UITS

utu ill b b an thy past

An initiative of PHP Family

University of Information Technology & Sciences

CSE 356

Software Engineering and System Analysis Lab

House Renting System

Software Requirements Specification

Version 1.0

May 27, 2025

Team Members:

Sazzad Hossain (0432220005101103)

A. O. M. Ramim Chowdhury (0432220005101146)

Suraia Akter Mim (0432220005101108)

Samin Yeaser Rafid (043231000510110)

Prepared for

Khandoker Nosiba Arifin

Lecturer

Department of CSE, UIT

Table of Contents

1. Introduction

The House Renting System Software Requirements Specification (SRS) document provides a comprehensive overview of the web-based house renting platform. This document contains all information needed by software engineers to adequately design and implement the rental management system that connects landlords, property managers, and tenants through an efficient online platform.

1.1 Purpose

The purpose of this SRS is to define the functional and non-functional requirements for the House Renting System. This document is intended for software developers, system architects, testers, project managers, and stakeholders involved in the development and deployment of the rental management platform. It serves as a contractual agreement between the development team and stakeholders regarding system capabilities and constraints.

1.2 Scope

This subsection identifies the House Renting System software product and its capabilities:

Software Product: Web-based House Renting Management System

What the software will do:

- Provide an online platform for property listing and management
- Enable tenant property search and rental application submission
- Automate rental processes including application tracking and lease management
- Offer role-based dashboards for landlords, property managers, and tenants
- Integrate with third-party services for background checks and credit verification
- Support mobile-responsive access across devices

What the software will not do:

- Handle physical property maintenance or repairs
- Provide legal advice or services
- Process actual financial transactions (integrates with external payment systems)
- Manage commercial real estate properties (initial scope limited to residential)

Application Description: The system targets residential rental properties including single-family homes, apartments, duplexes, and vacation rentals. Benefits include streamlined property management, improved tenant experience with 24/7 platform access, automated administrative processes, and data-driven decision making through analytics and reporting capabilities.

1.3 Definitions, Acronyms, and Abbreviations

- **SRS:** Software Requirements Specification
- **MFA:** Multi-Factor Authentication
- **UI:** User Interface
- **API:** Application Programming Interface
- **CSS:** Cascading Style Sheets
- **HTML:** HyperText Markup Language
- **MySQL:** Relational Database Management System
- **Landlord:** Property owner who rents out properties
- **Tenant:** Individual seeking or renting a property
- **Property Manager:** Professional who manages properties on behalf of landlords
- **Lease Agreement:** Legal contract between landlord and tenant

1.4 References

1. IEEE Guide to Software Requirements Specification (ANSI/IEEE Std. 830-1984)
2. WSU-TC CptS 322 Course Materials
3. Web Development Best Practices Documentation
4. MySQL Database Documentation
5. Bootstrap Framework Documentation

1.5 Overview

This SRS document is organized into five main sections:

- Section 1 provides an introduction and overview of the document
- Section 2 presents the general description of the system including product perspective and user characteristics
- Section 3 details specific functional and non-functional requirements
- Section 4 describes analysis models including use cases and activity diagrams
- Section 5 outlines the change management process for requirement updates

2. General Description

This section describes the general factors affecting the House Renting System and its requirements, providing context without stating specific technical requirements.

2.1 Product Perspective

The House Renting System is a standalone web-based application that operates independently while integrating with external services. The system interfaces with:

- Third-party background check services for tenant verification
- Credit reporting agencies for financial assessment
- Email services for automated notifications
- Payment gateways for transaction processing
- Cloud storage services for document and image management

The system is designed to be self-contained with its own database, user management, and core functionality while leveraging external APIs for specialized services.

2.2 Product Functions

The House Renting System provides the following major functions:

- **Property Management:** Create, update, and manage rental property listings
- **User Registration and Authentication:** Secure account creation and login for all user types
- **Property Search and Filtering:** Advanced search capabilities for tenants
- **Rental Application Processing:** Online application submission and tracking
- **Tenant Screening:** Integration with background check and credit services
- **Dashboard Management:** Role-based interfaces for different user types
- **Maintenance Request Tracking:** System for managing property maintenance issues
- **Lease Agreement Generation:** Digital lease creation and management
- **Communication Tools:** Messaging system between landlords and tenants
- **Reporting and Analytics:** Data insights for property performance

2.3 User Characteristics

Primary Users:

- **Tenants:** Individuals aged 18-65 seeking rental properties, varying technical expertise levels, access system via multiple devices
- **Landlords:** Property owners aged 25-70, moderate to high technical skills, manage multiple properties
- **Property Managers:** Professional property management companies, high technical proficiency, manage large property portfolios

Secondary Users:

- **System Administrators:** Technical staff responsible for system maintenance and user support

All users expect intuitive interfaces, reliable performance, and secure data handling. Mobile accessibility is crucial as users frequently access the system on-the-go.

2.4 General Constraints

Technical Constraints:

- Web browser compatibility requirements (Chrome, Firefox, Safari, Edge)
- Mobile responsive design constraints
- Database performance limitations with MySQL
- Third-party API rate limits and availability

Regulatory Constraints:

- Fair housing law compliance requirements
- Data privacy regulations (CCPA, GDPR considerations)
- Local rental law compliance features

Business Constraints:

- Budget limitations for third-party service integrations
- Timeline constraints for initial system deployment
- Scalability requirements for future expansion

2.5 Assumptions and Dependencies

Assumptions:

- Users have reliable internet connectivity
- Modern web browsers are available to all users
- Third-party services maintain consistent API availability

- Users possess basic computer literacy skills

Dependencies:

- Availability of third-party background check services
- Credit reporting agency API access
- Email service provider reliability
- Web hosting infrastructure stability
- MySQL database system compatibility
-

3. Specific Requirements

This section provides detailed functional and non-functional requirements that guide system design, implementation, and testing.

3.1 External Interface Requirements

3.1.1 User Interfaces

- **Responsive Web Interface:** Compatible with desktop, tablet, and mobile devices
- **Minimum Screen Resolution:** 320px width for mobile compatibility
- **Browser Support:** Chrome 90+, Firefox 88+, Safari 14+, Edge 90+
- **Accessibility:** WCAG 2.1 AA compliance for users with disabilities
- **Loading Time:** Pages must load within 3 seconds on standard broadband
-

3.1.2 Hardware Interfaces

- **Client Hardware:** Standard computing devices (smartphones, tablets, laptops, desktops)
- **Server Hardware:** Web server with minimum 8GB RAM, 4-core processor
- **Storage Requirements:** Minimum 100GB for database and file storage
- **Network Interface:** Standard HTTP/HTTPS web protocols

3.1.3 Software Interfaces

- **Operating System:** Linux-based web server environment
- **Database:** MySQL 8.0 or higher
- **Web Server:** Apache or Nginx

- **Development Environment:** VS Code, XAMPP for local development
- **Third-party APIs:** Background check services, credit reporting APIs

3.1.4 Communications Interfaces

- **HTTP/HTTPS Protocols:** Secure web communication
- **Email SMTP:** Automated notification system
- **API Communications:** RESTful APIs for third-party integrations
- **WebSocket Support:** Real-time messaging capabilities

3.2 Functional Requirements

3.2.1 User Registration and Authentication

3.2.1.1 Introduction: System must provide secure user registration and login capabilities for all user types.

3.2.1.2 Inputs: Email address, password, user type selection, personal information

3.2.1.3 Processing: Validate input data, encrypt passwords, create user accounts, send verification emails

3.2.1.4 Outputs: User account creation confirmation, login success/failure messages

3.2.1.5 Error Handling: Invalid email format errors, password strength requirements, duplicate account prevention

3.2.2 Property Listing Management

3.2.2.1 Introduction: Landlords must be able to create, edit, and manage property listings. **3.2.2.2 Inputs:** Property details, images, rental terms, pricing information

3.2.2.3 Processing: Validate property information, store images, update availability status

3.2.2.4 Outputs: Published property listings, listing management interface

3.2.2.5 Error Handling: Image upload failures, invalid pricing data, incomplete property information

3.2.3 Property Search and Filtering

3.2.3.1 Introduction: Tenants must be able to search and filter available properties.

3.2.3.2 Inputs: Search criteria (location, price range, property type, amenities)

3.2.3.3 Processing: Query database, apply filters, sort results

3.2.3.4 Outputs: Filtered property list, detailed property pages

3.2.3.5 Error Handling: No results found messages, invalid search parameters

3.2.4 Rental Application Processing

3.2.4.1 Introduction: System must handle online rental application submission and tracking. **3.2.4.2 Inputs:** Tenant application data, supporting documents

3.2.4.3 Processing: Store application data, notify landlords, track application status

3.2.4.4 Outputs: Application confirmation, status updates, approval/rejection notifications **3.2.4.5 Error Handling:** Incomplete applications, document upload failures

3.3 Use Cases

3.3.1 Use Case #1: Submit Rental Application

Actor: Tenant **Precondition:** Tenant is logged in and has selected a property

Main Flow:

1. Tenant navigates to property details page
2. Tenant clicks "Apply Now" button
3. System displays application form
4. Tenant fills out personal and financial information
5. Tenant uploads required documents
6. Tenant submits application
7. System validates application data
8. System sends confirmation to tenant and notification to landlord **Postcondition:** Application is stored and landlord is notified

3.3.2 Use Case #2: Manage Property Listing

Actor: Landlord **Precondition:** Landlord is logged in **Main Flow:**

1. Landlord accesses property management dashboard
2. Landlord selects "Add New Property" or selects existing property
3. System displays property form
4. Landlord enters property details and uploads images
5. Landlord sets rental terms and pricing
6. Landlord publishes listing
7. System validates information and makes property searchable **Postcondition:** Property is available for tenant search

3.4 Classes / Objects

3.4.1 User Class

3.4.1.1 Attributes:

- userID (unique identifier)
- email (login credential)
- password (encrypted)
- userType (tenant, landlord, admin)
- firstName, lastName
- phoneNumber
- registrationDate

3.4.1.2 Functions:

- authenticate()
- updateProfile()
- resetPassword() *References: FR 3.2.1*

3.4.2 Property Class

3.4.2.1 Attributes:

- propertyID (unique identifier)
- landlordID (foreign key)
- address
- propertyType
- rentAmount
- availabilityStatus
- description
- amenities
- images

3.4.2.2 Functions:

- createListing()
- updateListing()
- setAvailability() *References: FR 3.2.2*

3.5 Non-Functional Requirements

3.5.1 Performance

- System must support 1000 concurrent users
- Database queries must execute within 2 seconds
- Image uploads must complete within 30 seconds
- Page load times must not exceed 3 seconds
-

3.5.2 Reliability

- System uptime must be 99.5% minimum
- Mean Time Between Failures (MTBF) > 720 hours
- Automated backup procedures every 24 hours
- Error recovery procedures for system failures

3.5.3 Availability

- System accessible 24/7 with planned maintenance windows
- Maximum downtime of 4 hours per month for maintenance
- Geographic redundancy for disaster recovery

3.5.4 Security

- All passwords must be encrypted using industry-standard algorithms
- Multi-factor authentication for sensitive operations
- HTTPS encryption for all data transmission
- Role-based access control implementation
- Regular security audits and vulnerability assessments

3.5.5 Maintainability

- Modular code architecture for easy updates
- Comprehensive system documentation
- Automated testing procedures
- Version control system implementation

3.5.6 Portability

- Cross-browser compatibility
- Mobile device responsiveness
- Database portability to other MySQL-compatible systems

3.6 Inverse Requirements

- System shall not store unencrypted sensitive personal information
- System shall not allow unauthorized access to user data
- System shall not process payments directly (uses external payment gateways)

3.7 Design Constraints

- Must use HTML5, CSS, Bootstrap, and JavaScript for frontend
- Backend must be implemented in JavaScript
- Database must be MySQL
- Development environment limited to VS Code and XAMPP
- Responsive design required for mobile compatibility

3.8 Logical Database Requirements

Database System: MySQL 8.0+ **Data Formats:** UTF-8 character encoding for international character support **Storage Capabilities:** Minimum 100GB with expansion capability **Data Retention:** User data retained for 7 years, inactive accounts purged after 2 years **Data Integrity:** ACID compliance, foreign key constraints, data validation rules **Backup Requirements:** Daily incremental backups, weekly full backups

3.9 Other Requirements

- Email notification system for application updates
- Integration capability with third-party background check services
- Multi-language support consideration for future expansion
- Analytics and reporting dashboard for landlords

4. Analysis Models

The following analysis models were used to develop the specific requirements and system design.

4.1 Use Case Diagram

Introduction: The use case diagram illustrates the interactions between different actors and the system functionalities.

Actors:

- Tenant: End users seeking rental properties
- Landlord/Property Manager: Property owners and managers
- Admin: System administrators
- Payment Gateway: External payment processing system

Primary Use Cases:

- Register/Login (All users)
- Search Properties (Tenant)
- Submit Application (Tenant)
- Manage Listings (Landlord)
- Process Applications (Landlord)
- Generate Reports (Admin)

4.2 Activity Diagram

Introduction: Activity diagrams model the workflow processes within the system.

Example Process: Rental Application Submission

1. Start: Tenant logs into system
2. Browse available properties

3. Select desired property
4. Fill application form
5. Upload required documents
6. Submit application
7. System validates data
8. Send notification to landlord
9. Landlord reviews application
10. Decision: Approve or Reject
11. Send notification to tenant
12. If approved: Generate lease agreement
13. End: Process complete

Narrative Description: This activity diagram traces the complete rental application process from initial tenant login through final lease generation, showing decision points and system interactions.

4.3 Data Flow Diagrams (DFD):

Level 0 DFD: Shows the system as a single process with external entities (Tenants, Landlords, Third-party Services) and data flows (Applications, Property Data, Verification Results).

Level 1 DFD: Decomposes the system into major processes:

1. User Management Process
2. Property Management Process
3. Application Processing
4. Reporting and Analytics

5. Change Management Process

Change Request Submission:

- Changes may be submitted by project stakeholders, development team members, or end users
- All change requests must be submitted through the designated project management system
- Requests must include rationale, impact assessment, and priority level

Change Approval Process:

1. Initial review by Lead Software Engineer
2. Impact analysis on existing requirements and system design
3. Stakeholder review and approval for significant changes
4. Documentation update requirements
5. Implementation timeline estimation

Change Implementation:

- Approved changes result in SRS document updates
- Version control maintained for all document revisions
- Affected team members notified of requirement changes
- Testing procedures updated to reflect new requirements

Change Documentation:

- All changes tracked in revision history
- Change rationale documented for future reference
- Impact on project timeline and budget recorded

Appendices

A.1 Appendix 1: Glossary of Terms

Additional definitions and technical terms used in the system

A.2 Appendix 2: Initial Conceptual Documents

References to preliminary system design documents and stakeholder meeting minutes