Hotel Reservation System

Design

GROUP 4
Irina Erofeeva
Durali Alagöz
Ömer Denizoğlu
M.Mert Dervişoğulları

Revision History

Date	Author	Description	Version
03.05.2021	Irina Erofeeva Ömer Denizoğlu Durali Alagöz M.Mert Dervişoğulları	First version	v1.0
09.05.2021	Irina Erofeeva Ömer Denizoğlu Durali Alagöz M.Mert Dervişoğulları	Updated according to Iteration2 reviews.	v1.1
20.06.2021	Irina Erofeeva Ömer Denizoğlu Durali Alagöz M.Mert Dervişoğulları	Small grammar fixes	v1.2

Design structure 3

Patterns 3

1. Singleton 3

Overview 3

Structure 3

2. Composite 4

Overview 4

Structure 4

3. Adapter 4

Overview 4

Structure 4

Requirement realizations 5

1. Realization of UC1: Manage Accounts 5

Basic scenario 6

2. Realization of UC2: Manage Reservation of Room 6

Basic scenario 6

3. Realization of UC3: Manage Reservation of Room Cleaning 6

Basic scenario 7

4. Realization of UC4: Manage Reservation of Other Services 7

Basic scenario 7

5. Realization of UC5: Make Payment in Cash 7

Basic scenario 8

6. Realization of UC6: Make Payment by Credit Card 8

Basic scenario 8

7. Realization of UC7: Monitor Reservations 8

Basic scenario 9

8. Realization of UC8: Manage Hotel Services 9

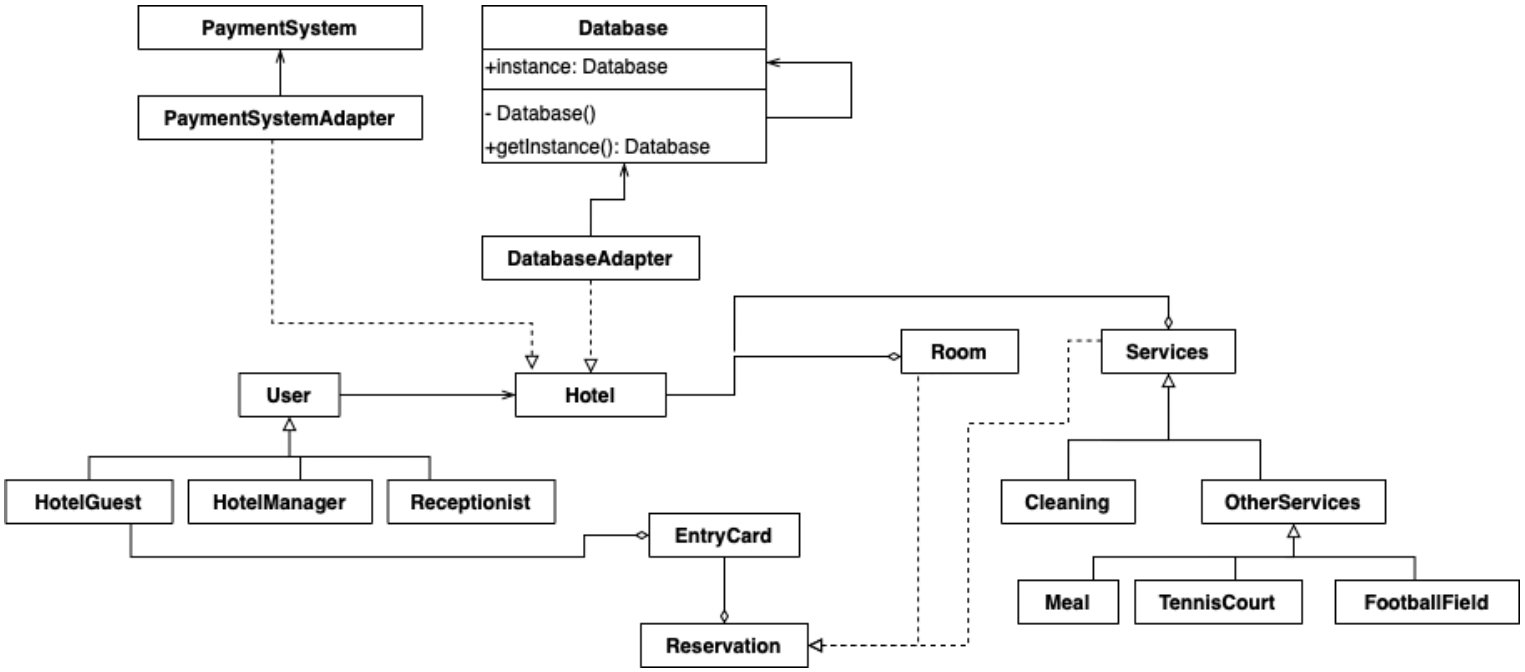
Basic scenario 10

9. Realization of UC9: Manage Users 10

Basic scenario 10

Design structure

This section contains high level design diagram, that was constructed based on Domain model together with Design patterns chosen for implementation of the system. The detailed information about design pattern is written in the following section of this document and Domain Model is taken from Architecture Notebook.



Patterns

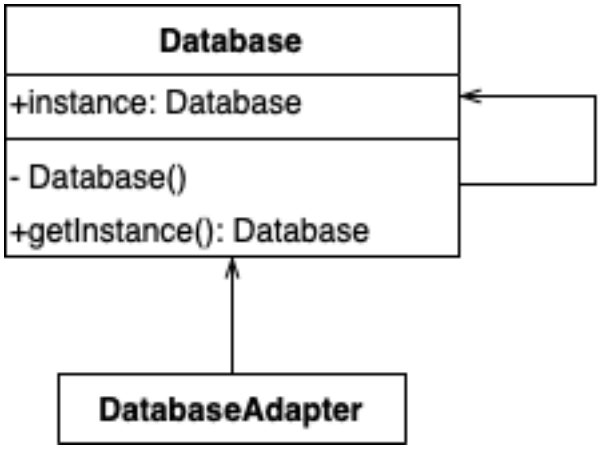
1. Singleton

Overview

Intent: Singleton is a creational design pattern that lets you ensure that a class has only one instance, while providing a global access point to this instance.

Motivation: There should be a single database object, so that different parts of the program share the same instance. Because of this there will be no conflicts while communicating to the same resource.

Structure



2. Composite

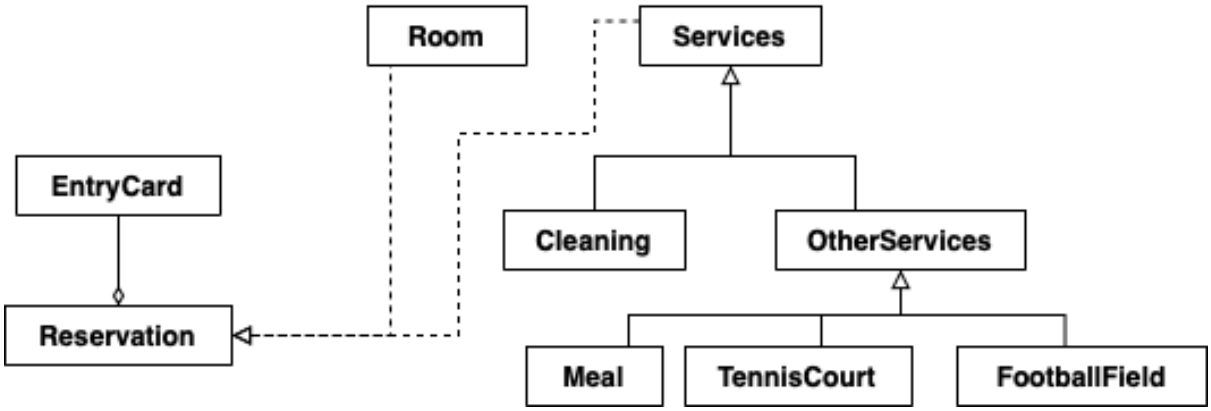
Composite. In order to calculate the bill - to show what was used in creating the total debt.

Overview

Intent: This pattern is used in order to combine element into tree structure and treat them as one.

Motivation: Composite pattern is going to be user in order to show a list of things that was used in creating total debt.

Structure



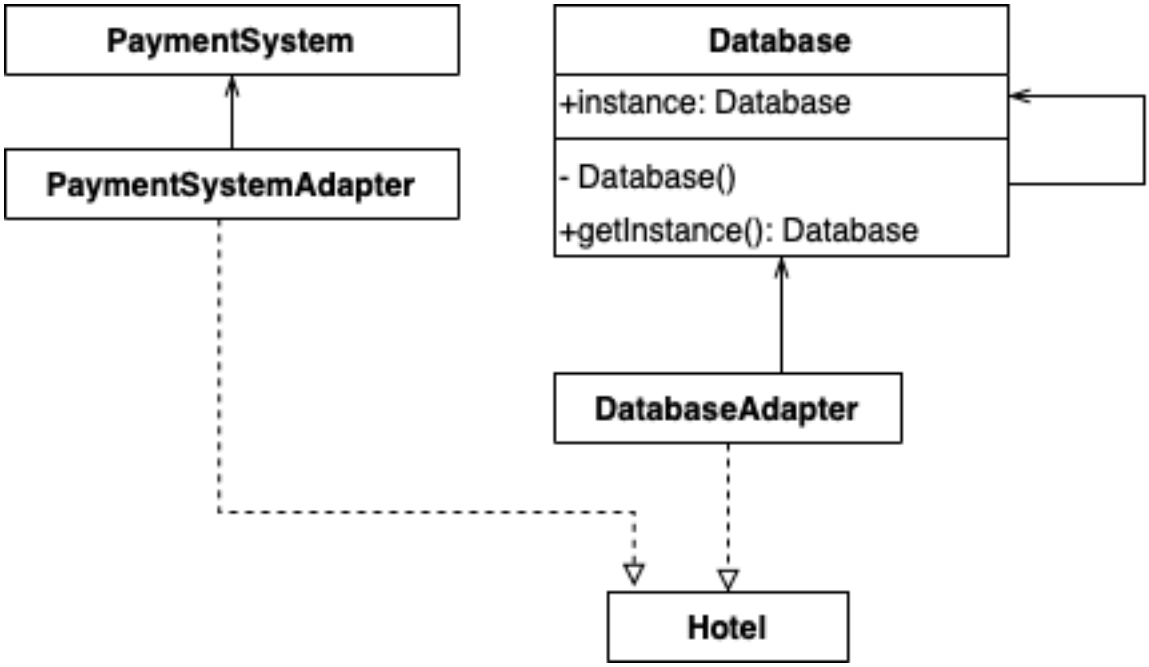
3. Adapter

Overview

Intent: Using this allows objects with not matching interfaces work together through adapter.

Motivation: Payment System is the external system for HRS, it's function should be adapted in order to retrieve them. Database also needs to be adapted with the interface that the main system may use.

Structure

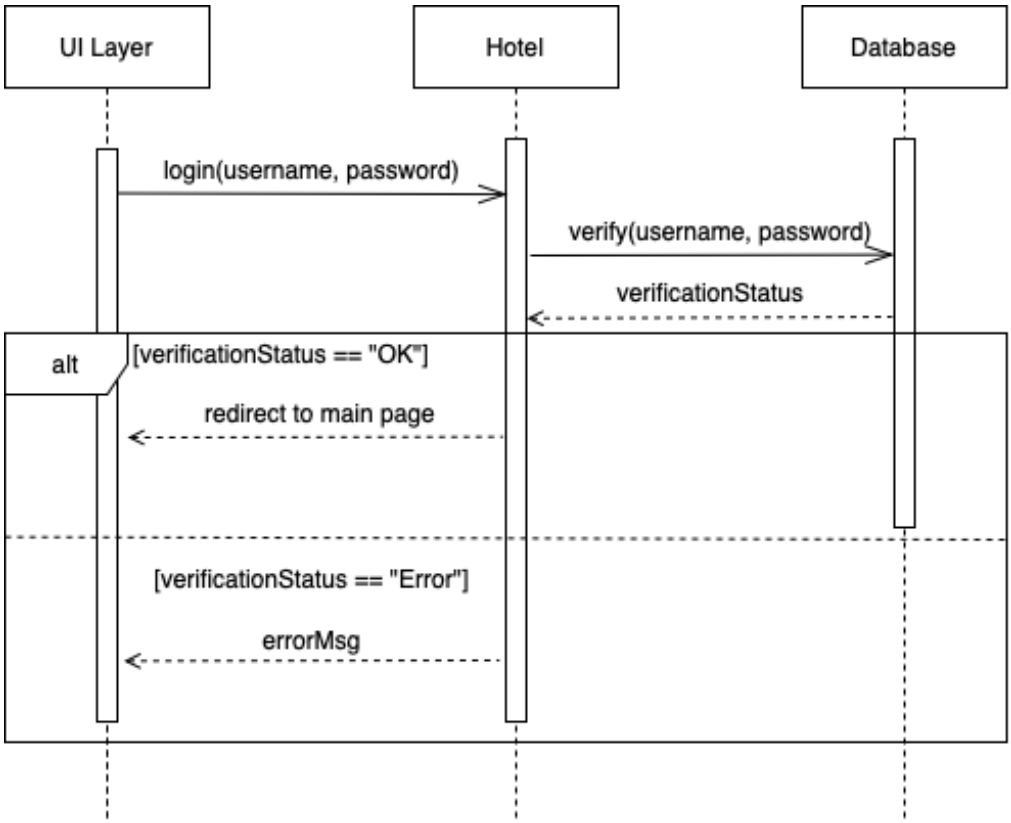


Requirement realizations

In this section the interaction and sequence diagrams for the main success scenarios of the use cases and the most frequently appearing extensions.

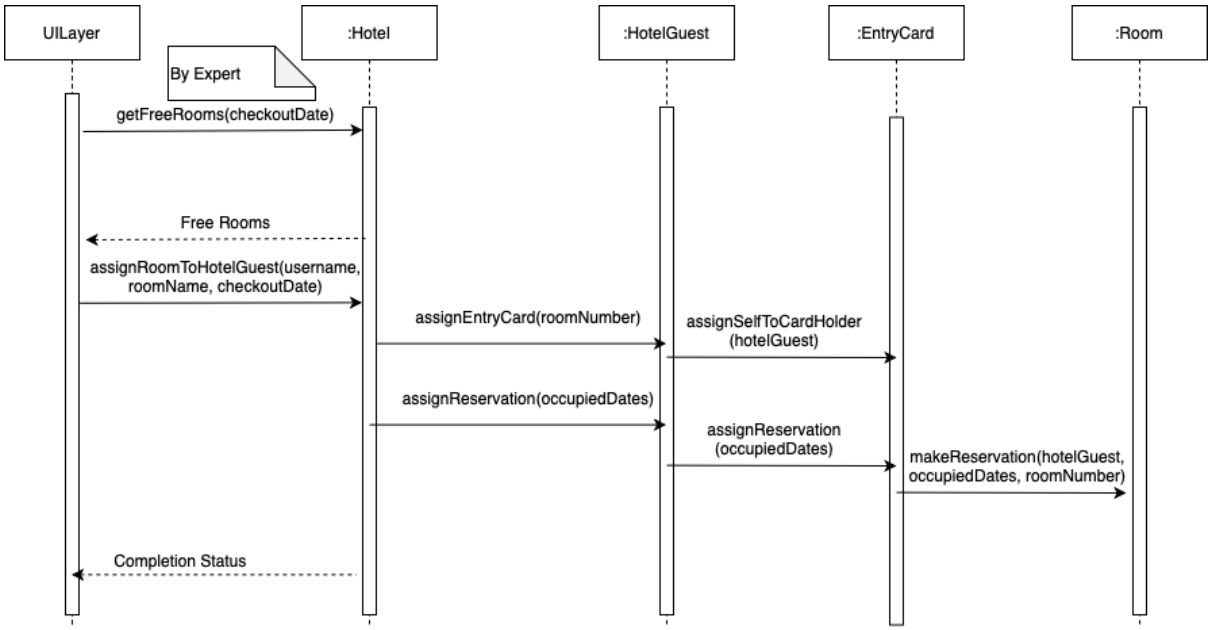
1. Realization of UC1: Manage Accounts

Basic scenario



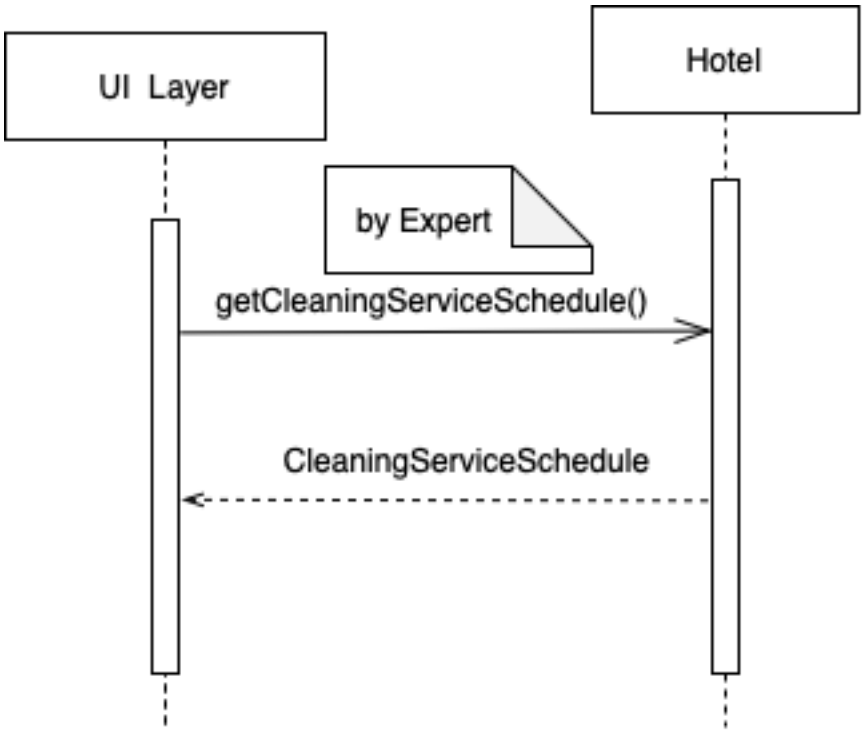
2. Realization of UC2: Manage Reservation of Room

Basic scenario



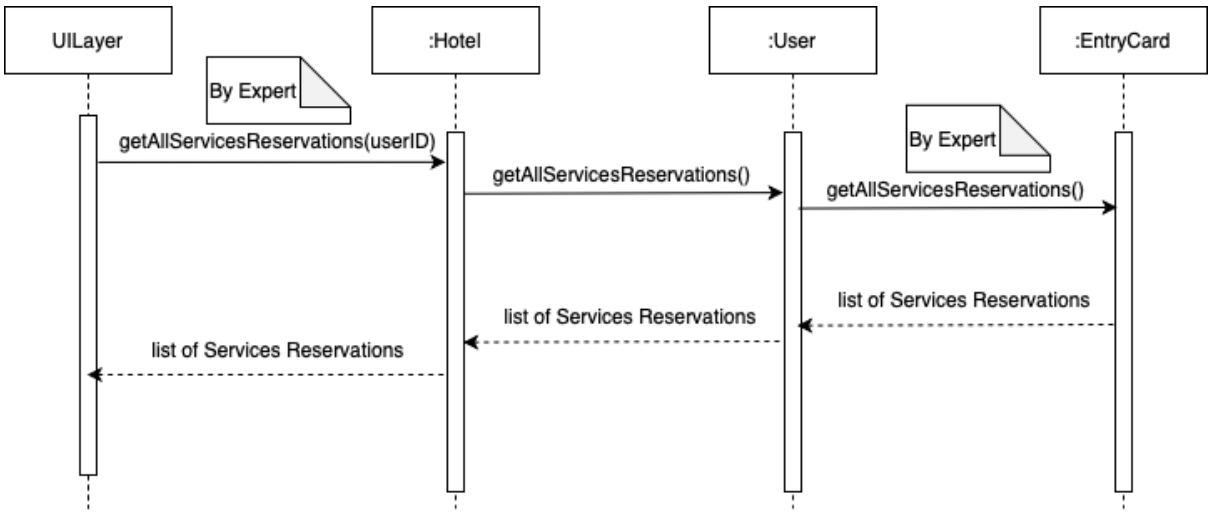
3. Realization of UC3: Manage Reservation of Room Cleaning

Basic scenario



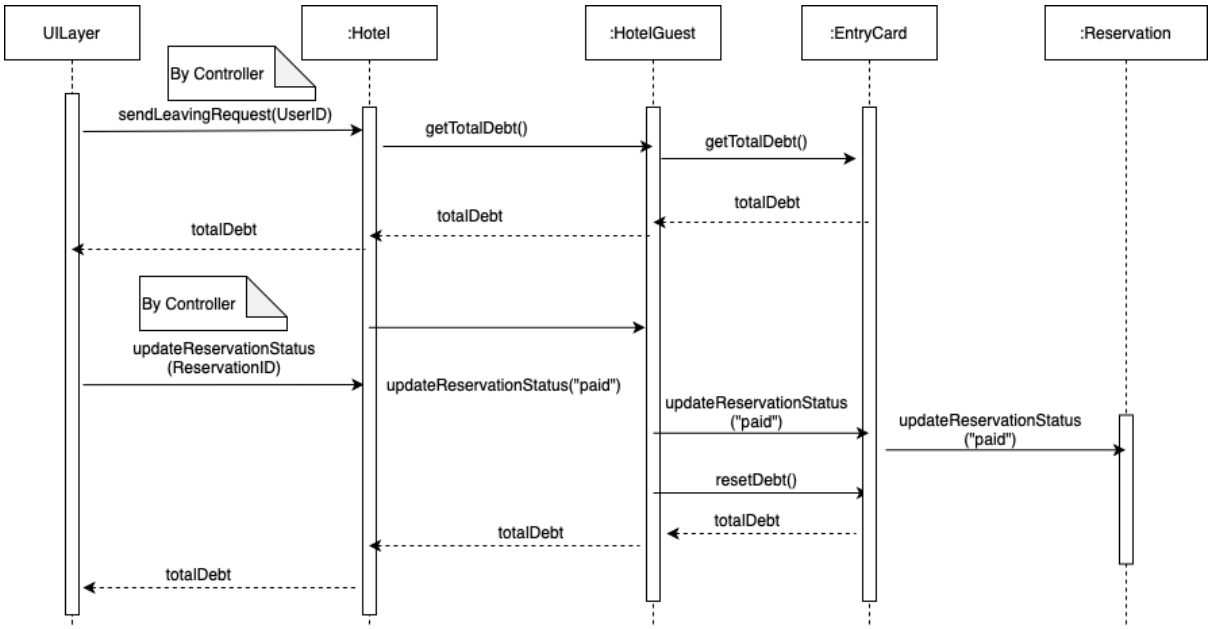
4. Realization of UC4: Manage Reservation of Other Services

Basic scenario



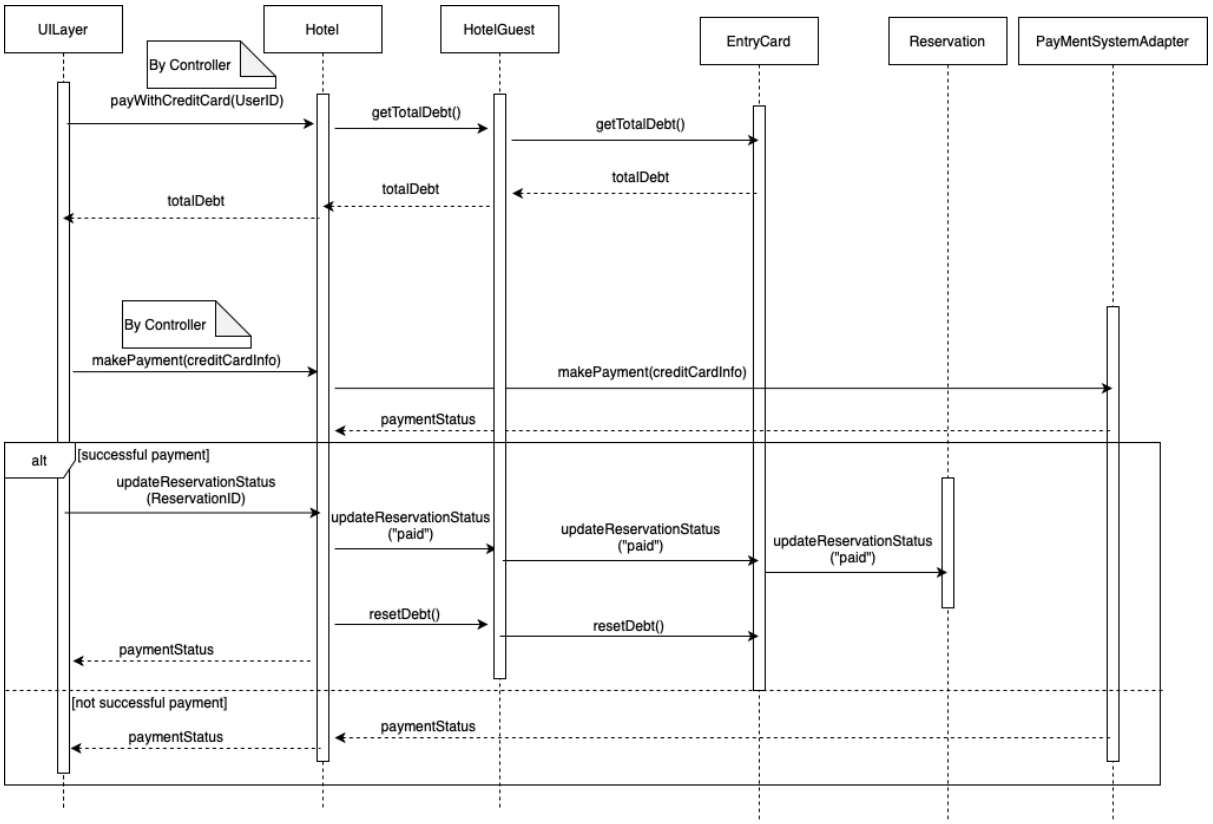
5. Realization of UC5: Make Payment in Cash

Basic scenario



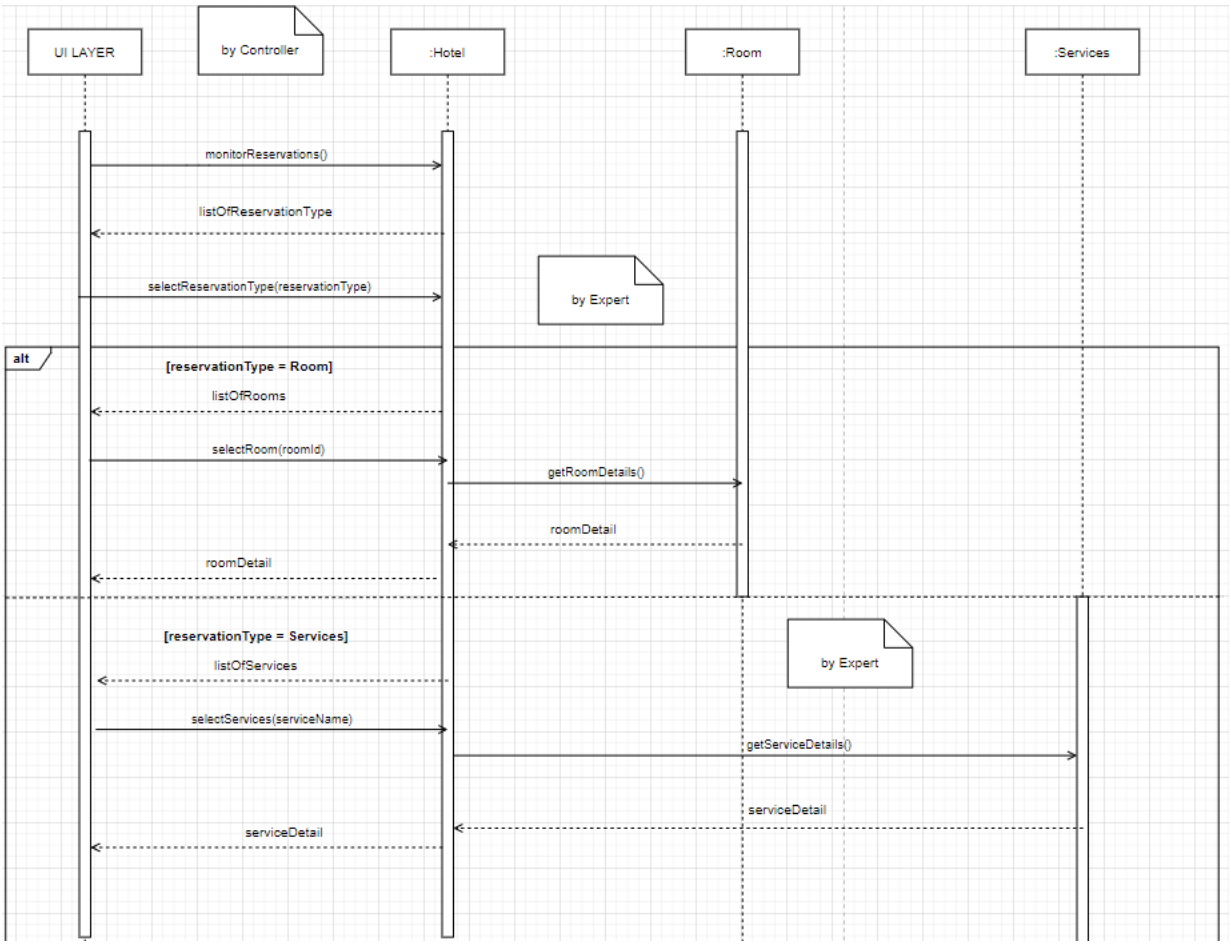
6. Realization of UC6: Make Payment by Credit Card

Fully Extended Scenario



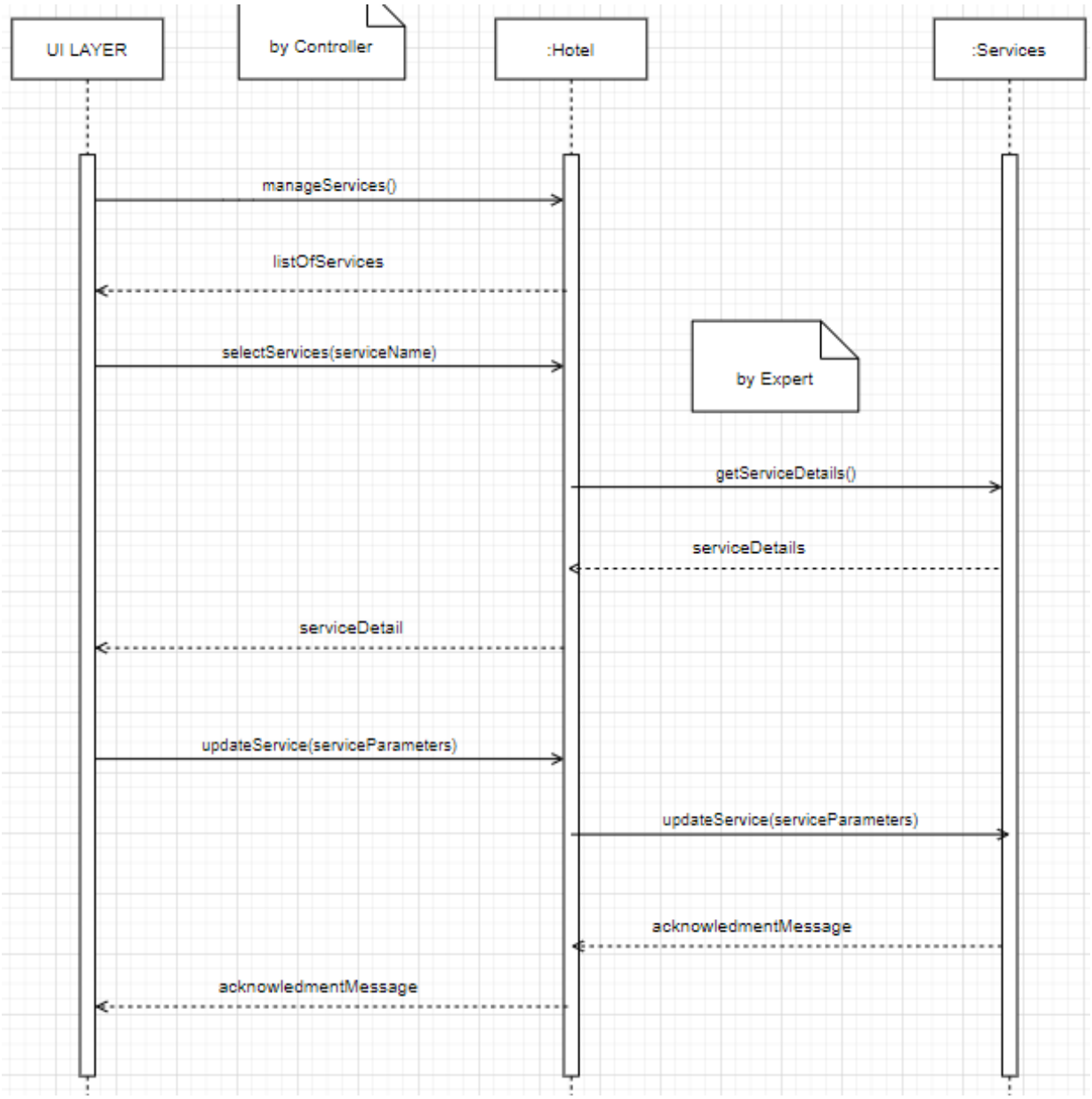
7. Realization of UC7: Monitor Reservations

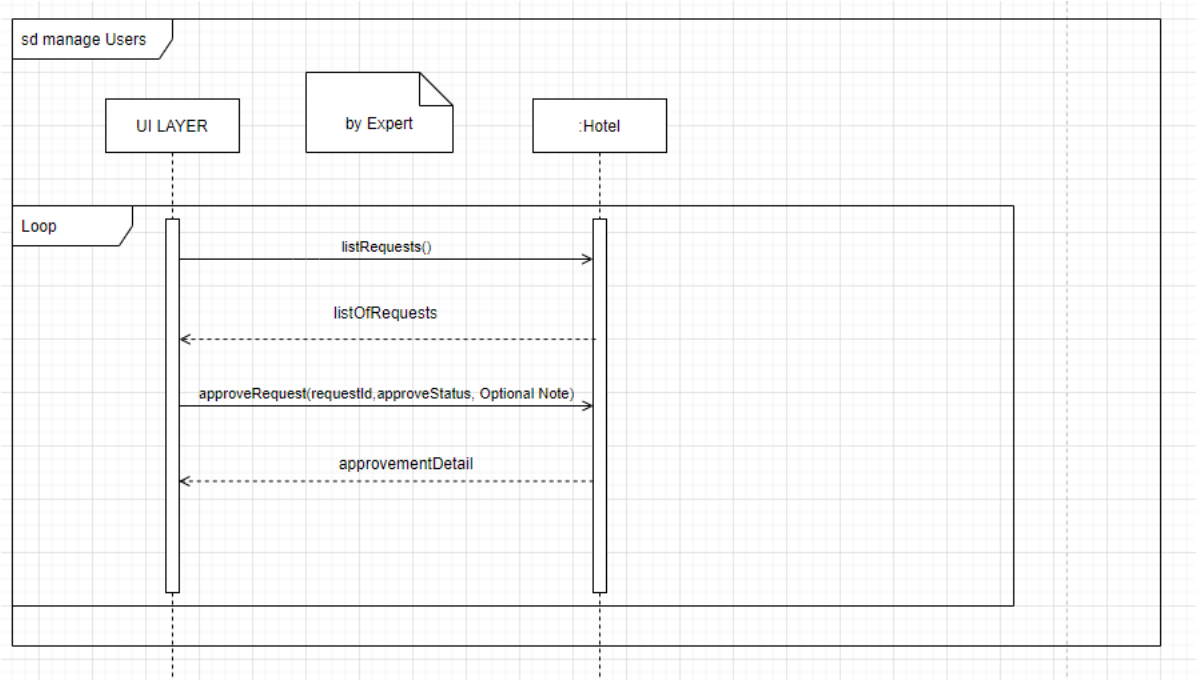
Basic scenario



8. Realization of UC8: Manage Hotel Services

Basic scenario



9. Realization of UC9: Manage Users

Fully Extended Scenario