SRS example

Computer Engineering (University of Mumbai)

Software Requirements Specification

For

Rental Property Management System

Version 1.0 approved

Prepared by

Archit Shinde

Girik Shroff

Kashita Talreja

Thadomal Shahani Engineering College

16th February, 2021

Table of Contents

Revision History

Name	Date	Reason For Changes	Version

1. Introduction

1.1 Purpose

The purpose of the software is to provide buyers and sellers of property a platform to buy, sell and rent property. Normally both buyers and sellers rely on brokers to help them find perfect buyer or seller for the property. But this costs money to both the buyer and seller. To overcome this cost a software solution is necessary to help people.

1.2 Document Conventions

Important points have been underlined to provide emphasis. Headings and Subheadings have been written in bold font to provide emphasis. The points in all sections have been written in the order of their priority, from higher priority points to lower priority points, so that important points are not missed out. Abbreviations are used in some places which will be understood by the developers of the application.

1.3 Intended Audience and Reading Suggestions

The intended audience is the team of developers who will be designing and implementing the Property Management System. Also, the document is to be utilized by the testing team who will be testing and evaluating the performance and design of the application. The document consists of all the necessary information that will be required by the team of software engineers who will be working on the project.

1.4 Product Scope

The Rental Property Management System aims to simplify the process of renting and leasing properties for its users. To make it accessible on all electronic devices the application will be a web-based application and will be mobile responsive as much as possible. The application will provide all necessary functionalities for searching properties, adding properties and their images. There will also be functionalities for posting advertisements for tenants of properties who are in search for roommates. In future, we plan to add additional functionalities and features in the application which include a payment gateway for transactions to take place within the application itself. We also plan to add an online rental agreement feature which will reduce the efforts taken for an agreement between the tenant and owner.

1.5 References

Websites:

- Django Documentation: https://docs.djangoproject.com/en/3.1/
- Bootstrap Framework: https://getbootstrap.com/

2. Overall Description

2.1 Product Perspective

The Rental Property Management System is intended to provide an alternative to the common way of buying, selling and renting properties which involves brokers. This way proves to be expensive for both the parties involved. The features of the application will allow the users to conveniently search for properties as well as buyers for properties. This eliminates the

need of having to avail the services of a property agent or broker, thus saving money for both the parties. Also, the contact is directly between the buyer and the seller and so clarifications and agreements can be done directly without the involvement of a third party.

2.2 Product Functions

- By using the Rental Property Management System, the need of a third party in the process of buying and selling properties is eliminated.
- Users can search properties in different areas and locations without having to travel to the location.
- Direct communication between the buyer and seller of the property. Clarifications can be done easily.
- Saves time for both the parties.
- Multiple available properties and buyers to choose from.

2.3 User Classes and Characteristics

The users of the application can be classified into two types. The ones who want to give a property on lease (property owners) and the ones who want to rent a property (tenants). There is no strict distinction between the two types of users. Both the users can access all the functionality of the application.

2.4 Operating Environment (OE)

Since the application is a web application it can work on any device having a browser.

- Device: Mobile Phone, Computer, Laptops, Tablets.
- Operating System: Windows, Linux distributions, Mac OS, Android
- RAM: 128 MB or more

- Disk Space: 20 MB or more.
- Browsers: Mozilla Firefox 30+, Google Chrome 27.0+, Microsoft Edge. Other browsers can also be used.
- Internet connection: Strong internet connection with speed of at least 1 Mbps for best experience.

2.5 Design and Implementation Constraints

CO-1:

The time allotted for this project is at most 3 months.

CO-2:

The front end of the application will be made using HTML, CSS and JavaScript.

CO-3:

Python will be used as the language for the backend of the application and PostgreSQL will be used for the database of the application.

CO-4:

The website will be in English language. Users who do not know English will face difficulties in using the website.

2.6 User Documentation

Appropriate instructions will be provided at every step in the application to ensure the users do not face any difficulties while using the application. In future, we plan to add a chatbot to guide users in case they face any difficulties. Instructions will be given while filling out forms, adding photos and locations. Proper error messages will be displayed in case the user

inadvertently fills wrong information or makes any mistake while using the application.

2.7 Assumptions and Dependencies

AS-1:

The application supports only English language. We assume the users of the application will be well versed with English.

AS-2:

The users of the application should have basic knowledge of uploading images and location.

DE-1:

The application will require Django web framework as a dependency, since we use Python as the backend language.

DE-2:

Bootstrap Framework will be used for the front end of the application.

DE-3:

For maps and geolocation, we will be using Mapbox APIs.

3. External Interface Requirements

3.1 User Interfaces

UI-1:

The website will start with a landing page. The landing page will have all information about the web application.

UI-2:

There will be a navigation bar at the top of the web page which will help users to navigate to different web pages.

UI-3:

Instructions will be provided to the users on top of forms to be filled.

UI-4:

There will be alerts and pop ups which appear in case the user makes a mistake while using the application.

UI-5:

The interface will be responsive for all screen sizes as much as possible to provide the users a seamless experience.

3.2 Hardware Interfaces

N/A

3.3 Software Interfaces

- Browsers: Mozilla Firefox 30+, Google Chrome 27.0+ are the preferred browsers.
- Operating System: Android, Windows 7, 8, 10, Mac OS, Linux distributions.

3.4 Communication Interfaces

The application will be using HTTPS protocol.

4. System Features

4.1 Authentication and Authorization

4.1.1 Description and Priority:

The application will be having multiple users and so authentication becomes a high priority system feature. The application will be using Django's authentication module in order to implement this functionality. When the user creates a new account on the application, they will have to provide their email address and password. The password must be at least 8 characters long and must have at least one uppercase character, one digit and one special character. The passwords in the system will be hashed and stored so that no other person can get to know the password.

4.1.2 Response Sequences:

Once the user registers in the application, they will be guided to a login page where they will have to enter their email address and password to login. After successful login, the user will be redirected to the landing page of the application. There will also be a logout button on the navigation bar. On clicking the logout button, the user will be logged out.

4.1.3 Functional Requirements:

REQ-1: We use Django's authentication module for authentication and authorization functionality. The authentication will be session-based authentication.

4.2 Add Property Feature

4.2.1 Description and Priority:

Feature to add properties on the website will be provided to the user (owner in this case). The owner will have to provide all specification about the property which he intends to put on lease. The feature is of high priority as the application is based on owners adding properties on the website to get more potential buyers.

4.2.2 Response Sequences:

The user will have to provide all the necessary information about the property. On filling out all the information the user will be redirected to a page where he will have the option to add photographs of the property. If there are any errors in the description of the property, they will be pointed out using alerts to notify the user.

4.2.3 Functional Requirements:

REQ-1: The information provided will be validated using JavaScript on the client side of the application.

4.3 Add Photographs Feature:

4.3.1 Description and priority:

The user gets to add photographs of the property which he intends to put on lease. This is again a high priority feature. The user gets the option to choose images using file system or simply drag and drop the pictures into the provided area.

4.3.2 Response Sequences:

The user can add images using the drag and drop functionality or simply choose images from the files that he wants to upload. In case any image exceeds the maximum file size the user will be notified

using alerts. On adding the images, the user is redirected to the add location page.

4.3.3 Functional Requirements:

REQ-1: For the drag and drop functionality we use Dropzone JavaScript library.

4.4 Add location Feature

4.4.1 Description and Priority:

The user gets to add the location of the property by using the map. The user will simply have to add a marker by clicking on the location where the property is located. The priority of the feature is moderate.

4.4.2 Response Sequences:

On adding the location, the property is successfully added to the database and the user is redirected to the landing page of the website.

4.4.3 Functional Requirements:

REQ-1: Mapbox APIs will be used for geolocation functionality.

4.5 Search Properties Feature:

4.5.1 Description and Priority:

The user can search different properties using the search functionality. This feature is of high priority and is primarily for tenants. The user can search based on his requirements such as desired location, area, number of bedrooms etc.

4.5.2 Response Sequence:

On filling the required information based on desired features, the results will be displayed on the webpage.

4.5.3 Functional Requirements:

REQ-1: Mapbox APIs will be required for displaying locations of the properties on the map.

4.6 Shortlisting Properties Feature

4.6.1 Description and Priority:

This feature will allow users to shortlist or save properties that they are interested in, for future reference. This is a moderate priority feature.

4.6.2 Response Sequence:

Once the user clicks on the shortlist button the property will be added to the list of properties that have been shortlisted by that particular user.

4.6.3 Functional Requirements:

REQ-1: For shortlisting properties, AJAX calls will be used. Therefore, JQuery will be used on the client side.

4.7 Delete Properties Feature

4.7.1 Description and Priority:

This feature will allow the owner of the property to remove the property from the system. This is a moderate priority feature.

4.7.2 Response Sequence:

Once the user clicks on the delete property button, a popup will appear asking the user if he is sure if he wants to delete the property. If the user response if Yes, then the property will be removed from the system.

4.7.3 Functional Requirements:

REQ-1: For deleting properties, AJAX calls will be used. Therefore, JQuery will be used on the client side.

4.8 Deleting Shortlisted Properties Feature

4.8.1 Description and Priority:

This feature will allow users to remove properties shortlisted by them. This feature is of moderate priority.

4.8.2 Response Sequence:

Once the user clicks on the remove button the property will be removed from the list of properties that have been shortlisted by that particular user.

4.8.3 Functional Requirements:

REQ-1: For removing shortlisting properties, AJAX calls will be used. Therefore, JQuery will be used on the client side.

4.9 Add Advertisement Feature

4.9.1 Description and Priority:

Feature to add advertisements on the website will be provided to the user (tenants in this case). This feature will be used by tenants who are looking for room partners. This is a moderate priority feature.

4.9.2 Response Sequences:

The tenant is required to provide all the necessary specifications about the property in which he is living currently along with a minimum of one photograph of the property. The user on submitting the details will be redirected to the landing page.

4.9.3 Functional Requirements:

REQ-1: The information provided will be validated using JavaScript on the client side of the application.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

5.1.1 Scalability:

The application should be scalable and should perform without any interruption for all the users.

5.2 Safety Requirements

- Backup power supply should be present for server, so that it does not stop functioning in case of power failure.
- API keys of the APIs used should not be made open source.
- Code backup should be taken at regular time intervals.

5.3 Security Requirements

- The passwords of the users are hashed and then stored in the database so that no person can access the passwords of the users.
- The passwords should be at least 8 characters long and must have at least one uppercase character, one digit and at least one special symbol.
- The website should HTTPS protocol for security.

- POST requests are used for transferring information regarding authentication, adding properties, adding advertisements etc. through forms.

5.4 Software Quality Attributes

5.4.1 Usability:

The user interface should be simple to use and not cluttered with a lot of information.

5.4.2 Availability:

- The system should be available at all times.
- The system should be reliable and there should be no loss of data in case the server breaks down when operations are going on.

5.4.3 Maintainability:

The code for the application should be written cleanly and should be well documented. The code should contain comments to help new programmers and developers make changes in the application.

5.4.4 Testability:

The code should be written with proper test cases to be tested upon so that no errors during production take place.

5.5 Business Rules

The administrator of the application has full permission of controlling the system.

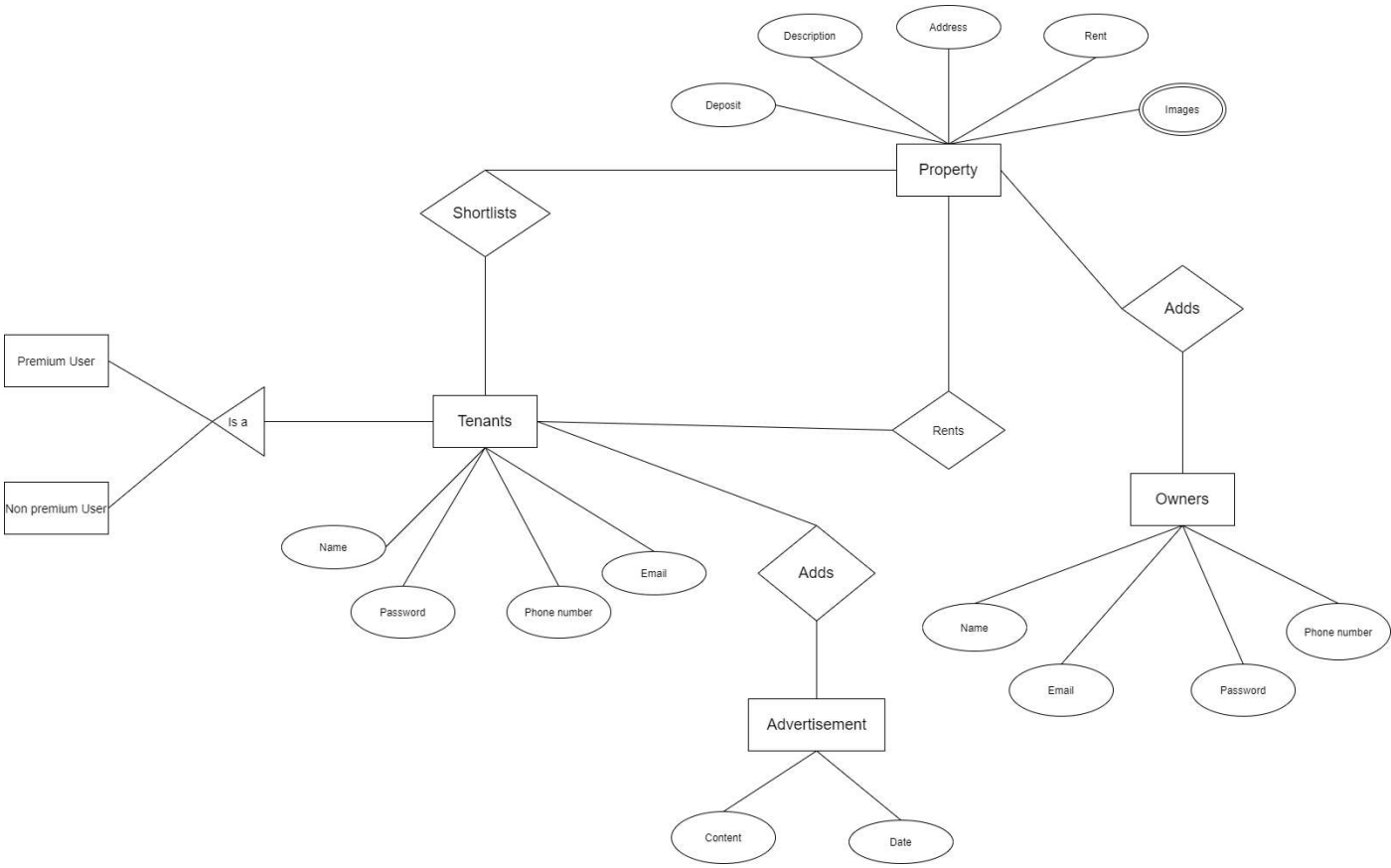
6. Other Requirements

Appendix A: Glossary

- HTTPS: Hypertext Transfer Protocol Secure
- API: Application Programming Interface
- GUI: Graphical User Interface

Appendix B: Analysis Models

ER Diagram of the Application



Use Case Diagram for the application