a desktop application using Java. Frontend will be made using Java Swing. The application will be able to add, remove, and update users, properties, staff, and tenants. Additionally, the application will provide tenants with a portal to make payments, submit maintenance requests and track them, see their contract, and payment history. The user will be able to issue contracts, add or remove or update details regarding tenants, staff, and property under their management.

**Detailed functional requirements of the system.**

Tenants:

: Log in using tenant id and password.

: Make payments securely.

: Add a late fee if payment is late. (depending on local laws)

: Get invoice and email confirmation of payment.

: See payment history.

: Request for maintenance whenever they need.

: Track maintenance request submitted by them.

: View and download a copy of their contract.

Landlords:

: Log in using landlord id and password.

: Add, delete, or update information regarding their property.

: Add, delete, or update information regarding their tenants.

: Add, delete, or update information regarding their staff.

: View details of rent payments, including who, amount etc.

: receive, process, and update maintenance requests.

: issue, view, and terminate contracts.

: View and edit staff timesheets.

: Process staff leave application.

: Schedule inspections and give feedback to tenants.

: Submit problems to admin.

Employee:

: Log in using employee id and password.

: Clock in and out for shifts and breaks.

: Submit requests for day off.

Admin

: Log in using Admin id and password.

: Add, remove, or update details regarding landlords.

: Add, delete, or update information regarding their property.

: Add, delete, or update information regarding their tenants.

: Add, delete, or update information regarding their staff.

: Access to error logs.

:

**Use cases and user stories covering key features.**

**Non-functional requirements.**

Explains the limitations of the system.

How it is done.

* User Interface
* Reliability
* Security
* Performance
* Maintenance
* Standards

**Prioritized list of requirements.**

: