Requirements Engineering

Holiday Home Rental System

Submitted By: Petrit Krasniqi (T-00192719)

Computing with Software Development

Date Submitted: 12/12/2017

**Table of Contents**

[1. Introduction/overview 3](#_Toc474162299)

[2. Functional Components 4](#_Toc474162300)

[3. User Requirements 5](#_Toc474162301)

[4. System Requirements 6](#_Toc474162302)

[4.1. System Level Use Case Diagram 6](#_Toc474162303)

[4.2. Module 1 (Use active Verb + Noun) 6](#_Toc474162304)

[4.2.1. Functional Requirement 1 (Use active verb + noun) 6](#_Toc474162305)

[4.2.2. Functional Requirement 2 (Use active verb + noun) 6](#_Toc474162306)

[4.2.3. Functional Requirement 3 (Use active verb + noun) 6](#_Toc474162307)

[4.3. Module 2 (Use active Verb + Noun) 6](#_Toc474162308)

[4.3.1. Functional Requirement 1 (Use active verb + noun) 6](#_Toc474162309)

[4.3.2. Functional Requirement 2 (Use active verb + noun) 6](#_Toc474162310)

[4.4. Module 3 (Use active Verb + Noun) 6](#_Toc474162311)

[4.4.1. Functional Requirement 1 (Use active verb + noun) 6](#_Toc474162312)

[4.4.2. Functional Requirement 2 (Use active verb + noun) 6](#_Toc474162313)

[5. System Model 7](#_Toc474162314)

[5.1. Level-0 DFD 7](#_Toc474162315)

[5.2. Level-1 DFD 7](#_Toc474162316)

[5.3. Level-2 DFD (Process P1: Title) 7](#_Toc474162317)

[5.4. Level-2 DFD (Process P2: Title) 7](#_Toc474162318)

[5.5. Level-2 DFD (Process P3: Title) 7](#_Toc474162319)

[6. Data Model (Class Diagram) 8](#_Toc474162320)

[6.1. Class Diagram 8](#_Toc474162321)

[6.2. Relational Schema 8](#_Toc474162322)

[6.3. Database Schema 8](#_Toc474162323)

[7. Conclusion 9](#_Toc474162324)

[8. Appendices 10](#_Toc474162325)

[8.1. Appendix A – Title 10](#_Toc474162326)

[8.2. Appendix B – Title 10](#_Toc474162327)

# Introduction/overview

**HolidayHome Rental System** is a system that involves the execution of functions such as property administration where all the properties are owned by the admin, booking administration, invoice administration and also generate revenue reports.

Primary functions of the system are adding, removing and updating properties, make bookings, cancel bookings, check – in, check – out, issue invoice and generate revenue reports for either all properties or for a specific property to aid decision making.

# Functional Components

# User Requirements

The functional user requirements for HolidayHomeSYS are described below:

## 3.1 HolidayHomeSYS will perform property administration

3.1.1 HolidayHomesSYS will register all the properties that are available for rent.

3.1.2 HolidayHomeSYS will record all the properties that have been recently booked and

Update their status.

3.1.3 HolidayHomeSYS will remove properties that are no longer available for rent due to maintenance.

3.1.4 HolidayHomeSYS will check for availability of properties, i.e. for properties that have been booked by user’s months in advance.

## 3.2 HolidayHomeSYS will perform booking administration

3.2.1 HolidayHomesSYS will allow the user to book a home.

3.2.2 HolidayHomeSYS will allow the user to cancel their booking.

3.2.3 HolidayHomeSYS will allow the user to check – in to the holiday home.

3.2.4 HolidayHomeSYS will allow the user to check – out of the holiday home.

## 3.3 HolidayHomeSYS will perform an invoice administration

3.3.1 HolidayHomesSYS will issue an invoice for tenants.

3.2.2 HolidayHomeSYS will perform a revenue analysis for properties.

3.2.3 HolidayHomeSYS will perform a revenue analysis for each property.

# 4. System Requirements

|  |  |  |
| --- | --- | --- |
| **Functional Requirements** | **Non-Functional Requirements** | **Domain Requirements** |
| 1. To allow for a property to be added. 2. To allow properties to be updated or deleted. 3. To allow for people to make bookings or cancel bookings. 4. To system will allow for an invoice to be issued for the guest. 5. The system will allow the admin to generate reports for all properties or for just one property | 1. The system must be quick and easy to use. Each entry will happen on one form. 2. The admin is the only person that can access the system he has his own unique user name and password that will allow him to enter. 3. The system must also be fully function and look nice. | 1. The admin will have access to the system in his office and the system will not be able to be accessed online. |

## System Level Use Case Diagram

HolidayHome will allow the admin to carry out functions for the management of properties. These functions include property registration, amendment, de-registration. A function to check the available properties is also included.

HolidayHome will allow the admin to carry out functions for the management of bookings. These functions include make booking, cancel booking, check-in and check-out.

HolidayHome will allow the admin to carry out functions that allow the system to issue an invoice, perform a revenue analysis and perform a revenue analysis for each property.

Admin

HolidayHome

## Manage Properties

This component includes functions for the management of properties. These functions include property registration, amendment, de-registration. A function to check the available properties is also included.

### **Register Property**

This function will register all the properties that are available for rent and will display them to the user. It will show the price, number of beds, facilities, the type of the property and its status.

Admin

<<Includes>>

<<Extends>>

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Register Property** | |
| **Use Case Id** | 1.1 | |
| **Priority** | High | |
| **Source** | Admin | |
| **Primary Business Actor** | Admin | |
| **Other Participating Actors** |  | |
| **Description** | This function registers a property on the system. | |
| **Preconditions** | The admin must complete a list of all the properties. | |
| **Trigger** | None | |
| **Expected Scenario** | **Actor Action** | **System Response** |
|  | **Step 1:** Admin starts the property register function.  **Step 4:** Admin enters property details:   * Type * Street * Town * County * No Beds * Garage * Garden * Daily Rate | **Step 2:** System determines the next Property Id  **Step 3:** System loads No Beds, Type and County into form control (List)    **Step 5:** System validates property details:   * All fields must be entered * No Beds must be entered * Type must be specified * Garage or Garden status must be specified i.e. yes or no radio buttons must be checked   **Step 6:** System sets property status = ‘Available’  **Step 7:** Save property details in Properties File:   * Prop No * Type * Street * Town * County * No Beds * Garage * Garden * Rent * Status   **Step 8:** Display confirmation message.  **Step 9:** Clear UI. |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
| **Required field not entered** | **Step 4:** The admin leaves required fields blank.  **Step 4:** The admin enters letters into a textbox that only accepts numbers.  **Step 4:** The admin enters numbers into a textbox that only accepts letters. | **Step 5:** The system displays appropriate error message and directs the admin to the field he/she left blank.  **Step 5:** The system displays error message and directs the admin to the textbox.  **Step 5:** The system displays error message and directs the admin to the textbox. |
| **Conclusions** | Properties are registered. | |
| **Post conditions** | This property may now be rented. | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

### **Update Property**

This function updates the details of a property on the system.

Admin

<<Includes>>

<<Extends>>

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Update Property** | |
| **Use Case Id** | 1.2 | |
| **Priority** | High | |
| **Source** | Admin | |
| **Primary Business Actor** | Admin | |
| **Other Participating Actors** | Guest | |
| **Description** | Property details are retrieved from the properties file. Status is then updated and saved to the properties file. | |
| **Preconditions** | The user must have booked a property | |
| **Trigger** | Make Booking Function | |
| **Expected Scenario** | **Actor Action** | **System Response** |
|  | **Step 1:** The admin invokes the Update Property function.  **Step 4:** The admin selects the property to update and updates the detail/details he/she wants to change:   * No\_beds * Garage * Garden * Rent | **Step 2:** The system retrieves the details of all properties that are not booked for in order of property number and displays them on a data grid view:   * Prop No * Type * Street * Town * County * No Beds * Garage * Garden * Rent * Status   **Step 3:** Load the prop\_no for the properties that are not booked for on form control (list)  **Step 5**: System validates the input data:   * All fields must be entered. * Garage check boxes must be ticked * Garden check boxes must be ticked * Rent rate must be entered   **Step 6:** System saves update details in properties file.  **Step 7:** Display confirmation message. |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
|  | **Step 2:** There are no properties to update.  **Step 4**: The admin leaves a field blank.  **Step 4:** The admin enters letters into a textbox that only accepts numbers.  **Step 4:** The admin enters numbers into a textbox that only accepts letters. | **Step 3:** Display appropriate error message**.**  **Step 5**: Display appropriate error message.  **Step 5:** The system displays error message and directs the admin to the textbox.  **Step 5:** The system displays error message and directs the admin to the textbox. |
| **Conclusions** | Properties have been updated | |
| **Post conditions** | Status cannot be changed | |
| **Business Rules** | Property cannot be updated if de-registered | |
| **Implementation Constraints** |  | |

### **Remove Property**

This function removes properties that have been made recently unavailable for reasons to do with the admin.

Admin

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Remove Property** | |
| **Use Case Id** | 1.3 | |
| **Priority** | High | |
| **Source** | Admin | |
| **Primary Business Actor** | Admin | |
| **Other Participating Actors** |  | |
| **Description** | Property details are retrieved from the properties file. The selected property is then removed. | |
| **Preconditions** | The admin must request for a property to be removed | |
| **Trigger** | None | |
| **Expected Scenario** | **Actor Action** | **System Response** |
|  | **Step 1:** The admin wishes for a property to be removed.  **Step 2:** The admin invokes the Remove Property function.  **Step 5:** The admin selects the property number and removes it from the properties list. | **Step 3:** The system retrieves the details of all properties that are not booked for and that have a status of A by property number and displays them on a data grid view:   * Prop No * Type * Street * Town * County * No Beds * Garage * Garden * Rent * Status   **Step 4:** The system displays the property number of the properties that are not booked for and loads them on a form control (list)  **Step 5:** System changes to property status to ‘D’.  **Step 6:** Display confirmation message. |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
|  | **Step 4:** All properties are booked for and cannot be deregistered.  **Step 4**: The admin doesn’t select a property to remove. | **Step 5:** Display appropriate error message.  **Step 5**: Display appropriate error message. |
| **Conclusions** | Property is removed | |
| **Post conditions** | Property cannot be rented | |
| **Business Rules** | Status must not be “D” | |
| **Implementation Constraints** |  | |

### **4.2.4 Check Availability**

This function will check the status of the properties i.e. is it available or booked.

Admin

|  |  |  |
| --- | --- | --- |
| **Expected Scenario** | **Actor Action** | **System Response** |
|  | **Step 1:** A guest books a property.  **Step 2:** The admin invokes the check availability function.  **Step 4:** The admin selects the property that he/she wants to check the status of. | **Step 3:** The system retrieves the details of all properties by property number that are not booked for and displays them on a data grid view:   * Prop No * Type * Street * Town * County * No Beds * Garage * Garden * Rent * Status   **Step 3:** The system displays the property number of those properties on a form control (list)  **Step 5**: System validates the input data:   * Status field must not be empty.   **Step 6:** System saves update details in properties file.  **Step 7:** Display confirmation message. |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
|  | **Step 3:** There were no properties deregistered.  **Step 4**: The admin leaves status field blank. | **Step 4:** Display error message**.**  **Step 5**: Display error message. |
| **Conclusions** | Properties have been updated | |
| **Post conditions** | Status cannot be changed | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

## Manage Bookings

This component includes functions for the management of bookings. These functions include make booking, cancel booking, check-in and check-out.

### **Make Booking**

This function allows users to book a property.

Guest

Admin

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Make Booking** | |
| **Use Case Id** | 2.1 | |
| **Priority** | High | |
| **Source** | Admin | |
| **Primary Business Actor** | Amin | |
| **Other Participating Actors** | Guest | |
| **Description** | The admin receives bookings from the users and enters the information into the system. | |
| **Preconditions** | The guest must call up to make a booking. | |
| **Trigger** |  | |
| **Expected Scenario** | **Actor Action** | **System Response** |
|  | **Step 1:** The admin invokes the Make Booking function.  **Step 3:** The admin selects the arrival date and departure date and the property type and county according to the customer’s request.  **Step 6:** The admin selects the property from the data grid view  **Step 8:** A grid box then appears which requires the admin to enter the users details:   * Surname * Forename * PhoneNumber * CardNumber | **Step 2:** The System loads property type and county into form control (list)  **Step 4:** The System validates the input data:   * Arrival and departure date must be entered. * Property Type must be entered. * County must be specified * Departure date must not be the same as the arrival date.   **Step 5:** The System loads the details of the properties of that type and in that county in order of prop\_no that are available for those dates:   * Prop No * Type * Street * Town * County * No Beds * Garage * Garden * Rent * Status   **Step 9:** The system validates the input data:   * Surname must be entered * Forename must be entered * Card Number must entered and must not contain letters * Phone Number must be entered and must not contain letters.   **Step 10:** The system generates the next bookingID and TenantID and saves the following details in the bookings file:   * BookingID * TenantID * Prop\_No * DateFrom * DateTo * Amount\_Paid   **Step 11:** The system also saves the following details in the tenants file:   * TenantID * Surname * Forename * Phone\_Number * Card\_Number   **Step 10:** Display confirmation message. |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
| **No properties available** |  | **Step 8:** No Properties available  **Step 9:** Display error message |
| **Conclusions** | New booking is created | |
| **Post conditions** |  | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

### **Cancel Booking**

This function allows users to cancel bookings

Guest

Admin

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Cancel booking** | |
| **Use Case Id** | 2.1 | |
| **Priority** | High | |
| **Source** | Admin | |
| **Primary Business Actor** | Admin | |
| **Other Participating Actors** | Guest | |
| **Description** | The admin retrieves booking details from bookings file and cancels booking | |
| **Preconditions** | A guest calls to cancel their booking | |
| **Trigger** | Make booking | |
| **Expected Scenario** | **Actor Action** | **System Response** |
|  | **Step 1:** The admin invokes the cancel booking function.  **Step 4:** The admin selects the bookingID of the booking he/she wishes to cancel. | **Step 2:** The system loads all the booking details from the bookings file that have a booking status of ‘B’ on a data grid view:   * BookingID * TenantID * Prop\_No * Booking\_Status * DateFrom * DateTo * Amount\_Paid   **Step 3:** The system loads those bookings id’s on a form control (list)  **Step 5:** The system deletes that booking from the bookings file.  **Step 6:** Display Confirmation message |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
|  |  |  |
| **Conclusions** | Booking is cancelled | |
| **Post conditions** |  | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

### **Check-In**

This function allows guests to check – in.

Guest

Admin

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Check-in** | |
| **Use Case Id** | 2.3 | |
| **Priority** | High | |
| **Source** | Admin | |
| **Primary Business Actor** | Admin | |
| **Other Participating Actors** | Guest | |
| **Description** | Guest arrives to check-in. The admin confirms check-in. | |
| **Preconditions** |  | |
| **Trigger** | Make booking | |
| **Expected Scenario** | **Actor Action** | **System Response** |
|  | **Step 1:** The guest arrives and wishes to check-in.  **Step 2:** The admin invokes the check-in function.  **Step 4:** The admin selects the guest’s booking\_id.  **Step 7:** The guest makes the payment and the admin changes the guest’s booking status to ‘C-IN’ to confirm the booking. | **Step 3:** The system loads the booking\_ids where the arrival date matches todays date on a form control (list).  **Step 5:** The system retrieves the guests booking details from the bookings file and displays them on a data grid view:   * BookingID * TenantID * Prop\_No * Booking\_Status * DateFrom * DateTo * Amount\_Paid   **Step 6:** The admin loads the booking status on a form control (list)  **Step 8:** The system generates a new payment id for the guest.  **Step 9:** The payment details are then saved in the payments file:  Payment\_id  Tenant\_id  Date\_paid  Amount\_paid  **Step 10:** The updated booking details are now saved in the bookings file.  **Step 11:** Display confirmation message. |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
| **If the guest arrives with no reservation** | **Step 1:** The admin invokes the Make Booking function.  **Step 5:** The admin selects the property from the data grid view  **Step 6:** A grid box then appears which requires the admin to enter the users details:   * Surname * Forename * PhoneNumber * CardNumber   **Step 9:** The guest makes the payment and the admin changes the guest’s booking status to ‘C-IN’ to confirm the booking. | **Step 2:** The System loads property type,and county into form control (list)  **Step 3:** The System validates the input data:   * Arrival and departure date must be entered. * Property Type must be entered. * No. Beds must be entered. * Street must be entered * Town must be entered * County must be specified * Yes or no option must be checked for garage * Yes or no option must be checked for garden   **Step 4:** The System loads the details of the properties of that type and in that county in order of prop\_no that are available for those dates:   * Prop No * Type * Street * Town * County * No Beds * Garage * Garden * Rent * Status   **Step 7:** The system validates the input data:   * Surname must be entered * Forename must be entered * Card Number must entered and must not contain letters * Phone Number must be entered and must not contain letters.   **Step 8:** The system generates the next bookingID and TenantID and saves the following details in the bookings file:   * BookingID * TenantID * Prop\_No * DateFrom * DateTo * Amount\_Paid   **Step 10:** The system also saves the following details in the tenants file:   * TenantID * Surname * Forename * Phone\_Number * Card\_Number   **Step 11:** The system generates a new payment id for the guest.  **Step 12:** The payment details are then saved in the payments file:  Payment\_id  Tenant\_id  Date\_paid  Amount\_paid  **Step 10:** Display confirmation message. |
| **No Properties Available** |  | **Step 8:** No Properties available  **Step 9:** Display error message |
| **Conclusions** | The guest checks-in. | |
| **Post conditions** |  | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

### **Check – out**

This function allows users to check-out.

Guest

Admin

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Check-out** | |
| **Use Case Id** | 2.4 | |
| **Priority** | High | |
| **Source** | Admin | |
| **Primary Business Actor** | Admin | |
| **Other Participating Actors** | Guest | |
| **Description** | The guest checks-out and the bill is prepared. | |
| **Preconditions** | The guest must have checked in. | |
| **Trigger** | Check-in | |
| **Expected Scenario** | **Actor Action** | **System Response** |
|  | **Step 1:** The guest wishes to check – out.  **Step 2:** The admin invokes the check – out function.  Step 4: The admin selects the booking id for the guest who wishes to check-out. | **Step 3:** The system loads the  Booking-ids where the departure date matches todays date.  **Step 4:** The guest booking\_status is retrieved from the bookings file and displayed on screen.  **Step 5:** The guest’s booking\_status is changed to C-out  **Step 8:** Updated booking details are saved in the bookings file. |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
|  |  |  |
| **Conclusions** | The guest checks out and the bill is paid. | |
| **Post conditions** |  | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

## Perform Admin

This component includes functions that allow the system to issue an invoice, perform a revenue analysis and perform a revenue analysis for each property.

### **Issue Invoice**

This function issues an invoice

<<Includes>>

Guest

Admin

<<Includes>>

<<Includes>>

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Issue Invoice** | |
| **Use Case Id** | 3.1 | |
| **Priority** | High | |
| **Source** | Admin | |
| **Primary Business Actor** | Admin | |
| **Other Participating Actors** | Guest | |
| **Description** | This function will issue and invoice to the guest. | |
| **Preconditions** | The guest must have checked into a property and when they check out they receive an invoice. | |
| **Trigger** | Check-out | |
| **Expected Scenario** | **Actor Action** | **System Response** |
|  | **Step 1:** The guest checks out.  **Step 2:** The admin invokes the issue invoice function.  **Step 3:** The admin enters the guest’s arrival date  **Step 5:** The admin selects the guest’s booking id. | **Step 4:** The system then loads the booking id’s with the guest’s name on a form control list**.**  **Step 6:** The system retrieves all the guest details from the bookings and tenants file:   * Prop\_No * Surname * Forename * ArrivalDate * DepartureDate * Amount\_Paid   **Step 7:** The admin prints out the invoice for the user.  **Step 8:** Display confirmation message. |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
|  |  |  |
| **Conclusions** | The guest has paid and has been issued an invoice | |
| **Post conditions** |  | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| *Invoice:*   |  | | --- | | 18-04-2018 |  |  | | --- | | 1 |   Arrival Date:  Property Number:   |  | | --- | | Krasniqi |   Surname:   |  | | --- | | Petrit |   Forename:   |  | | --- | | 18-04-2018 |   ArrivalDate:   |  | | --- | | 20-04-2018 |   DepartureDate:   |  | | --- | | 500 |   Amount: |

### **Perform Revenue Analysis**

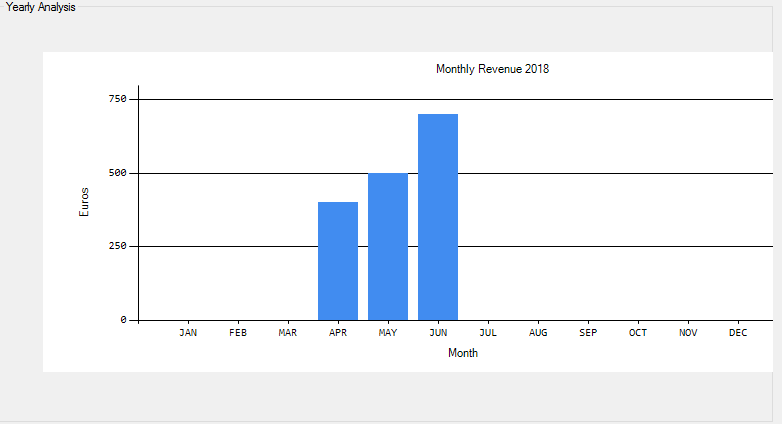
This function performs a revenue analysis report.

Admin

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Revenue Analysis** | |
| **Use Case Id** | 3.2 | |
| **Priority** | High | |
| **Source** | Admin | |
| **Primary Business Actor** | Admin | |
| **Other Participating Actors** |  | |
| **Description** | This function performs a revenue analysis report. | |
| **Preconditions** | Properties must have been booked by guests. | |
| **Trigger** | Make Booking. | |
| **Expected Scenario** | **Actor Action** | **System Response** |
|  | **Step 1:** The admin starts the revenue analysis function.  **Step 3:** The admin selects the year that he/she wishes to perform a revenue analysis report on. | **Step 2:** The system loads year into form control (list).  **Step 4:** The system generates a chart analysis for the specified year.  **Step 5:** The admin prints the report.  **Step 7:** Display confirmation message. |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
|  |  |  |
| **Conclusions** | The admin has a revenue analysis report. | |
| **Post conditions** |  | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

**Revenue Analysis by report: HolidayHomeSYS**

Sample chart for 2018:



### **4.4.3 Perform Revenue Analysis by property**

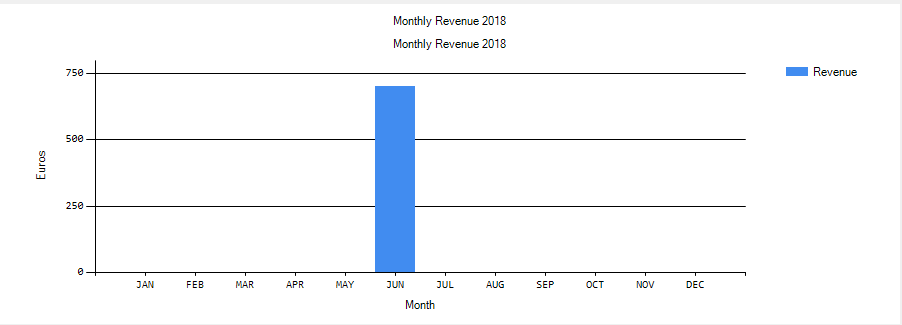
This function performs a revenue analysis report by property

Admin

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Revenue Analysis by property** | |
| **Use Case Id** | 3.3 | |
| **Priority** | High | |
| **Source** | Admin | |
| **Primary Business Actor** | Admin | |
| **Other Participating Actors** |  | |
| **Description** | This function performs a revenue analysis report by property | |
| **Preconditions** | Properties must have been booked by guests. | |
| **Trigger** | Make Booking | |
| **Expected Scenario** | **Actor Action** | **System Response** |
|  | **Step 1:** The admin starts the revenue analysis by property function.  **Step 3:** The admin enters the prop\_no of the property he wishes to analysis. | **Step 2:** The system loads year into form control (list).  **Step 4:** The system generates a revenue analysis chart by property.  **Step 5:** The admin prints the chart.  **Step 6:** Display confirmation message. |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
|  |  |  |
| **Conclusions** | The admin has a revenue analysis report | |
| **Post conditions** |  | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

**Revenue Analysis by property chart: HolidayHomeSYS**

Sample revenue analysis chart for prop\_No 13



# System Model

The following dataflow diagrams have been produced for the system:

## Level-0 DFD

Booking request

Guest

HolidayHomeSYS

Booking confirmation

## Level-1 DFD

P1

Manage

Properties

Property Details

Guest

Booking Details

P2

Manage Bookings

D1

Properties file

Property Details

Booking Details

D2

Bookings file

Booking Details

P3

Process Payments

Guest

Payment Details

D4

Payments file

## Level-2 DFD (Manage Properties)

P1.1

Register

Property

P1.4

Check

Availability

P1.3

Deregister

Property

P1.2

Update

Property

Deregister Property

Property Details

New Property Details

Current Property Details

Current Property Details

New Property Details

New Property Details

D4

Properties file

D2

Bookings file

## Level-2 DFD (Manage Bookings)

Cancel Request

Guest

Booking Confirmation

Booking Request

P2.2

Cancel

Booking

P2.1

Make

Booking

Check – in Details

P2.4

Check - out

Booking Request

Property Details

D2

Bookings file

Status

Booking Details

P2.3

Check - in

D1

Properties file

Property Status

Payment Details

D4

Payments file

## Level-2 DFD (Perform Admin)

Invoice No

Payment Details

Guest

Payment Details

D4

Payments file

P3.1

Issue

Invoice

# Data Model (Class Diagram)

## Class Diagram



## Relational Schema

Tenants (Tenant\_ID, Surname, Forename, Phone\_No, Card\_No)

Properties (Prop\_No, Street, Town, County, Garage, Garden, Status, Daily\_rate, Prop\_Type)

Types (Prop\_type, Description)

Bookings (Booking\_ID, Prop\_No, tenant\_ID, Date\_From, Date\_To, Booking\_Status, Amout\_Paid)

Payments (Payment\_ID, Tenant\_ID, Amount, Date\_Paid)

## Database Schema

**Schema:** **Holiday Home**

**Relation:** Tenants

Attributes:

Tenant\_ID numeric (10) NOT NULL

Forename char (20) NOT NULL

Surname char (20) NOT NULL

Phone\_No varchar (15) NOT NULL

Card\_No numeric (20) NOT NULL

**Primary Key:** Tenat\_ID

**Relation:** Properties

Attributes:

Prop\_Type char (1) NOT NULL

Description char (15) NOT NULL

Status char (1) NOT NULL

Street char (20) NOT NULL

Town char (20) NOT NULL

County char (20) NOT NULL

Eir\_Code varchar (11) NOT NULL

Garage char (1) NOT NULL

Garden char (1) NOT NULL

Weekly\_Cost numeric (10) NOT NULL

**Primary Key:** Prop\_No

**Foreign Key:** Description REFERENCES Description in Types Relation

**Relation:** Types

Attributes

Prop\_Type char (1) NOT NULL

Description char (15)

**Primary Key:** Prop\_Type

**Relation: Bookings**

Attributes:

Booking\_ID numeric (10) NOT NULL

Prop\_Type char (1) NOT NULL

Tenant\_ID numeric (10) NOT NULL

Date\_From date NOT NULL

Date\_To date NOT NULL

Booking\_Status char (1) NOT NULL

Amount\_Paid numeric (10) NOT NULL

**Primary Key:** Booking\_ID

**Foreign Key:** Prop\_Type REFERENCES Prop\_No in Properties Relation

**Foreign Key:** Tenant\_ID REFERENCES Tenant\_ID in Tenants Relation

**Relation:** Payments

Attributes:

Payment\_ID numeric (10) NOT NULL

Amount int (20) NOT NULL

Date\_Paid date NOT NULL

**Primary Key:** Payment\_ID

# Conclusion

The aim of my project was to generate a system that can be used by the admin to add properties for rent. It contains specific user and system requirements which describe in detail the functions that the system must perform. Diagrams were used to show the interaction between each process and the databases and how the databases related to one another.

I created screen prototypes using Visual Studio to show how the software should look and what actions were taking place behind the user interface. An important factor in creating the prototypes was making them user friendly and easy to understand.

An extra function that could be added to the system in the future is the ability to record the details of all individual guests that left without checking out. This function would bring back their booking ids and update their status to C-out..

# Appendices

## Appendix A – Title

## Appendix B – Title

Might include:

* **Lookup / Reference tables**
* **Sample reports / Listings**