Wydział Matematyki i Nauk Informacyjnych

Politechnika Warszawska

Title	Project #2: Hostel Management
Subject	Enterprise Applications in .NET Framework
Year	2016-2017
Version	1.0
Authors	Mario Juez Gil juezm@student.mini.pw.edu.pl Maryan Plakhtiy plakhtiym@student.mini.pw.edu.pl

Revision history

Date	Author	Changes	Revision
18/11/2016	Mario Juez	First version of this document.	1.0

Table of contents

1 Project #2: Hostel Management 3
1.1 Business Description 3
1.1.1 Technologies 3
1.2 Functional Requirements 4
1.2.1 Use Case Diagram 4
1.2.2 User Stories 5
1.3 Non-Functional Requirements 6
1.4 Schedule 6

Table of figures

Figure 1: Use Case Diagram. 4

Table 1: Functional requirements. 5
Table 2: Project schedule. 6

1 Project #2: Hostel Management

1.1 Business Description

The second project product will be a desktop application for generic hostel management. We understand hostel management as guest check-ins and bedrooms management by a hostel manager.

Guest check-in will be a form that the manager will have to fill up with guest personal data like ID number, name, surname, birth date, and sex. That form will also contain check-in and check-out dates, and room number fields. Depending of bedroom size, the form will allow to the manager register one or more guests into the same check-in.

After check-in storage, the manager will be able to generate and print an invoice for the guest.

For bedroom management, the application will have a view with bedroom listing including filtering options by size, price, bathroom type, bed type and availability. Through that view, manager will be able to open the bedroom editor for creating or modifying a bedroom, remove bedrooms, or open the check-in registration view for one specific bedroom.

Also there will be a historical check-in list where the manager will be able to see all check-ins, or all check-ins between two dates with information about the total earnings of the hostel with the check-ins in that list.

In order to track operations and avoid usage of non-authenticated users, hostel manager login will be obligatory to use the application.

1.1.1 Technologies

This project is going to use the following technologies:

1. **Windows Presentation Foundation (WPF)**: User Interface.
2. **Windows Communication Foundation (WCF)**: Communication.
3. **NoSQL Database MongoDB**: Database.

1.2 Functional Requirements

This part of the document, defines project functional requirements by usage of use case diagrams and user stories.

1.2.1 Use Case Diagram

The following figure shows project use case diagram.

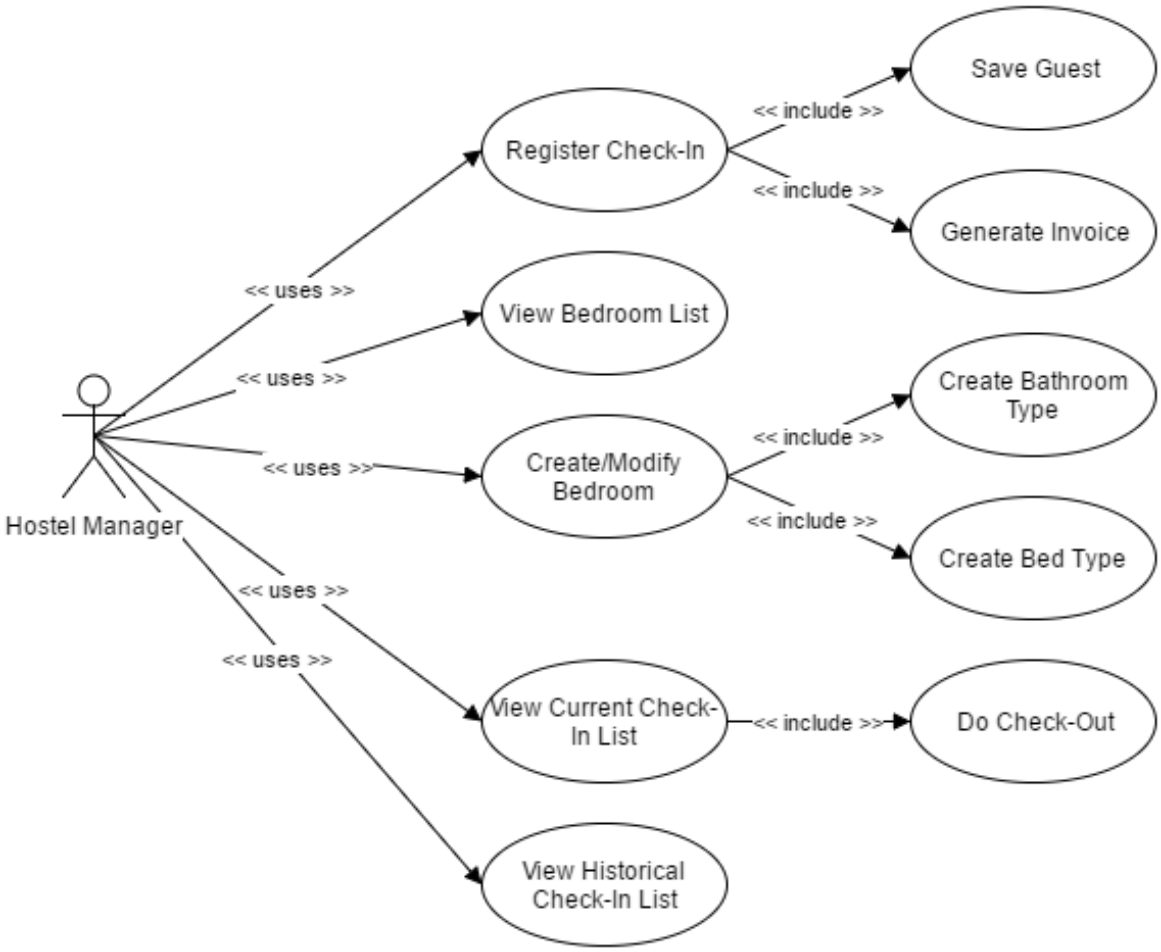

Figure 1: Use Case Diagram.

Actor	Name	Description
Hostel Manager	Register Check-In	The user will enter check-in and check-out date, select the bedroom associated to, and finally the guest information.
	Save Guest	The user will fill the Guest information (ID, Name, Surname, Birth Date and Sex) and then it will be stored when saving the check-in.
	Generate Invoice	After saving a check-in, the user will be able to generate an invoice of that check-in, the invoice will contain the total price of guest accommodation and guest information.
	View Bedroom List	The user is able to view the hostel bedroom list, this view provides filtering options to filter the rooms by its features like price, bathroom type… The list will show information about bedroom availability. Inside this view, user will find buttons to create a new room, and modifying or deleting existing ones.
	Create/Modify Bedroom	If bedroom does not exist, user will enter bedroom number, bed number and type, bathroom type, and regular price per night into a blank form. That form will contain bedroom data if the user is modifying it. Bathroom and bed type fields has to have autocomplete feature for those types which already exists.
	Create Bathroom Type	If the introduced bathroom type does not exist in the database, the system has to create it.
	Create Bed Type	If the introduced bed type does not exist in the database, the system has to create it.
	View Current Check-In List	The user is able to view the hostel current check-in list (without check-out). In this view, the user can remove a check-in (check-out).
	Do Check-Out	The user is able to remove a check-in (check-out operation) It will change the check-in state from active to inactive.
	View Historical Check-In List	The user is able to view the historical check-in list, and he can filter that list between two dates. Information about the hostel earnings is visible in this view.

Table 1: Functional requirements.

1.2.2 User Stories

1. Check-In Registration:
   As hostel manager, I need an agile method to register guest check-ins, that allows me to enter guest required information as quick as possible.
2. Guest Storage:
   As hostel manager, I need the system to store all hostel guest data in order to comply with the law.
3. Invoice Generation:
   As hostel manager, I need the application to generate and print invoices for my clients, because sometimes they want an accommodation receipt.

4. Bedroom Listing:
   As hostel manager, I need an easy method to list all bedrooms of the hostel and see its features and availability, in order to offer my clients the bedroom that best fits their needs.
5. Bedroom creation and modification:
   As hostel owner, I have a hostel chain with different hostels placed in different cities. I need flexible solution which allows my hostel managers to configure each specific hostel bedrooms. Besides, bedroom features like price are subject to change along the time, and managers should have the possibility to modify it.
6. Bathroom types:
   As hostel owner, bedrooms in my hostels has many bathroom configurations, like private bathroom + private shower, shared bathroom, private bathroom + shared shower etc.
7. Bed types:
   As hostel owner, bedrooms in my hostels has many beds configurations, like single bed, double bed, two single beds, bunk bed, etc.
8. Check-In Listing:
   As hostel manager, I need a method to list current check-ins without check-out.
9. Check-out Operation:
   As hostel manager, when a guest leaves the hostel, I need to remove the check-in from the current check-in list, in other words, I need to do a check-out.
10. Historical Check-in Listing:
    As hostel manager, I need a method to list historical check-ins or check-ins that where between two dates in order to know the hostel earnings with those check-ins.

1.3 Non-Functional Requirements

Non-functional requirements are the following:

1. Interface:
   a. Desktop application must work properly on screens with a minimum resolution of 1366x768.
2. Portability:
   a. Desktop application must work properly on Microsoft Windows 10 and above.
   b. Business logic must be a different application with public WCF services that will be used by desktop application.
3. Security:
   a. User must authenticate into the application with valid credentials.

1.4 Schedule

Date Range	Task
17/11/2016 – 18/11/2016	Initial documentation definition.
19/11/2016 – 25/11/2016	WCF technology learning
19/11/2016 – 25/11/2016	Invoice feature implementation
25/11/2016 – 06/11/2016	Migration of business logic to WCF
06/11/2016 – 09/12/2016	Presentation
09/12/2016 – 14/12/2016	Tests and bugfixes.
14/12/2016 – 16/12/2016	Final documentation and Delivery.

Table 2: Project schedule.