Requirements Engineering

Property Rental System

Submitted By: Adam O’Mahony ( T00227115 )

Computing with Software Development

Date Submitted: 05/12/2022

Table of Contents

[Introduction/overview 4](#_Toc120979879)

[1. Functional Components 5](#_Toc120979880)

[2. User Requirements 6](#_Toc120979881)

[3.1. PropertySYS will record property Owners 6](#_Toc120979882)

[3.2. PropertySYS will manage Properties 6](#_Toc120979883)

[3.3. PropertySYS will record Tenants 6](#_Toc120979884)

[3.4. PropertySYS will manage Rentals 6](#_Toc120979885)

[3.5. PropertySYS will perform administrative reporting 6](#_Toc120979886)

[3. System Requirements 7](#_Toc120979887)

[4.1. System Level Use Case Diagram 8](#_Toc120979888)

[4.2. Manage Owner’s. 9](#_Toc120979889)

[4.2.1. Add Owner 9](#_Toc120979890)

[4.2.2. Update Owner 11](#_Toc120979891)

[4.3. Manage Properties(Use active Verb + Noun) 13](#_Toc120979892)

[4.3.1. Add Property Type 13](#_Toc120979893)

[4.3.2. Add Property 16](#_Toc120979894)

[4.3.3. Update Property 19](#_Toc120979895)

[4.3.4. Update Property Type 22](#_Toc120979896)

[4.4. Manage Tenant’s. 24](#_Toc120979897)

[4.4.1. Add Tenant. 24](#_Toc120979898)

[4.4.2. Update Tenant 26](#_Toc120979899)

[4.5. Manage Rentals. 29](#_Toc120979900)

[4.5.1. Rent Property. 29](#_Toc120979901)

[4.5.2. Update Rental 31](#_Toc120979902)

[4.5.3. Record Payment 33](#_Toc120979903)

[4.6. Produce Administrative Reports. 35](#_Toc120979904)

[4.6.1. Produce Yearly Commission Report. 35](#_Toc120979905)

[4.6.2. Produce Rentals in Year Report 37](#_Toc120979906)

[4. System Model 39](#_Toc120979907)

[5.1. Level-0 DFD 39](#_Toc120979908)

[5.2. Level-1 DFD 39](#_Toc120979909)

[5.3. Level-2 DFD (Process P1: Manage Owners) 40](#_Toc120979910)

[5.4. Level-2 DFD (Process P2: Manage Properties) 40](#_Toc120979911)

[5.5. Level-2 DFD (Process P3: Manage Tenants) 41](#_Toc120979912)

[5.6. Level-2 DFD (Process P4: Rentals) 41](#_Toc120979913)

[5.7. Level-2 DFD (Process P5: Administration Reports)) 42](#_Toc120979914)

[5. Data Model 43](#_Toc120979915)

[6.1. Class Diagram 43](#_Toc120979916)

[6.2. Relational Schema 44](#_Toc120979917)

[6.3. Database Schema: PropertySYS 45](#_Toc120979918)

[6. Conclusion 47](#_Toc120979919)

[7. Appendices 48](#_Toc120979920)

[8.1. Appendix A – Declaration Of Originality. 48](#_Toc120979921)

# Introduction/overview

This project represents the entirety of a property rental system of a local estate agency, the agency is owned and operated by the main Estate Agent.

The system represents the requirement for all the functions of the business including setting up owner and tenant profiles, Property Details, Rental Contracts, Incoming Payments, & Administrative reports.

Some business rules have been implemented based on the Estate agent’s requests and business requirements.

The system will streamline the process of registering new tenants and owners and property details into the system as well as allowing for ease of file management for the estate agency.

The System will also prompt the estate agent where necessary to ensure the correct data types are entered and that the business rules are followed.

A prototype of the system has been created and will be included alongside this document to give a visual and tactile representation of the system in action.

The Methodology for the approach to creating this system follows the Waterfall Method. While this approach is slower than the agile method, its advantages lie in the documentation and the step by step approach to creating a functional product.

# Functional Components

This System is a Local Property Rental System for managing all aspects of a small to medium estate agency that manages property rentals for owners of property.

# User Requirements

## PropertySYS will record property Owners

* + 1. PropertySYS will add owner profiles.
    2. PropertySYS will update owner profiles.

## PropertySYS will manage Properties

* + 1. PropertySYS will add a property type.
    2. PropertySYS will add a property.
    3. PropertySYS will update a property type.
    4. PropertySYS will update a property.

## PropertySYS will record Tenants

* + 1. PropertySYS will add tenant profiles.
    2. PropertySYS will update tenant profiles.

## PropertySYS will manage Rentals

* + 1. PropertySYS will rent properties
    2. PropertySYS will update rental contracts.
    3. PropertySYS will record rental payments.

## PropertySYS will perform administrative reporting

* + 1. PropertySYS will produce a yearly Commission Report
    2. PropertySYS will produce a Rentals in Year report.

# System Requirements

The system should take into account all aspects of the process of renting a property and managing the profiles of Owners and Tenants as well as generation of administrative reports.

Owner profiles will be created for payment of rent and for contact information.

The Properties grouping will be able to add property profiles as well as keep track of property types & status of the properties.

Tenants’ profiles will keep track of information relating to the tenants.

Rentals will encapsulate all areas relating to the rental of properties to tenants as well as keeping track of payments from the tenants.

The Administration grouping will relate to the generation of financial and statistical reports for use in business strategies and planning.

## System Level Use Case Diagram

The following system level use case diagram illustrates the high-level system requirements.

Diagram

Description automatically generated

## Manage Owner’s.

### Add Owner

This Function will Add Owner’s details to the owner’s data store. It will record all the details of the owner that are relevant to our system.

|  |
| --- |
| **Use Case:** *Add Owner* |

Estate Agent

<<includes>>

<<includes>>

<<includes>>

<<includes>>

Owner

<<extends>>

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Add Owner | |
| **Use Case Id** |  | |
| **Priority** | 1 | |
| **Source** | Estate Agent | |
| **Primary Business Actor** | Estate Agent | |
| **Other Participating Actors** | Owner | |
| **Description** | This function will add owner’s details to the system. | |
| **Preconditions** |  | |
| **Trigger** |  | |
| **Expected Scenario** | **Estate Agent** | **System** |
|  | **Step 1:** Invoke add owner  **Step 3:** Estate Agent enters owner’s data:   * firstName. * lastName. * phoneNumber. * email * eircode. * IBAN | **Step 2:** Display add owner User interface.  **Step 4:** Validate data:   * All fields must not be empty * phoneNumber must be numeric and be valid format. * email must be valid format. * eircode must be valid format * IBAN must be valid.   **Step 5:** Set owners’ status to Active (‘A’)  **Step 6:** Assign Owner an owner ID.  **Step 7:** Save data in Owners file.  **Step 8:** Display confirmation message.  **Step 9:** Reset UI. |
| **Alternate Scenarios** | **Actor** | **System** |
| **Invalid Data Entered:** |  | **Step 4:** Invalid data entered.  **Step 5:** Display appropriate error message.  **Step 6:** Return to step 3. |

### Update Owner

This function will update the owner’s details on the owners’ data file.

|  |
| --- |
| **Use Case:** *Update Owner* |

Estate Agent

<<includes>>

<<includes>>

<<includes>>

<<includes>>

Owner

<<extends>>

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Update Owner | |
| **Use Case Id** |  | |
| **Priority** | 2 | |
| **Source** | Estate Agent | |
| **Primary Business Actor** | Estate Agent | |
| **Other Participating Actors** | Owner | |
| **Description** | This function will update owner’s details on the system. | |
| **Preconditions** |  | |
| **Trigger** |  | |
| **Expected Scenario** | **Estate Agent** | **System** |
|  | **Step 1:** Invoke Update owner  **Step 3:** Agent enters owner surname.  **Step 6:** Agent Selects appropriate owner.  **Step 8:** Agent edits owner’s data:   * firstName. * lastName. * phoneNumber. * email * eircode. * IBAN * Status | **Step 2:** Display Update owner User interface.  **Step 4:** Retrieve all owner data with matching surname.   * firstName * lastName * phoneNumber * ownerID   **Step 5:** Load UI with data.  **Step 7:** Retrieve the full owner details and load into UI.   * firstName. * lastName. * phoneNumber. * email * eircode. * IBAN * Status   **Step 9:** Validate data:   * All fields must be entered * phoneNumber must be numeric and be valid format. * email must be valid format. * eircode must be valid format * IBAN must be valid. * Status can either be Active (‘A’) or Inactive(‘I’).   **Step 10:** Update data in Owners file.  **Step 11:** Display confirmation message.  **Step 12:** Reset UI. |
| **Alternate Scenarios** | **Actor** | **System** |
| **No Data Found:**  **Invalid Data Entered:** |  | **Step 4:** No matching data found.  **Step 5:** Display appropriate error message.  **Step 6:** Return to step 3.  **Step 9:** Invalid data entered.  **Step 10:** Display appropriate error message.  **Step 11:** Return to step 8. |
| **Conclusions** | Owner’s details have been updated in the owners data file. | |
| **Post conditions** | Owner’s details are now corrected. | |
| **Business Rules** | Owners status can be Active or Inactive.  Owners must provide an IBAN must be valid format as it is required for payments / rental contracts. | |
| **Implementation Constraints** |  | |

## Manage Properties(Use active Verb + Noun)

Manage the Properties Registered on the system and add new Properties to the system.

### Add Property Type

This function will add Property Type’s to the database to be attached to property details.

Estate Agent

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Add Property Type | |
| **Use Case Id** |  | |
| **Priority** | 1 | |
| **Source** | Estate Agent | |
| **Primary Business Actor** | Estate Agent | |
| **Other Participating Actors** |  | |
| **Description** | This Function records a property type | |
| **Preconditions** |  | |
| **Trigger** |  | |
| **Expected Scenario** | **Estate Agent** | **System** |
|  | **Step 1:** Estate agent invokes the Add Property Type function.  **Step 3:** Estate Agent Enters Required Data:   * typeCode (2 characters) * Description | **Step 2:** Display Add Property Type User Interface.  **Step 4:** Validate Data Entered:   * All fields must be entered. * typeCode must not already exist. * typeCode must be 2 characters in length. * Description must not be numeric.   **Step 5:** Save data in property types file:   * typeCode. * Description.   **Step 6:** Display confirmation message.  **Step 7:** Reset User Interface. |
| **Alternate Scenarios** | **Actor** | **System** |
| **Invalid Data Entered:** |  | **Step 4:** Invalid data detected.  **Step 5:** Display appropriate error message.  **Step 6:** Return to step 3. |
| **Conclusions** | The property type has been added to the system. | |
| **Post conditions** | Properties can now be assigned this property type. | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

### Add Property

This Function will Add Properties to the Properties Data Store, it will record all property Details.

<<includes>>

Estate Agent

Owner

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Add Property | |
| **Use Case Id** |  | |
| **Priority** | 1 | |
| **Source** | Estate Agent | |
| **Primary Business Actor** | Estate Agent | |
| **Other Participating Actors** | Owner | |
| **Description** | This function will add Property Details to system. | |
| **Preconditions** |  | |
| **Trigger** |  | |
| **Expected Scenario** | **Estate Agent** | **System** |
|  | **Step 1:** Invoke add property  **Step 4:** Estate Agent Searches by Owner Surname.  **Step 7:** Estate Agent enters Property data:   * Select Appropriate owner. * Select PropertyType. (from list) * houseName. (text) * Eircode (Text) * rentalPrice. * Property Description. (text) * Total Rooms. (number) * Standard bedrooms. (number) * Ensuite bedrooms. (number) * Bathrooms. (number) * Garden space? (yes / no). * Parking Spaces? (number). * Pets allowed? (yes/ no). * Heating source (text). * Wifi Included? (yes/No). * Owner Occupied? (Yes/No) | **Step 2:** Retrieve Property type Details from PropertyTypes file and Lod into UI.  **Step 3:** Display the AddProperty User interface.  **Step 5:** Retrieve List of Owners with surname entered from the Owners file and load into UI.  **Step 6:** Display Add Property Details.  **Step 8:** Validate property data:   * House Name or Number must be entered. * Property type must be selected. * Owner must be selected. * Eircode must be valid. * rentalPrice must be positive number. * Description must be entered. * Total Rooms be positive numeric & minimum 1. * Bedrooms must be positive numeric. * Bedrooms cannot exceed total rooms. * Ensuite bedrooms must be positive numeric. * Ensuite bedrooms cannot exceed total bedrooms & cannot exceed total bathrooms. * Bathrooms must be positive numeric. * Bathrooms, bedrooms cannot exceed total rooms. * Parking spaces must be positive numeric. * Heating source must be entered.   **Step 9:** Setproperty status to Available.  **Step 10:** Save data in Properties file.  **Step 11:** Display confirmation message.  **Step 12:** Reset UI. |
| **Alternate Scenarios** | **Actor** | **System** |
| **Search Field Returns None:** |  | **Step 5:** No owners with searched surname found.  **Step 6:** display error message.  **Step 7:** return to step 4. |
| **Invalid Data Entered:** |  | **Step 8:** Invalid data entered.  **Step 9:** Display appropriate error message.  **Step 10:** Return to step 7. |
| **Conclusions** | This Property’s details have been added to the system. | |
| **Post conditions** | This Property added can now be rented. | |
| **Business Rules** | Property Status can be Available, Rented, or Unavailable.  Properties must have at least 1 room to be added. | |
| **Implementation Constraints** |  | |

### Update Property

This Function will update Properties in the Properties Data Store, it will record all property Details.

Estate Agent

Owner

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Update Property | |
| **Use Case Id** |  | |
| **Priority** | 1 | |
| **Source** | Estate Agent | |
| **Primary Business Actor** | Estate Agent | |
| **Other Participating Actors** | Owner | |
| **Description** | This function will add Property Details to system. | |
| **Preconditions** |  | |
| **Trigger** |  | |
| **Expected Scenario** | **Estate Agent** | **System** |
|  | **Step 1:** Invoke Update property  **Step 3:** Estate Agent enters Eircode for property to update.  **Step 6:** Estate Agent Updates Properties Data:   * Select Property Owner. * Select PropertyType. (from list) * houseName. (text) * Property Status (text) * rentalPrice. * Property Description. (text) * Total Rooms. (number) * Standard bedrooms. (number) * Ensuite bedrooms. (number) * Bathrooms. (number) * Garden space? (yes / no). * Parking Spaces? (number). * Pets allowed? (yes/ no). * Heating source (text). * Wifi Included? (yes/No). * Owner Occupied? (Yes/No) | **Step 2:** Display Update Property UI.  **Step 4:** Retrieve the properties details from the Properties data file.  **Step 5:** Populate Update Property UI with this data.  **Step 7:** Validate property data:   * House Name or Number must be entered. * Property type must be selected. * Owner must be selected. * rentalPrice must be positive number. * Status must be ‘A’ Available , ‘R’ Rented, or ‘U’ Unavailable. * Description must be entered. * Total Rooms be positive numeric & minimum 1. * Bedrooms must be positive numeric. * Bedrooms cannot exceed total rooms. * Ensuite bedrooms must be positive numeric. * Ensuite bedrooms cannot exceed total bedrooms & cannot exceed total bathrooms. * Bathrooms must be positive numeric. * Bathrooms, bedrooms cannot exceed total rooms. * Parking spaces must be positive numeric. * Heating source must be entered.   **Step 8:** Setproperty status to Available.  **Step 9:** Update data in Properties file.  **Step 10:** Display confirmation message.  **Step 11:** Reset UI. |
| **Alternate Scenarios** | **Actor** | **System** |
| **Eircode Search Field Returns None:** |  | **Step 4:** No Property with eircode found.  **Step 5:** display error message.  **Step 6:** return to step 3. |
| **Owner Search Field Returns None:** |  | **Step 6:** No Owner’s with typed surname found.  **Step 7:** display error message.  **Step 8:** return to step 6. |
| **Invalid Data Entered:** |  | **Step 7:** Invalid data entered.  **Step 8:** Display appropriate error message.  **Step 9:** Return to step 6. |
| **Conclusions** | This Property’s details have been updated on the system. | |
| **Post conditions** | This Property now has correct data. | |
| **Business Rules** | Property Status can be Available, Rented, or Unavailable.  Properties must have at least 1 room to be added. | |
| **Implementation Constraints** |  | |

### Update Property Type

This function will Update Property Type’s data on the Property Type data file.

Estate Agent

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Update Property Type | |
| **Use Case Id** |  | |
| **Priority** | 2 | |
| **Source** | Estate Agent | |
| **Primary Business Actor** | Estate Agent | |
| **Other Participating Actors** |  | |
| **Description** | This Function updates the data for a property type on the property type data file. | |
| **Preconditions** |  | |
| **Trigger** |  | |
| **Expected Scenario** | **Estate Agent** | **System** |
|  | **Step 1:** Estate agent invokes the Update Property Type function.  **Step 4:** Estate Agent selects the property type to edit.  **Step 6:** Estate Agent Enters Required Data:   * Description | **Step 2:** Retrieve property type data from property type data store:   * typeCode * Description   **Step 3:** Display Update Property Type User Interface.  **Step 5:** Populate UI with the data of the type selected.  **Step 7:** Validate Data Entered:   * Description cannot be empty. * Description must not be numeric.   **Step 8:** Update data in property types file:   * Description.   **Step 9:** Display confirmation message.  **Step 7:** Reset User Interface. |
| **Alternate Scenarios** | **Actor** | **System** |
| **Invalid Data Entered:** |  | **Step 7:** Invalid data detected.  **Step 8:** Display appropriate error message.  **Step 9:** Return to step 6. |
| **Conclusions** | The property type has been updated on the Property Types data file. | |
| **Post conditions** |  | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

## Manage Tenant’s.

### Add Tenant.

This Function will add a Tenants’ details to tenants data file.

Estate Agent

<<includes>>

<<includes>>

<<includes>>

Tenant

<<extends>>

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Add Tenant | |
| **Use Case Id** |  | |
| **Priority** | 1 | |
| **Source** | Estate Agent | |
| **Primary Business Actor** | Estate Agent | |
| **Other Participating Actors** | Tenant | |
| **Description** | This function will add Tenant’s details to the system. | |
| **Preconditions** |  | |
| **Trigger** |  | |
| **Expected Scenario** | **Estate Agent** | **System** |
|  | **Step 1:** Invoke add Tenant  **Step 3:** Estate Agent enters Tenant’s data:   * firstName. * lastName. * phoneNumber. * email. * IBAN | **Step 2:** Display add Tenant User interface.  **Step 4:** Validate Tenant’s data:   * All fields must be entered * phoneNumber must be numeric & valid format. * Email must be valid format. * IBAN must be valid format.   **Step 5:** Save data to Tenant details file.  **Step 6:** Display confirmation message.  **Step 7:** Reset UI. |
| **Alternate Scenarios** | **Actor** | **System** |
| **Invalid Data Entered:** |  | **Step 4:** Invalid data entered.  **Step 5:** Display appropriate error message.  **Step 6:** Return to step 3. |
| **Conclusions** | Tenant’s details have been added to the system. | |
| **Post conditions** | Tenant can now be added to rental contracts. | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

### Update Tenant

This function will update the Tenants’ profile in the tenant’s data file.

Estate Agent

<<includes>>

<<includes>>

<<includes>>

Tenant

<<extends>>

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Update Tenant | |
| **Use Case Id** |  | |
| **Priority** | 2 | |
| **Source** | Estate Agent | |
| **Primary Business Actor** | Estate Agent | |
| **Other Participating Actors** | Tenant | |
| **Description** | This function will update Tenant’s details on the system. | |
| **Preconditions** |  | |
| **Trigger** |  | |
| **Expected Scenario** | **Estate Agent** | **System** |
|  | **Step 1:** Invoke Update Tenant  **Step 3:** Agent enters Tenant surname.  **Step 6:** Agent Selects appropriate Tenant.  **Step 8:** Agent edits Tenant’s data:   * firstName. * lastName. * phoneNumber. * email * IBAN | **Step 2:** Display Update Tenant User interface.  **Step 4:** Retrieve all Tenant data with matching surname.   * firstName * lastName * phoneNumber * TenantID   **Step 5:** Load UI with data.  **Step 7:** Retrieve the full Tenant details and load into UI.   * firstName. * lastName. * phoneNumber. * email * eircode. * IBAN * Status   **Step 9:** Validate data:   * All fields must be entered * phoneNumber must be numeric and be valid format. * email must be valid format. * eircode must be valid format * IBAN must be valid. * Status can either be Active (‘A’) or Inactive(‘I’).   **Step 10:** Update data in Tenants file.  **Step 11:** Display confirmation message.  **Step 12:** Reset UI. |
| **Alternate Scenarios** | **Actor** | **System** |
| **No Data Found:** |  | **Step 4:** No matching data found.  **Step 5:** Display appropriate error message.  **Step 6:** Return to step 3. |
| **Invalid Data Entered:** |  | **Step 9:** Invalid data entered.  **Step 10:** Display appropriate error message.  **Step 11:** Return to step 8. |
| **Conclusions** | Tenant’s details have been updated in the Tenants data file. | |
| **Post conditions** | Tenant’s details are now updated. | |
| **Business Rules** | Tenants must provide an IBAN must be valid format as it is required for payments / rental contracts. | |
| **Implementation Constraints** |  | |

## Manage Rentals.

### Rent Property.

This function will set up a rental contract and add it to the system.

Estate Agent

Tenant’s

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Rent Property | |
| **Use Case Id** |  | |
| **Priority** | 1 | |
| **Source** | Estate Agent | |
| **Primary Business Actor** | Estate Agent | |
| **Other Participating Actors** | Tenant’s (one or multiple). | |
| **Description** | This function will rent an available property to tenants. | |
| **Preconditions** | Properties must be added & tenant profile created prior to rental. | |
| **Trigger** |  | |
| **Expected Scenario** | **Estate Agent** | **System** |
|  | **Step 1:** Estate Agent will invoke the Rent Property function.  **Step 3:** Agent enters in Eircode into search.  **Step 7:** Estate agent enters in the data for the rental contract:   * Rental Start Date (Date) * Rental Duration (months) * Deposit & First Month Rent Paid (Yes/No) * Direct Debit Set Up. (Yes/No) * Select Tenant’s | **Step 2:** Display Rent Property UI.  **Step 4:** property with matching eircode is retrieved from Properties data file.  **Step 5:** Set Property found as the Property to rent.  **Step 6:** Display Rental UI.  **Step 8:** Validate Rental Data:   * Date must be valid future date. * Rental Duration must be numeric whole number. * At least one tenant must be added to the rental.   **Step 9:** Set rental status to ‘A’ for active. If deposit is paid & Direct Debit set up, set status to ‘V’ for valid.  **Step 10:** Generate a Rental ID for rental contract.  **Step 11:** Create Tenant Rental for each Tenant.  **Step 12:** Set Property status to Rented.  **Step 13:** Save rental to rentals data store.  **Step 14:** Display confirmation message.  **Step 15:** Reset UI. |
| **Alternate Scenarios** | **Estate Agent** | **System** |
| **No Property Found:** |  | **Step 4:** No data for searched eircode found.  **Step 5:** Display appropriate error message.  **Step 6:** return to step 3. |
| **Invalid Data Entered:** |  | **Step 8:** Invalid data entered.  **Step 9:** Display appropriate error message.  **Step 10:** Return to step 8. |
| **Conclusions** | Property selected has now been rented. | |
| **Post conditions** |  | |
| **Business Rules** | Maximum 5 tenants on a rental contract.  Deposit & First month must be paid and Direct Debit must be set up before a contract is valid.  Cash payments only valid for First month & deposit. | |
| **Implementation Constraints** |  | |

### Update Rental

This function will update a rental contract and save it in the rentals data file.

Estate Agent

Tenant’s

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Update Rental | |
| **Use Case Id** |  | |
| **Priority** | 1 | |
| **Source** | Estate Agent | |
| **Primary Business Actor** | Estate Agent | |
| **Other Participating Actors** | Tenant’s (one or multiple). | |
| **Description** | This function will rent an available property to tenants. | |
| **Preconditions** | Properties must be added & tenant profile created prior to rental. | |
| **Trigger** |  | |
| **Expected Scenario** | **Estate Agent** | **System** |
|  | **Step 1:** Estate Agent will invoke the Rent Property function.  **Step 3:** Agent enters in Eircode into search.  **Step 7:** Estate agent Updates the data for the rental contract:   * Rental Start Date (Date) * Rental Duration (months) * Deposit & First Month Rent Paid (Yes/No) * Direct Debit Set Up. (Yes/No) * Select Tenant’s | **Step 2:** Display Update Rental Property UI.  **Step 4:** Active Rental with matching Eircode is Retrieved from the Rental data file.  **Step 5:** Load Rental’s data into Rental UI.  **Step 6:** Display Rental UI.  **Step 8:** Validate Rental Data:   * Date must be valid future date. * Rental Duration must be numeric whole number. * At least one tenant must be added to the rental.   **Step 9:** Update rental in rentals data store.  **Step 10:** Display confirmation message.  **Step 11:** Reset UI. |
| **Alternate Scenarios** | **Estate Agent** | **System** |
| **No Property Found:** |  | **Step 4:** No data for searched eircode found.  **Step 5:** Display appropriate error message.  **Step 6:** return to step 3. |
| **Invalid Data Entered:** |  | **Step 8:** Invalid data entered.  **Step 9:** Display appropriate error message.  **Step 10:** Return to step 8. |
| **Conclusions** | Rental Contract has been updated. | |
| **Post conditions** |  | |
| **Business Rules** | Maximum 5 tenants on a rental contract.  Deposit & First month must be paid and Direct Debit must be set up before a contract is valid.  Cash payments only valid for First month & deposit. | |
| **Implementation Constraints** |  | |

### Record Payment

This function will Record a Payment into the Payments File.

Estate Agent

<<includes>>

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Record Payment | |
| **Use Case Id** |  | |
| **Priority** | 1 | |
| **Source** | Estate Agent | |
| **Primary Business Actor** | Estate Agent | |
| **Other Participating Actors** | Tenant’s (one or multiple). | |
| **Description** | This function will Record payments into the payments file. | |
| **Preconditions** |  | |
| **Trigger** |  | |
| **Expected Scenario** | **Estate Agent** | **System** |
|  | **Step 1:** Estate Agent will invoke the Record Payment function.  **Step 3:** Agent enters in phone number of tenant.  **Step 5:** Agent Enters in the Payment Details:   * Amount * Date | **Step 2:** Display Record Payment UI.  **Step 4:** Tenant Found is placed as the Payer & UI Loaded with this information.  **Step 6:** Validate payment details:   * Amount must be positive numeric. * Date must be before or on current date.   **Step 7:**  Assign Payment ID.  **Step 8:**  Record Payment in Payments data store.  **Step 9:** Display confirmation message.  **Step 10:** Reset UI. |
| **Alternate Scenarios** | **Estate Agent** | **System** |
| **No Tenant Found:** |  | **Step 4:** No Tenant Data found with that phone number or invalid Phone Number.  **Step 5:** Display appropriate error message.  **Step 6:** return to step 3. |
| **Invalid Data Entered:** |  | **Step 6:** Invalid data entered.  **Step 7:** Display appropriate error message.  **Step 8:** Return to step 5. |
| **Conclusions** | Payment has been recorded in the payments file. | |
| **Post conditions** |  | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

## Produce Administrative Reports.

### Produce Yearly Commission Report.

This function will generate a graphical report on the commission in a year.

Estate Agent

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Produce Yearly Commission Report | |
| **Use Case Id** |  | |
| **Priority** | 3 | |
| **Source** | Estate Agent | |
| **Primary Business Actor** | Estate Agent | |
| **Other Participating Actors** |  | |
| **Description** | This function will gather data on the commission earned from rentals in the year. | |
| **Preconditions** |  | |
| **Trigger** |  | |
| **Expected Scenario** | **Actor Name** | **System** |
|  | **Step 1:** Estate Agent invokes Produce Yearly Report.  **Step 3:** Estate Agent selects year of report. | **Step 2:** Display Produce Yearly Commission Report UI.  **Step 4:** Retrieve Data from Payments File for every entry of that year.  **Step 5:** Display Data in graph. |
| **Alternate Scenarios** | **Actor** | **System** |
|  |  |  |
| **Conclusions** | Commission Report Generated & Displayed. | |
| **Post conditions** |  | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

Example Commission Report Graph’s:

Year 2018:

Chart, line chart

Description automatically generated

Year 2019:

Chart, line chart

Description automatically generated

### Produce Rentals in Year Report

This function will gather data on the number of rentals in a month within the year, and the number of rentals of each type of rentals that month.

Estate Agent

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Produce Rentals in Year Report | |
| **Use Case Id** |  | |
| **Priority** | 3 | |
| **Source** | Estate Agent | |
| **Primary Business Actor** | Estate Agent | |
| **Other Participating Actors** |  | |
| **Description** | This function will gather data on the number of rentals in a month, and the number of rentals of each type of rentals that month. | |
| **Preconditions** |  | |
| **Trigger** |  | |
| **Expected Scenario** | **Actor Name** | **System** |
|  | **Step 1:** Estate Agent invokes Rentals in Year Report.  **Step 3:** Estate Agent selects year of report. | **Step 2:** Display Produce Rentals in Year Report UI.  **Step 4:** Retrieve Data from Rentals data file within the year selected & the Property Details associated with that Rental from the property file.  **Step 5:** Display Data in graph. |
| **Alternate Scenarios** | **Actor** | **System** |
|  |  |  |
| **Conclusions** | Rentals in Year Report Generated & Displayed. | |
| **Post conditions** |  | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

Example Rentals in Year Graph’s:

Year 2018:

A screenshot of a computer

Description automatically generated with medium confidence

Year 2019:

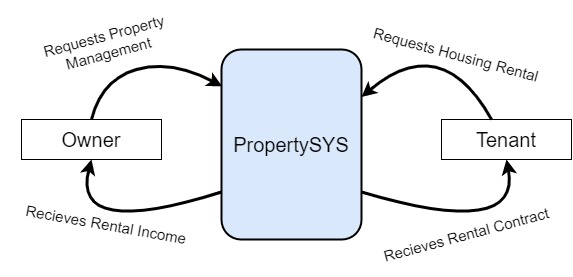
A screenshot of a computer

Description automatically generated with medium confidence

# System Model

The following dataflow diagrams have been produced for the system:

## Level-0 DFD



## Level-1 DFD

Diagram

Description automatically generated

## Level-2 DFD (Process P1: Manage Owners)

Diagram

Description automatically generated

## Level-2 DFD (Process P2: Manage Properties)

Diagram

Description automatically generated

## Level-2 DFD (Process P3: Manage Tenants)

Diagram

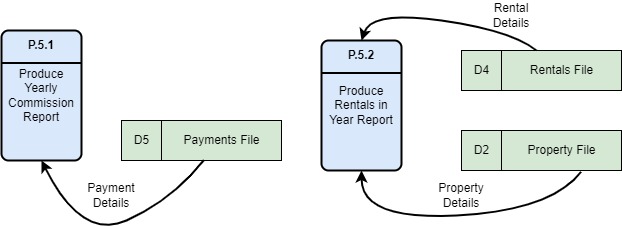
Description automatically generated

## Level-2 DFD (Process P4: Rentals)

Diagram

Description automatically generated

## Level-2 DFD (Process P5: Administration Reports))



# Data Model

In this section there will be the data modelling for the Rental Property System, it includes the UML Class Diagrams, Relational Schema & Database Scheme.

These will be available for the data engineer to create the required data structure for the project. Including the types of data and the tables required by the system we have designed.

This section represents the data in a graphical & textual format from which a database file system may be designed.

## Class Diagram

Property Rental System UML Class Diagram:

Diagram

Description automatically generated

## Relational Schema

Relational schema for the data requirements - Using ***bracket notation***

**Owners (**Owner\_ID, firstName, lastName, phone, email, eircode, IBAN)

**PropertyTypes (**typeCode, description)

**Properties (**Eircode, houseName, description, monthlyRent, totalRooms, bedrooms, en-suiteBedrooms, bathrooms, parkingSpaces, heatingSource, gardenSpace, petsAllowed, ownerOccupied, Wi-Fi, status, Owner\_ID, typeCode)

**Rentals (**rental\_ID,startDate, endDate, depositPaid, directDebit, Eircode)

**Tenants (**tenant\_ID, firstName, lastName, phone, email, IBAN)

**TenantRentals (**tenantRental\_ID, rental\_ID, tenant\_ID)

## Database Schema: PropertySYS

**Relation Owners**

Owner\_ID Numeric (5)

firstName String (25) NOT NULL

lastName String (30) NOT NULL

phone Numeric (10) NOT NULL

email String (50) NOT NULL

eircode String (7) NOT NULL

IBAN String (34) NOT NULL CHECK > 22

**Primary Key:** Owner\_ID

**Relation PropertyTypes**

typeCode String (2)

Description String (100) NOT NULL

**Primary Key:** typeCode

**Relation Properties**

Eircode String (7)

houseName String (30) NOT NULL

description String (200) NOT NULL

monthlyRent Numeric (10,2) NOT NULL

totalRooms Numeric (3) NOT NULL CHECK > 0

bedrooms Numeric (2)

en-suiteBedrooms Numeric (2)

bathrooms Numeric (2)

parkingSpaces Numeric (2)

heatingSource String (25) NOT NULL

gardenSpace Boolean DEFAULT false

petsAllowed Boolean DEFAULT false

ownerOccupied Boolean DEFAULT false

status Char (1) DEFAULT ‘A’ CHECK ‘A’ OR ‘U’ OR ‘R’

Owner\_ID numeric (5) NOT NULL

typeCode String (2) NOT NULL

**Primary Key:** Eircode

**Foreign Key:** Owner\_ID References Owners,

typeCode References PropertyTypes

**Relation Rentals**

Rental\_ID Numeric (7)

startDate Date NOT NULL

endDate Date NOT NULL

depositPaid Boolean DEFAULT false

directDebitSetUp Boolean DEFAULT false

Eircode String (7) NOT NULL

**Primary Key:** Rental\_ID

**Foreign Key:** Eircode References Properties

**Relation Tenants**

Tenant\_ID Numeric (5)

firstName String (25) NOT NULL

lastName String (30) NOT NULL

phone Numeric (10) NOT NULL

email String (50) NOT NULL

IBAN String (34) NOT NULL CHECK > 22

**Primary Key:** Tenant\_ID

**Relation TenantRentals**

TenantRental\_ID Numeric (10)

Tenant\_ID Numeric (5) NOT NULL

Rental\_ID Numeric (7) NOT NULL

**Primary Key:** TenantRental\_ID

**Foreign Key:** Tenant\_ID References Tenants

Rental\_ID References Rentals

# Conclusion

Throughout the creation of this system document, we have learned about several processes and implemented them within the process of each section.

Our user Requirements outlined the functional requirements of the property rental system giving a high level overview of what to expect from the system.

By completing the Use case Narrative’s, we have seen how the functionality of the system in question can be implemented including both the expected path and alternate scenarios.

The use of the waterfall approach allowed the systematic creation of each layer of the system, working from the left to right of the Functional Requirements hierarchy the prototype and use cases were created for each component of the system before approaching the next high level process.

With the System Model & DFD diagrams we can see how the data can flow from each process, where it will be stored and how it can be used by other processes to complete the overall functionality of the system.

The Data Model allows a data engineer to create a database system that would store the required data from the property rental system in an efficient and logical format.

The prototype alongside this document allows the client and end users to get a real-life experience of what to expect from the completed system and would be highly beneficial to show the cost benefit of the Property Rental System.

While there is no direct “Remove” button in the system prototype, within the update Status options we essentially can archive objects in the system without effecting referential Integrity. One potential issue with this is the fact of GDPR a tenant or owner may ask for their data to be removed. As a solution I would suggest implementation of a function to anonymise the data in the system files relating to people’s data this would sustain referential integrity in the system.

Throughout the creation of this system, it is clear the advantages of proper documentation and planning of the system can greatly reduce the creation time of the system. Early on it was discovered that the orientation of the functional requirements reflects the dependencies of following high level processes that follow each other. This also helps with delegation of priority of functions. If the system was being carried out by a larger organisation, I would suggest the addition of Network Analysis to give a greater estimation of the possible time frame to complete and deliver a fully functional system.

# Appendices

## Appendix A – Declaration Of Originality.

Graphical user interface, text, application, table

Description automatically generated with medium confidence