Software design description for content management project for tourism agency

Ramez Nabil, Khaled Nader ,Nour Eldeen Mohamed, Ahmed Mahdy

May 10, 2020

SDD Version	Date	Reason for Change
1.0	20-April-2020	SDD First version's specifications are defined
1.1	28-April-2020	Scope updated
1.2	06 - May - 2020	Edited Project scope, Goals and objectives and added photos to interface viewpoint and human interface design
1.3	07 - May - 2020	Created context diagram and architecture diagram and updated Logical Viewpoint with it's rationale. Update use case diagrams
1.4	08 - May - 2020	Updated class diagram and divided it to patterns with explanation of each pattern. Created sequence diagrams. Updated algorithm viewpoint
1.5	09 - May - 2020	Added Project timeline and requirements matrix

Table 1: Document version history

GitHub: https://github.com/Ramez007/Tourism-Project

1 Introduction

1.1 Overview

This is an SDD (Software Design Document) describes architecture and system design of content management for Speedo Tours travel agency, a system that aims to enhance booking online for travel packages and hotels. It details the components of the system, the architecture used in creating it. How the data is

handled and the relations and design between the data. It also explains the algorithms used throughout the system.This document explains all these aspects with the help of Class diagrams, Database diagram, Use case diagrams, Context diagram and Sequence diagrams.

1.2 Scope

This web application aims to help facilitating booking online and creation of a suitable database to store the data of packages and hotels communicating with the travel agency and also to store guests data for using them when booking. It is run by admin, support operators, and accessed by the guests . It allows the admin to edit the contents of the web application from the Admin's page,the admin has several tabs that allows him to preform different operations on the web application like editing content of main page,editing content of about us page,preform operations on hotels and packages like adding,editing and suspending package or hotel.The support operator can send news to subscribed emails to newsletter and send some information about a user's inquiry.The guest book packages or hotels and preform operations on the profile page like editing profile,view reservations history,track reservation or cancel the reservation.

1.3 Purpose

This software design description (SDD) describes the architecture and system design of Speedo tours web application. The architecture used in this web application is the MVC model which is the model view controller,the main purpose of using this model is to facilitate the reading of the code for further development and maintenance , also it lessens the heavy use of the database as it uses only one connection throughout the whole system.

1.4 Intended audience

There are variety of stakeholders acting and interacting with the website. The users include guests that surf the website to viewing out hotels and packages and they shall be able to book any of those packages or hotels.Admins also can interact with the website by confirming reservations or editing any page in the website. Also support operators of the company are important stakeholders as they are able to send various mails to guests and reply to their inquires. The system shall be secured by encrypting guests passwords and restricting exposition of data.

2 Project Overview

The system is designed to make reservations for hotels and packages easier and more flexible with easy to use interface and clear guidelines for all the users of the website including the employees of the agency and registered guests. The project sponsor is Speedo Tours which is a travel agency. Users are anyone who is interested in travelling with the agency. The developers are students in Misr International University and their names are stated above in the title of the document.

2.1 Project Scope

Used tools and methods:

- Visual Studio Code
- XAMPP

- GitHub Desktop
- Google Recaptha API
- PHP Mailer Plugin
- Google Login API

Used technologies:

- BootStrap
- jQuery
- JavaScript
- PHP
- SweetAlert JavaScript library

Benefits of the project:

- Easy handling of web application's content
- Easy access and edit to the web application's content
- User-friendly interface for booking and browsing the services offered by the agency in the application.

End result:

- A content management web application for traveling agency (Speedo Tours).

2.2 Goals and objectives

- Provide a user-friendly interface to help users navigate the web application.
- Digitize the traveling agency's data and transactions to maintain stable history records.
- Enable the admin to edit every component in the web application to the agency's needs.
- Provide a user-friendly interface to the support operator to be able to respond to inquiries made by the users and send news wire through email.

2.3 Project Timeline

- Phase 1 (5 February - 15 February): Planning Phase: after getting the requirements from the client. this phase is mainly gathering information about the software needed.
- Phase 2 (15 February - 29 February): Making the first version of SRS with scope, overview, problem statement, functional requirements and system flow and system architecture (class diagram).
- Phase 3 (1 March - 15 March): Starting the front end of the base by choosing the suitable HTML,CSS template and establishing the database to store the data in. Version 1.1,1.2,1.3 of SRS Document.

- Phase 4 (16 March - 31 March): Counting the front-end and establishing new pages needed for our web application (package,hotel,support,profile,admin) and finishing the front-end. Version 1.4,1.5 of SRS Document.
- Phase 5 (1 April - 22 April): Counting the front-end: Removing all hard-coded data and done reading each page data from the database established in previous phases using model,view,controller to achieve the reading from database process.
- Phase 6 (23 April - 26 April): Beginning The Back-end of the project and establishing MVC model and connecting its components with the main-pages. Version 1.7 of SRS Document.
- Phase 8 (26 April - 4 May): Making the core code of back-end by implementing functions, adding APIS, use observer design pattern, begin making the SDD document Version 1. Version 1.8 of SRS Document.
- Phase 7 (4 May - 7 May): Finishing Validations,Security Procedures, Encrypting sensitive date and finalizing documentation of the project.
- Phase 8 (7 May - 10 May): Testing Phase: Testing the whole project and fixing errors and preparing PowerPoint, presentation sequence for discussion (live Demo).

3 Context viewpoint

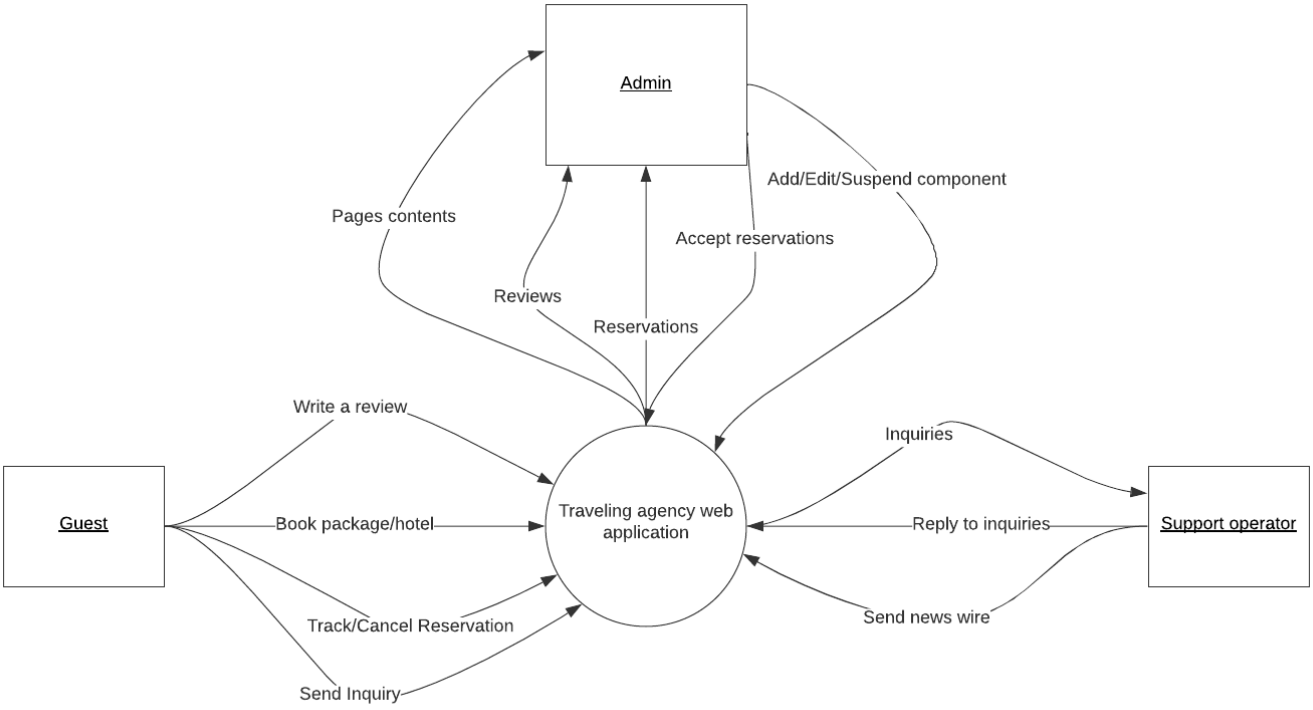

Figure 1: Context Diagram

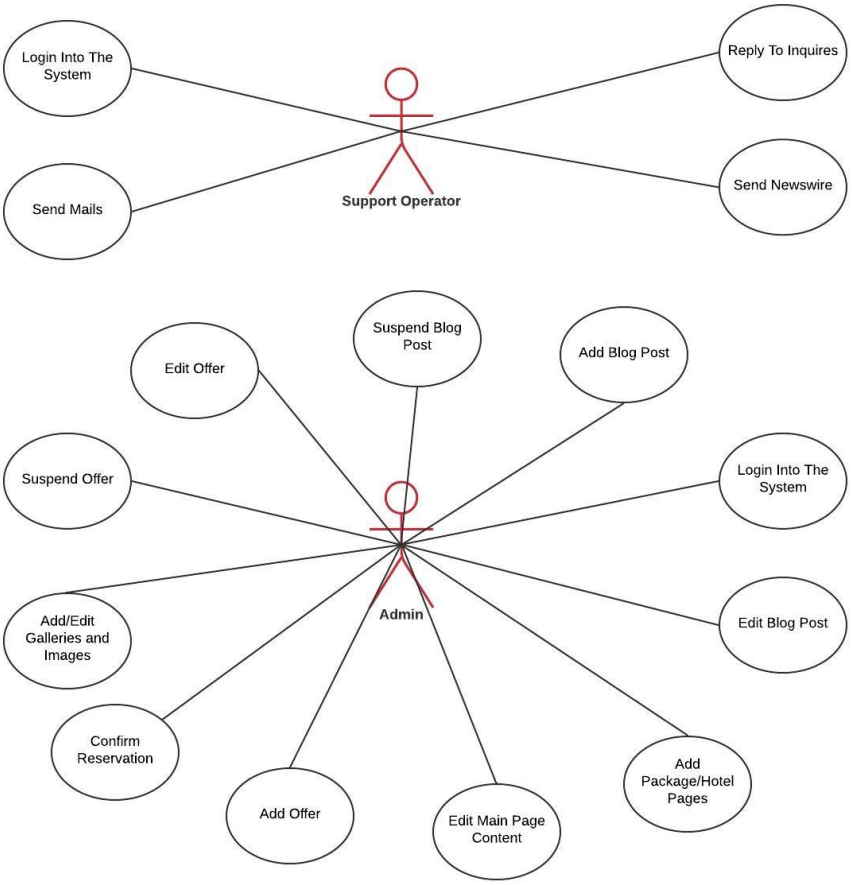

Figure 2: Use case diagram part 1

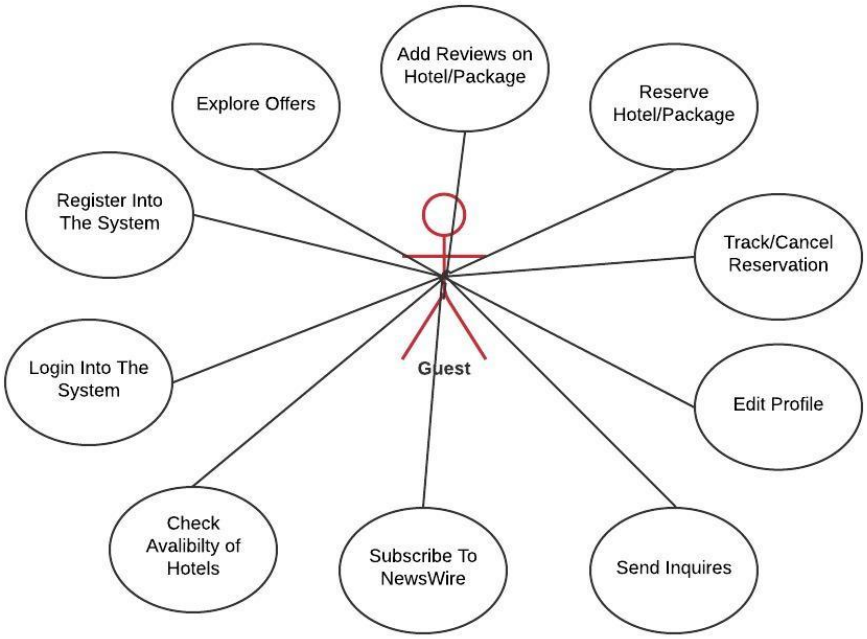

Figure 3: Use case diagram part 2

4 System Architecture Design

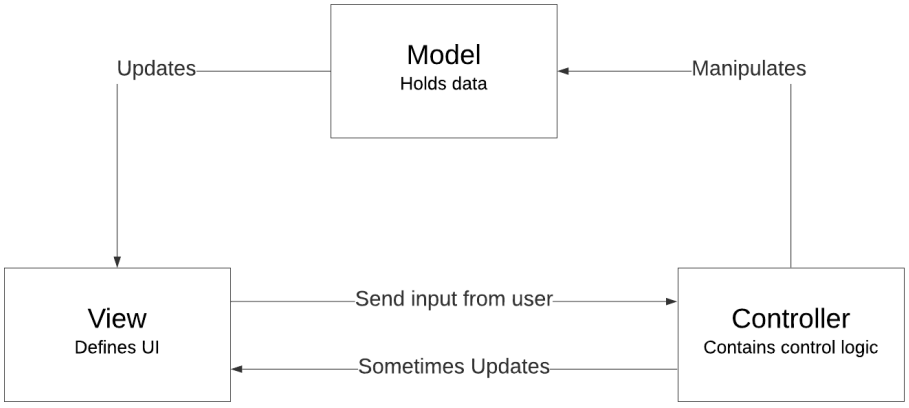

Figure 4: Architectural Design

4.1 Logical viewpoint

The system consists of 3 parts as shown in figure 4. Model is responsible for holding the data retrieved from the database and holds the implementation of functions that use the data retrieved or received and updates the view with the held data, Also the model has means of exception handling in case of any thrown exceptions. The controller coordinates the user's input and choice with the corresponding function in the model and if necessary update the view with the retrieved data from the model. The view contains the user interface and sends the input data to the controller to be handled by the model and then retrieve the updates from the model and therefore output the new data to the user. The process of data exchange between the 3 parts is monitored to ensure data is valid.

4.1.1 Design Rationale

The reason behind selecting the MVC approach is to separate functionality from data and representation in order to avoid damage or unexpected conflicts in data.

4.2 Patterns use viewpoint

4.2.1 MVC Pattern

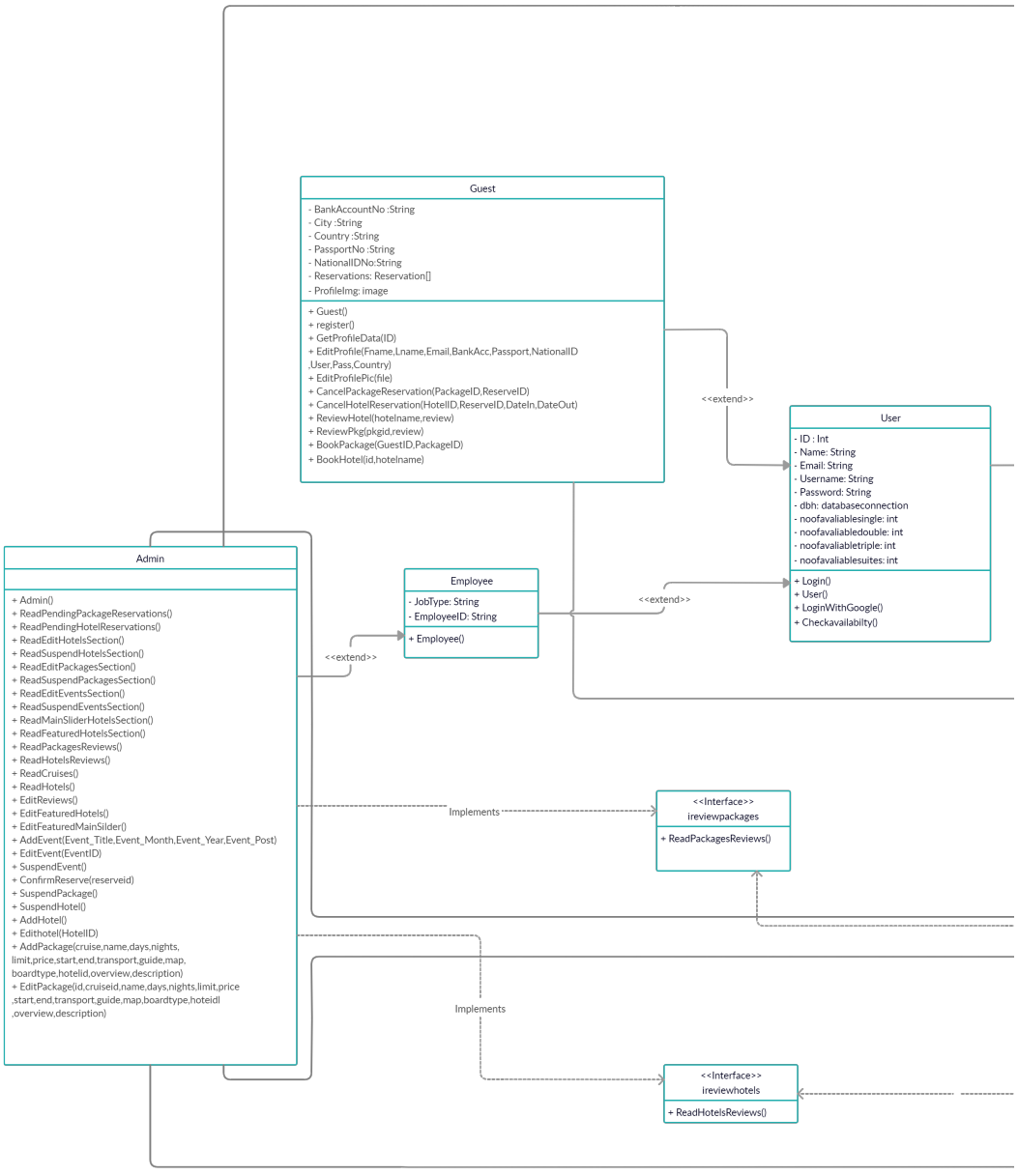

Figure 5: MVC Pattern part 1

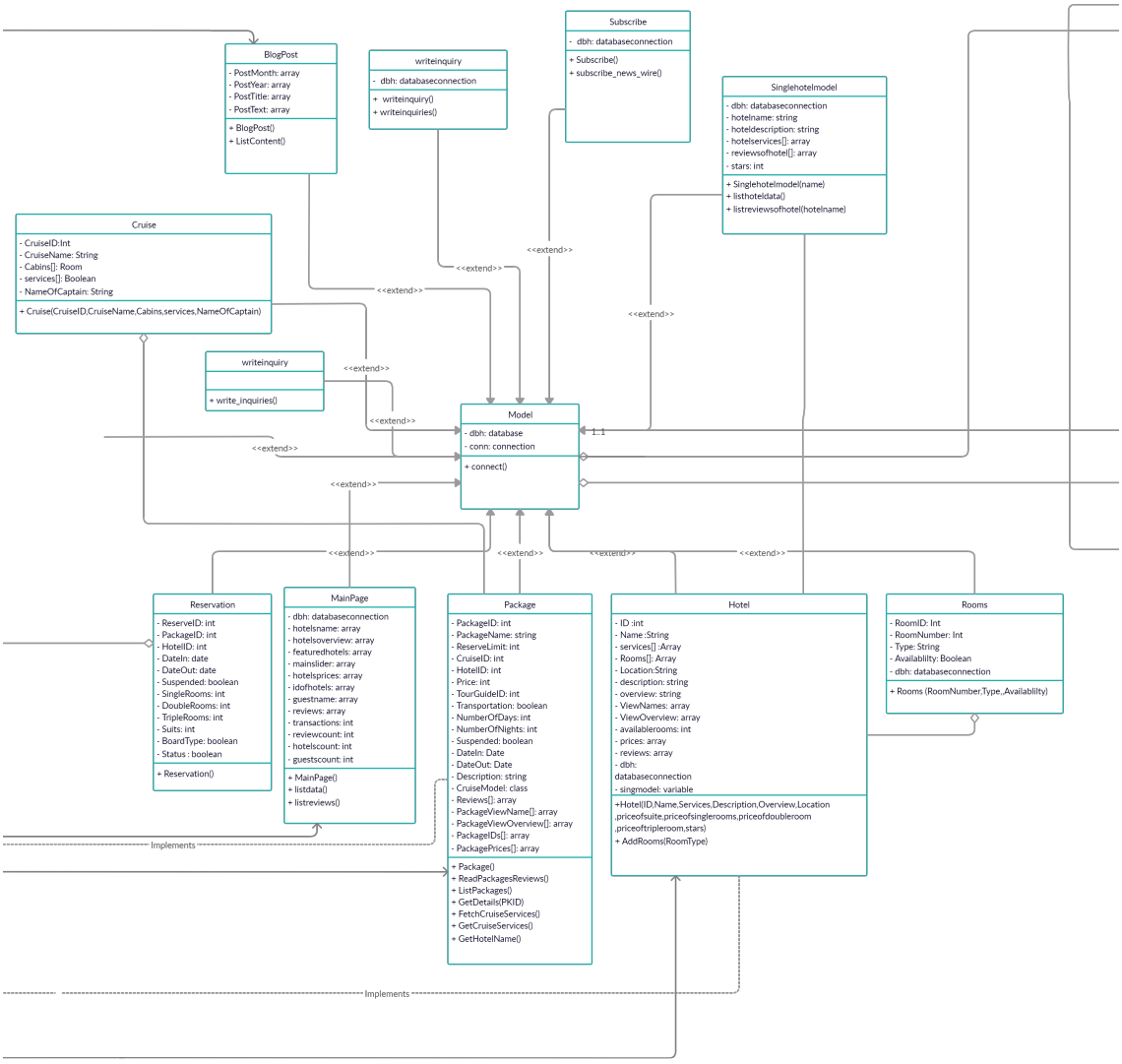

Figure 6: MVC Pattern part 2

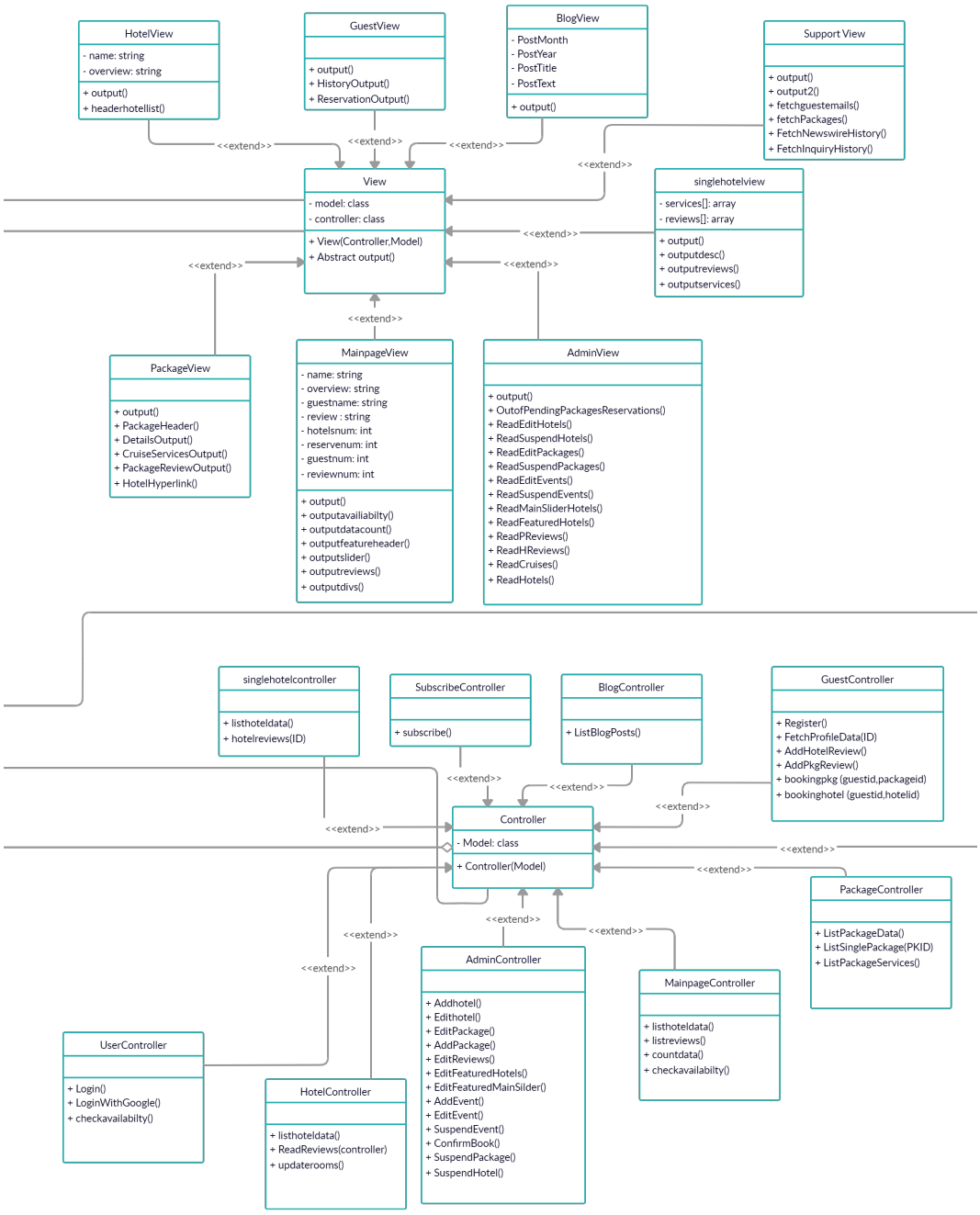

Figure 7: MVC Pattern part 3

The first pattern implemented is the MVC which stands for Model-View-Controller. The pattern works as follows, Every class that holds data and manipulate it or perform function on it extends or inherits from Model super class. Then, the controllers that are responsible for transferring data from View to Model

and calling the desired functions inherit from the Controller super class. Finally, the view class that are responsible for the output of the data stored by the Model inherit from the super class View.

4.2.2 Observer Pattern

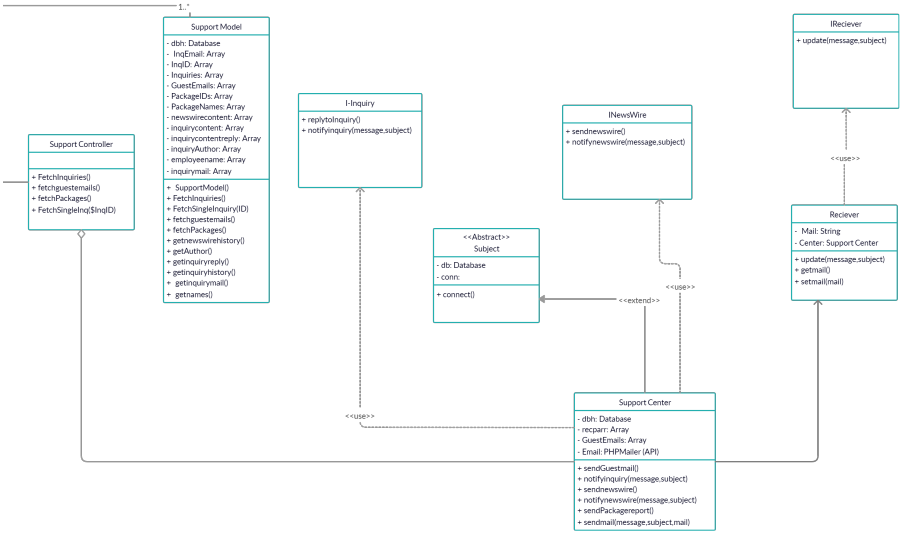

Figure 8: Observer Pattern

Next we implemented the Observer pattern to be able send emails and reply to inquiries using two classes which are the support center which holds the data and send the information and act as the subject and the receivers which are the people who are subscribed into the news wire or sent an inquiry who act as the observers by having an association relation between both classes and support center implements two interfaces which are iInquiry and INewswire and the receiver class which implements interface ireciever.

4.2.3 Design Rationale

The MVC pattern was used to separate functionality from data structures and their representation in order to maintain integrity and keep the data from unintentional damage or corruption. As for the Observer pattern,

we implemented it to be able to provide the users with updates and mails internally with the use of an official mail sanctioned by the company.

4.3 Composition viewpoint

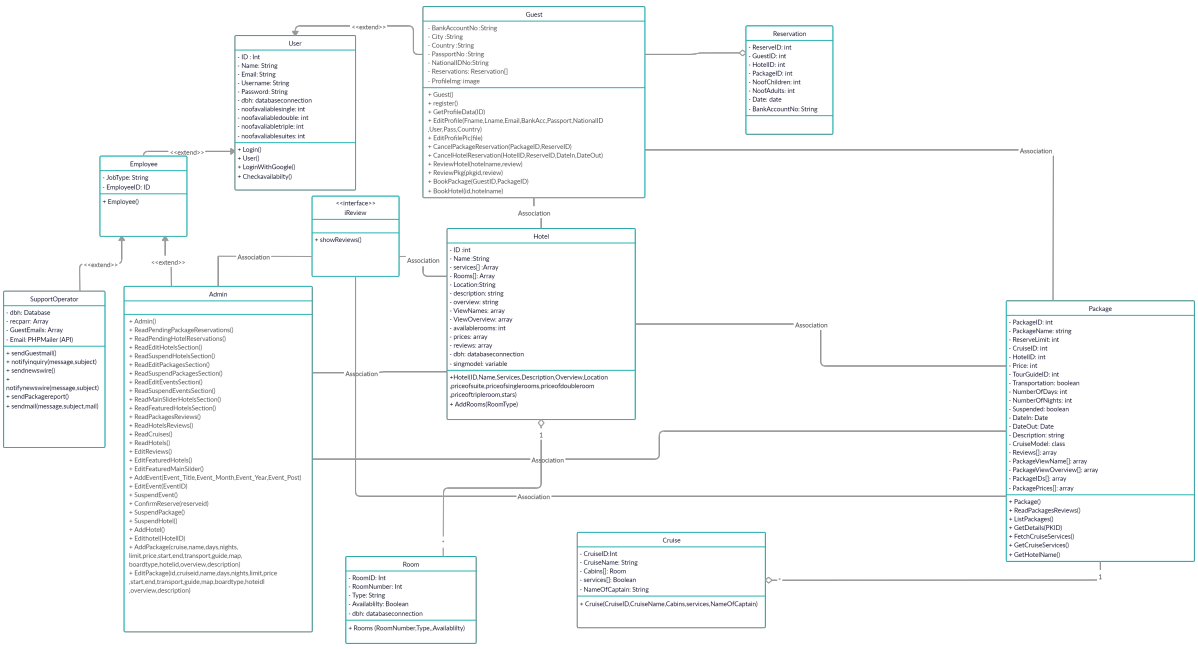

Figure 9: Abstract class Diagram

All users who interact with the system whether guests, system admin or support they inherit from the User super class and then are separated into their respective classes. Then, every class is associated with the components that it uses. First, the guest class. The guest class can make reservations and therefore has an aggregation relation with class Reservation. This reservation can be for a hotel or a package therefore it has associative relation with both hotel and package. A package contains a hotel or several hotels so it has associative relation with hotel. A hotel contains rooms and so it has aggregation relation with rooms. A package can have a cruise or more so it has aggregation relation with class Cruise. The interface shows the reviews made by guests on hotels and packages, And since it is used by essentially any user of the system it was made as an interface.

4.4 Structure viewpoint

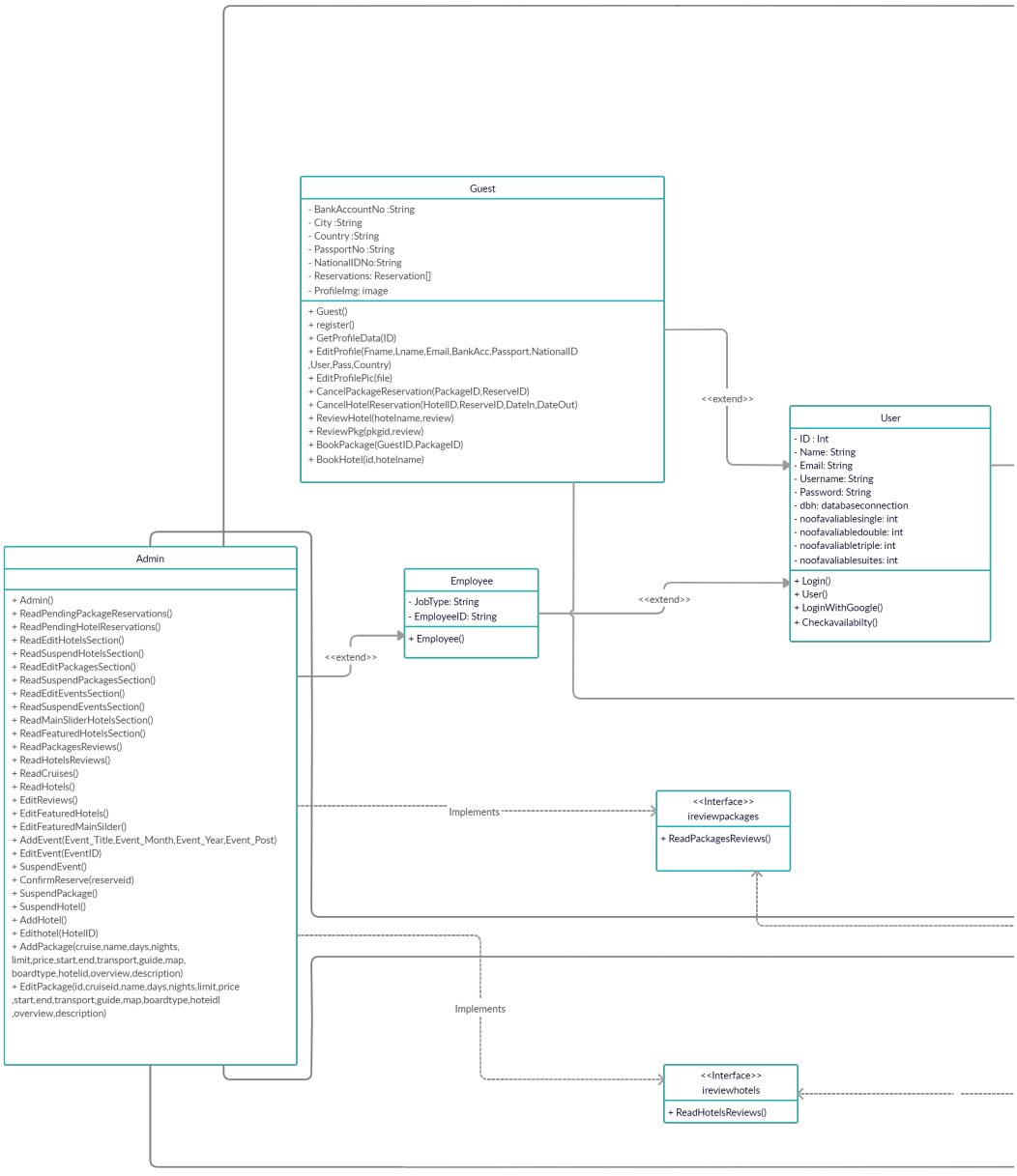

Figure 10: Class Diagram part 1

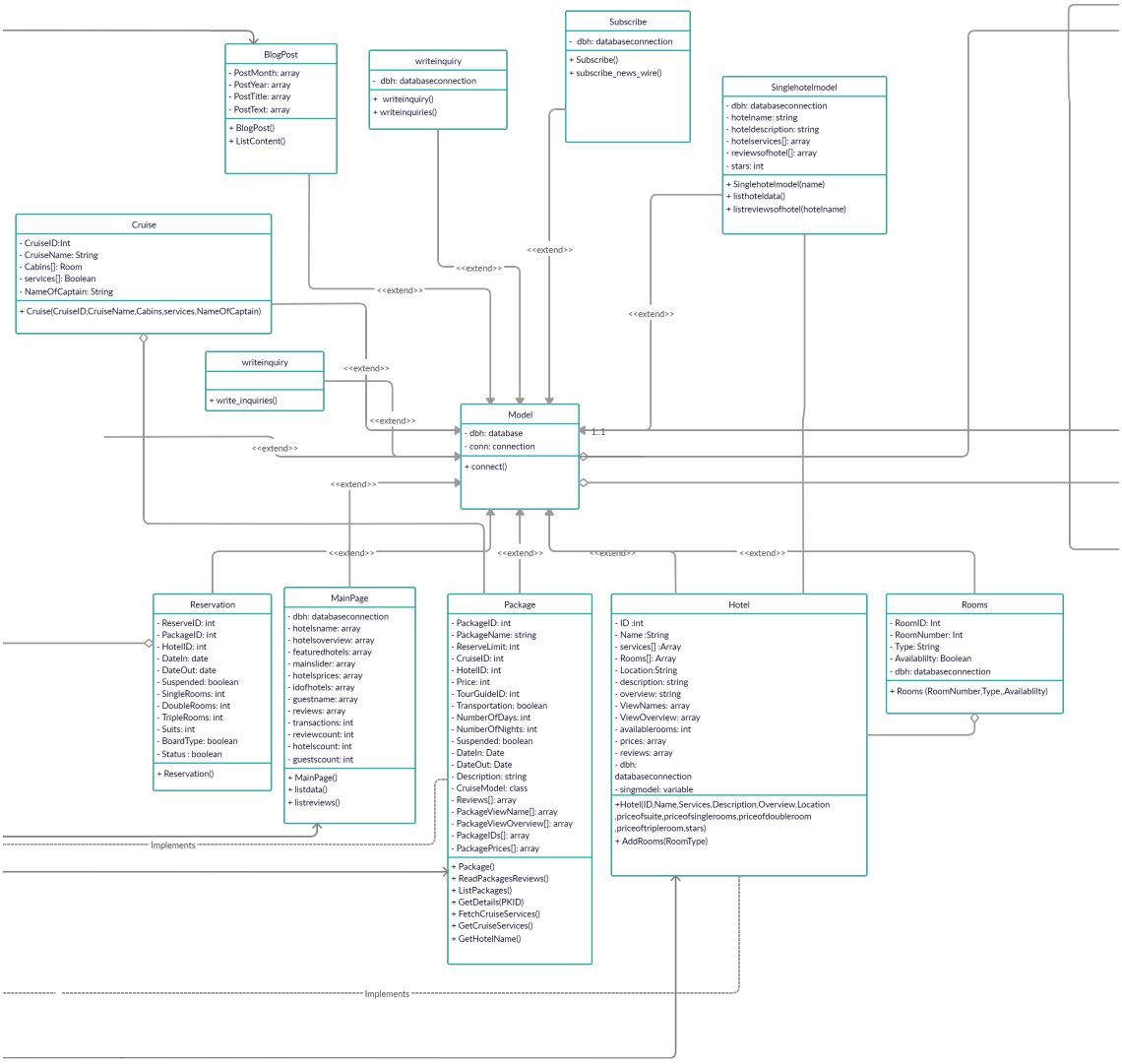

Figure 11: Class Diagram part 2

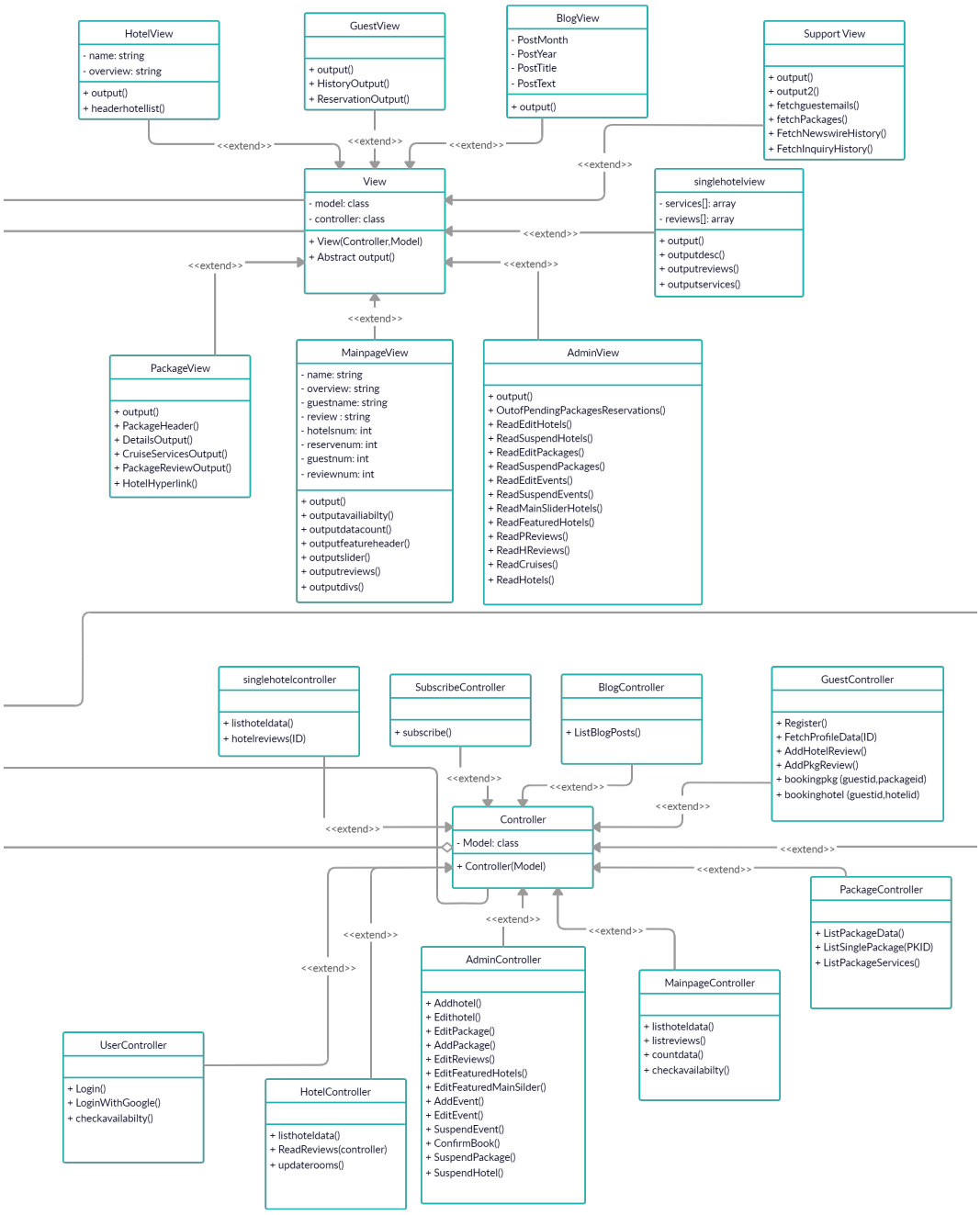

Figure 12: Class Diagram part 3

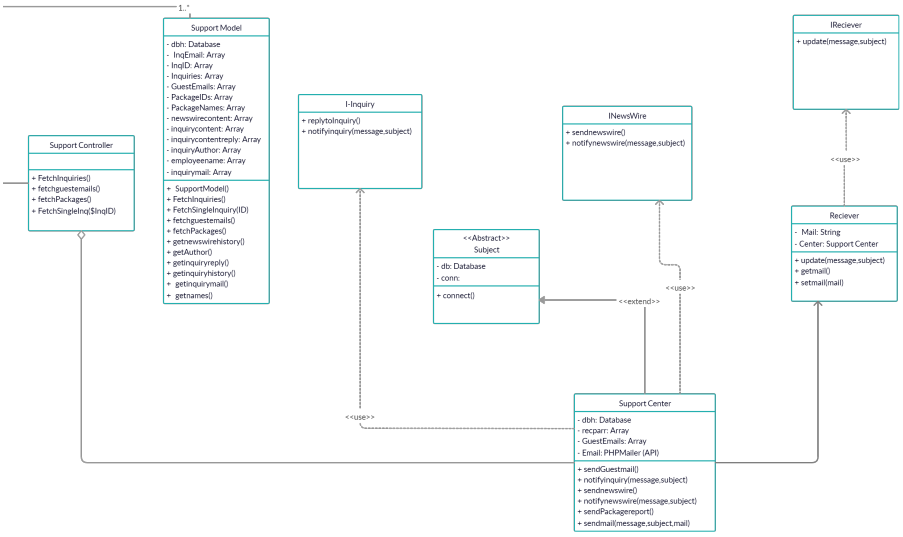

Figure 13: Class Diagram part 4

4.5 Algorithm viewpoint

```
 <div id="edit-hotel-subsec">
  <form action="" method="post">
  <h4 class="text-center">Edit Hotel</h4>
    <?php $AdminView->ReadEditHotels(); ?>
  <input class="btn btn-primary mb-2" type="submit" name='submit-edit-hotel' value="Save
     Editing Hotel">

<script>

 document.getElementById("hotels-editing-dropdown").addEventListener("change",function(){
 var res=document.getElementById("hotels-editing-dropdown").value.split("&");
 document.getElementById("edithotelname").value=res[0];
 document.getElementById("edithotellocation").value=res[1];
var inputs = document.querySelectorAll('.check');
```

```
var val=2;
for (var i = 0; i < inputs.length; i++) {
   if(res[val]=="TRUE")
 {
 inputs[i].checked = true;
 }
 else
{
 inputs[i].checked=false;
}
val++
}
 document.getElementById("edithoteldescription").value=res[9];
 document.getElementById("edithoteloverview").value=res[10];
 document.getElementById("HotelId").value=res[11];
 document.getElementById("pricesingle").value=res[12];
 document.getElementById("pricedouble").value=res[13];
 document.getElementById("pricetriple").value=res[14];
 document.getElementById("pricesuites").value=res[15];


if(res[16]=="1")
{
  document.getElementById("s1").checked=true;
}
 else if (res[16]=="2")
{
  document.getElementById("s2").checked=true;
}
 else if (res[16]=="3")
{
document.getElementById("s3").checked=true;
}
else if (res[16]=="4")
{
   document.getElementById("s4").checked=true;
}
else if (res[16]=="5")
{
   document.getElementById("s5").checked=true;
 }
 });


</script>
</form>
</div>
```

The provided section of code functions as follows. It is responsible for displaying the hotels that exist in the database in a drop down list and when selecting different hotels it changes the subsequent components that have information about the selected hotel. The interface is stored in the ReadEditHotels() function

that is called by the admin view class. The functions performs SQL queries to fetch the hotel data from the database. Every single hotel is fetched and it's component is separated by a "&" symbol. And so, the JavaScript function provided above splits the components of the retrieved data and assign it to it's element in the interface returned by the function ReadEditHotels().

4.6 Interaction viewpoint

4.6.1 Admin functions

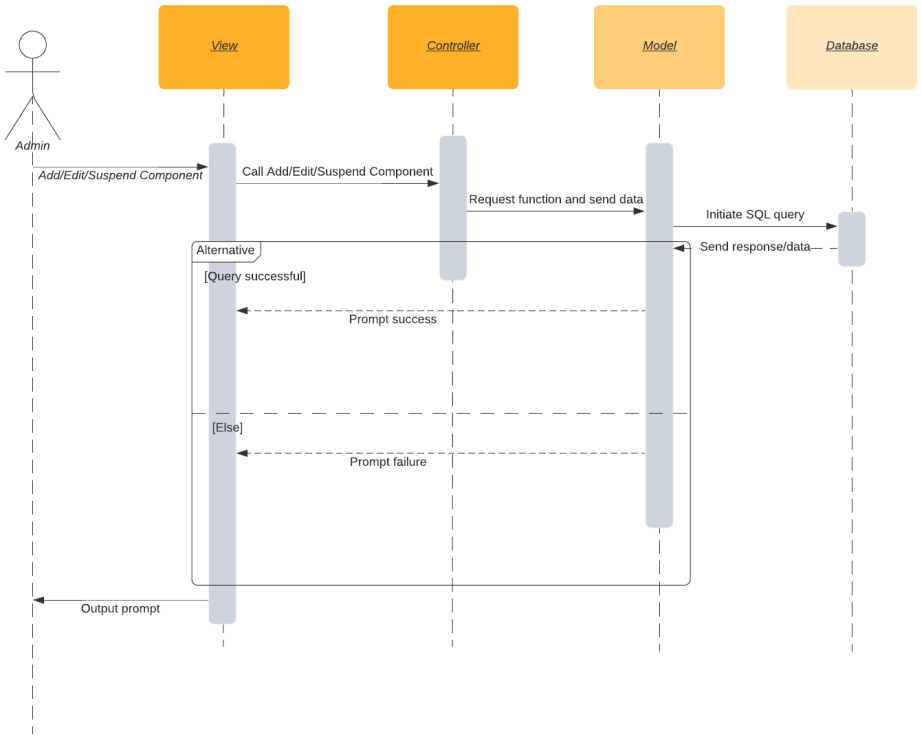

Figure 14: Admin functions

4.6.2 User login

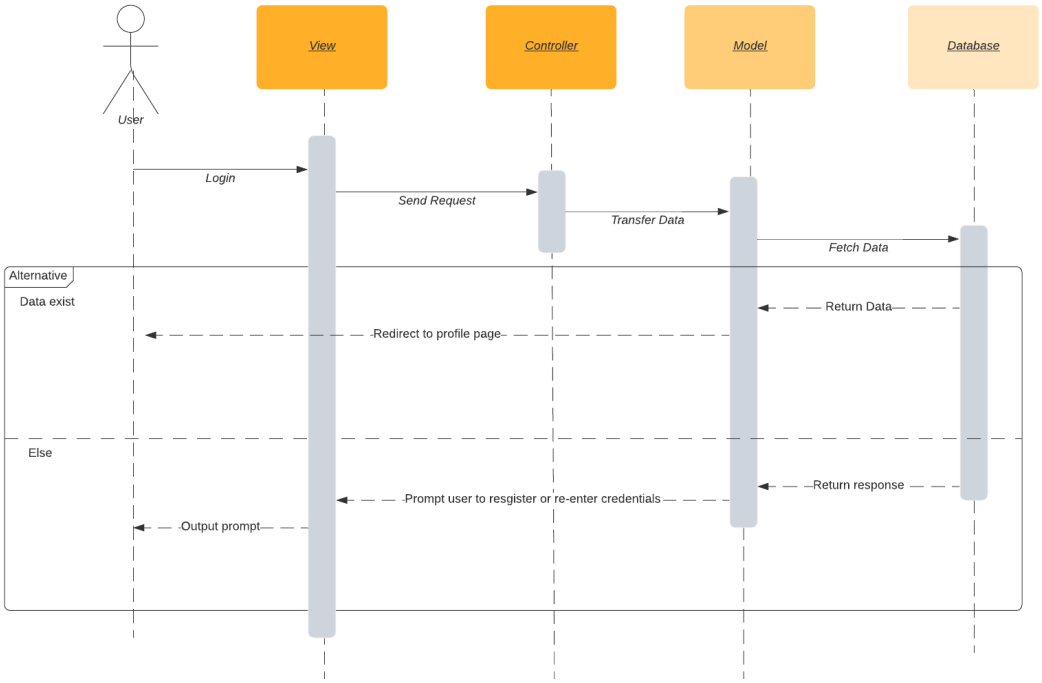

Figure 15: User login

4.6.3 User booking

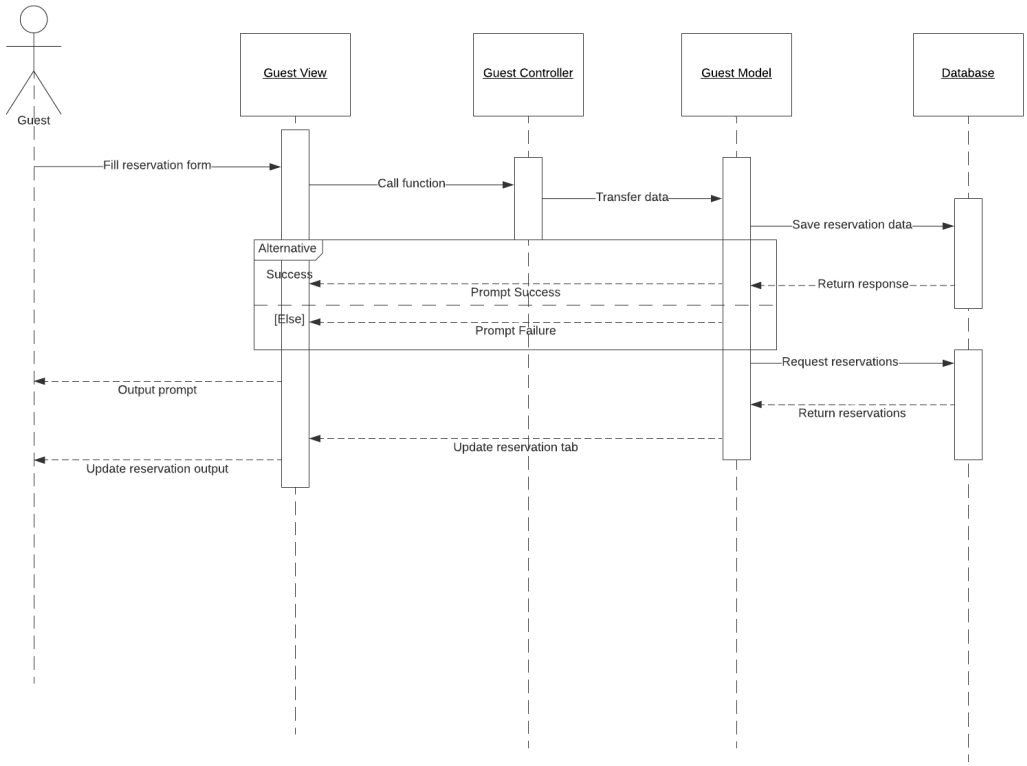

Figure 16: Guest booking a hotel/package

4.6.4 User - Support interaction

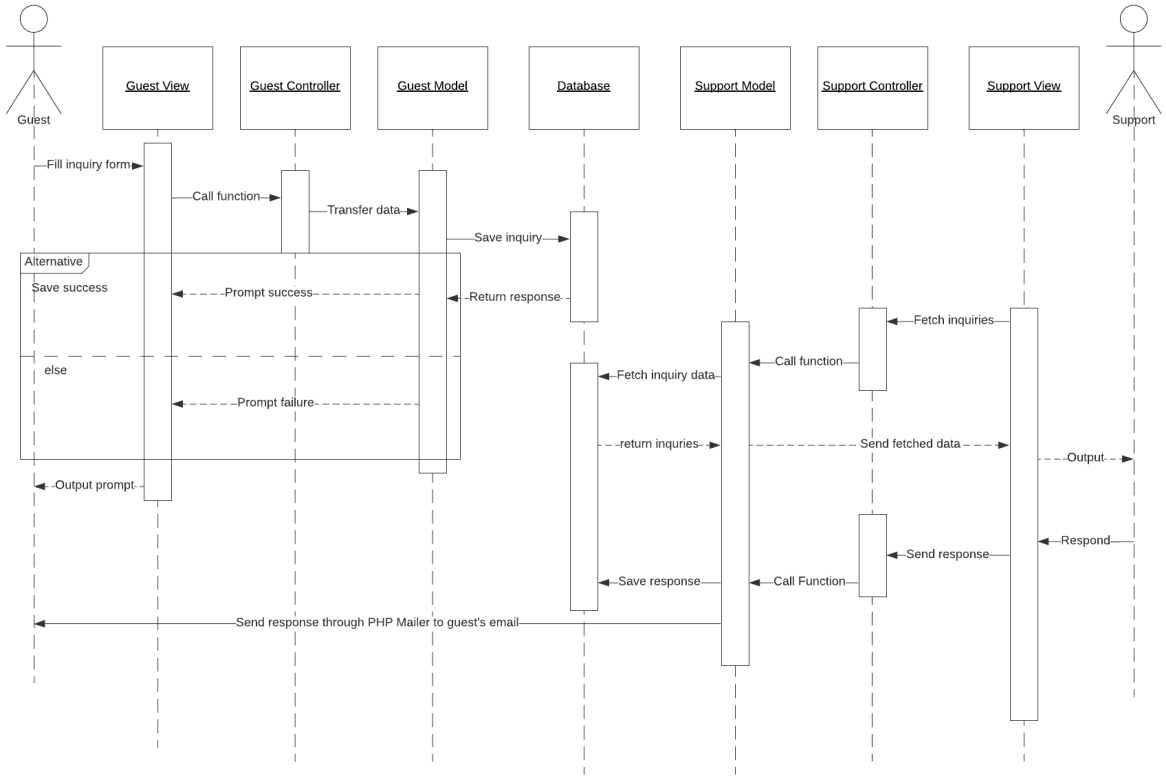

Figure 17: User and support inquiry interaction

4.7 Interface viewpoint

4.7.1 Admin interface

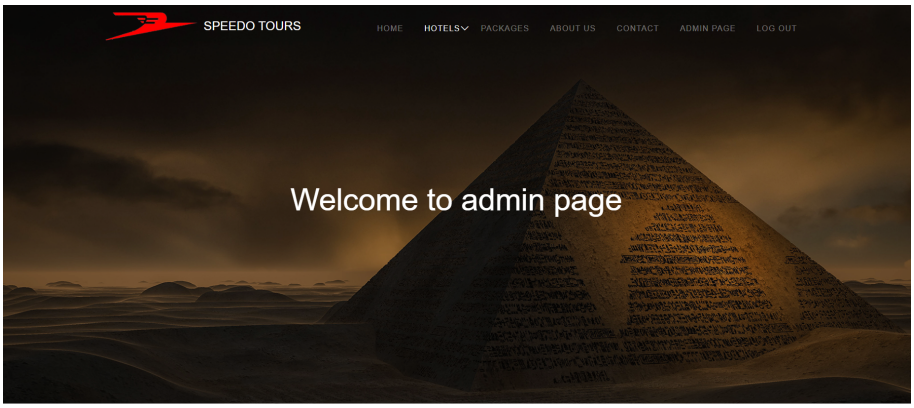

Figure 18: Admin interface part 1

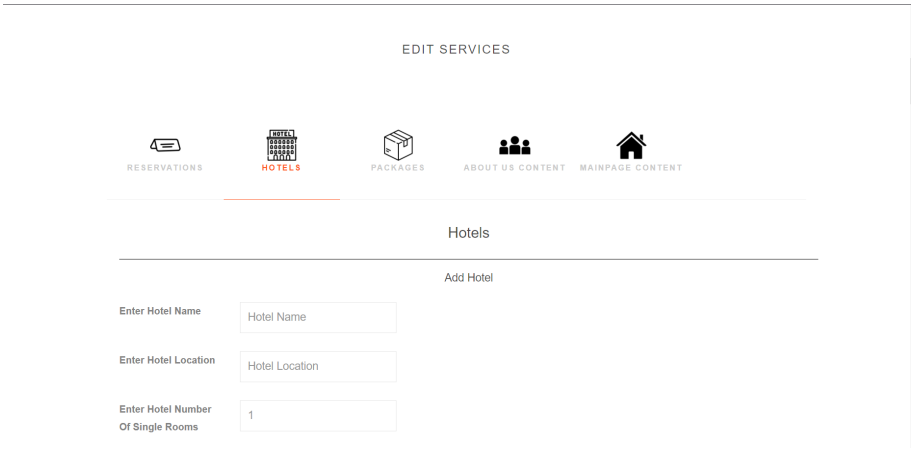

Figure 19: Admin interface part 2

The admin interface has several sections. Every section is responsible for a different component in the web application. The reservations tab handles the reservations made by guests and lets the admin confirm that the agency have proceeded with the reservation with the desired hotel. Hotel tab lets the admin control every hotel that exists in the database, furthermore the admin can add hotels to the database and also suspend certain hotels when needed (Suspension is to disable interaction with the component and not deleting it from the database.). The about us tab lets the admin update the agency's contact information in case of any changes and also add posts to the blog. Lastly, main page tab lets the admin control what is viewed in the main page of the web application like featured hotels, best offers and customers feedback.

4.7.2 Support Center interface

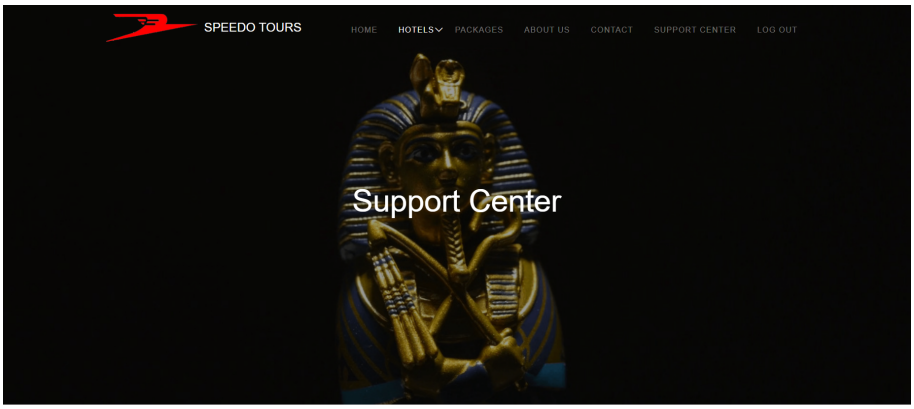

Figure 20: Support Center interface part 1

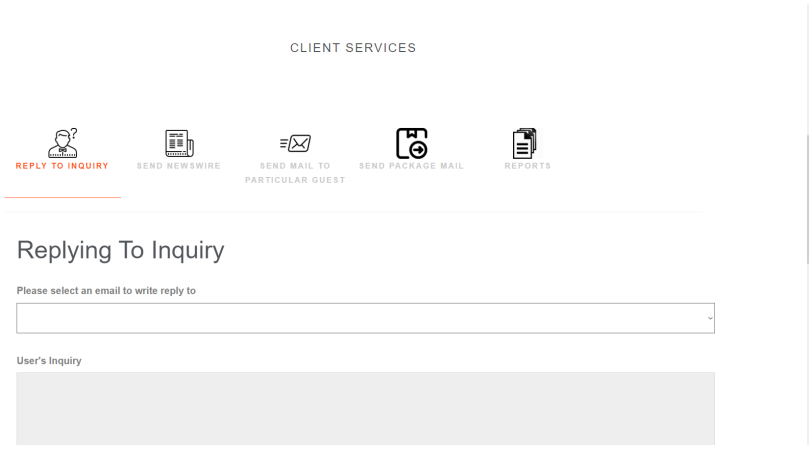

Figure 21: Support Center interface part 2

The first tab in the support center interface lets the support operator reply to the inquiries sent by users, Users here means that anyone who uses the web application can send an inquiry and doesn't have to be a registered user. Next tab is to send the news wire to the subscribed users. Third tab lets the support operator send an email to a particular guest. Fourth tab lets support operator send updates regarding a package to the guests who reserved this package. Last tab lets support operator view the reports regarding sent inquiries and news wire notifications.

5 Data Design

5.1 Data Description

The original format of the data was paper based. So, the optimal format to capture the data into the system was by using web page forms in order to save the data in the database. The database is expected to be large in terms of guests,reservations, hotel rooms, inquiries, reviews and news wire. Of course the numbers are governed by the market. The expected number of customers is by the hundreds and may reach thousands, again the numbers are governed by the market. The entities IDs are simple numbers that start from 1. All transactions bear time stamps that are saved in the database in the format (YYYY-MM-DD).

5.2 Database design description

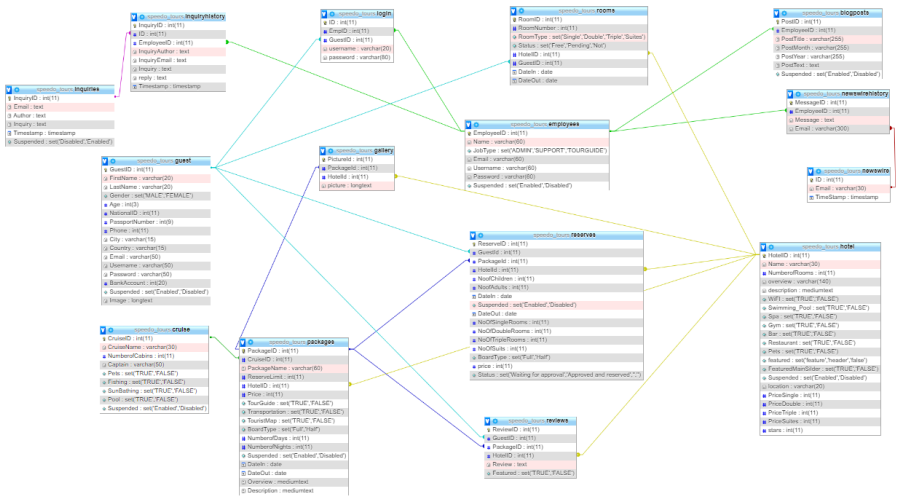

Figure 22: Database Diagram

6 Human Interface Design

6.1 User Interface

The guest user shall be able to view all hotels currently has highest offers on the main page and new hotels depending on what the admins desires to view on the main page.The user shall be view all hotels that can be booked in hotels page when the users chooses a page the data shall be retrieved and shown in the page to the user,the user shall be able to book a hotel from the booking section in the hotel.The user shall be able to view all packages in the packages page and upon clicking one of them he shall be able to view the package details and services and shall be able to book a package from the booking tab in the page.The user shall be to write an inquiry which is sent to the support operator and then receives a reply from the support operator on the user's email.

6.2 Screen Images

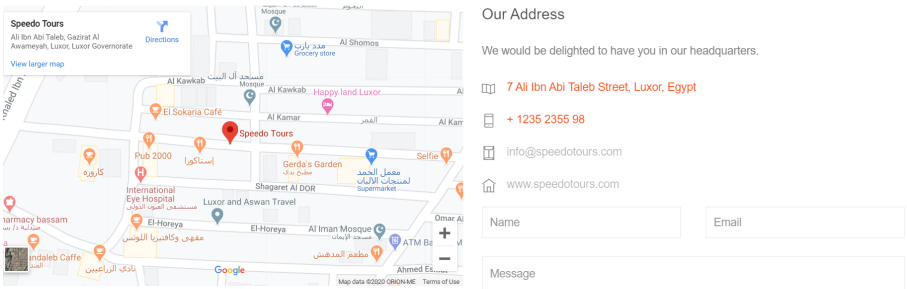

Figure 23: About Us page

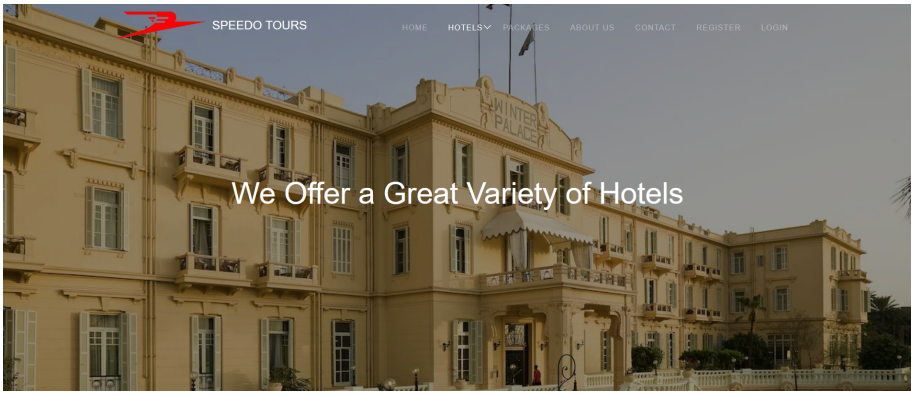

Figure 24: Hotels part 1

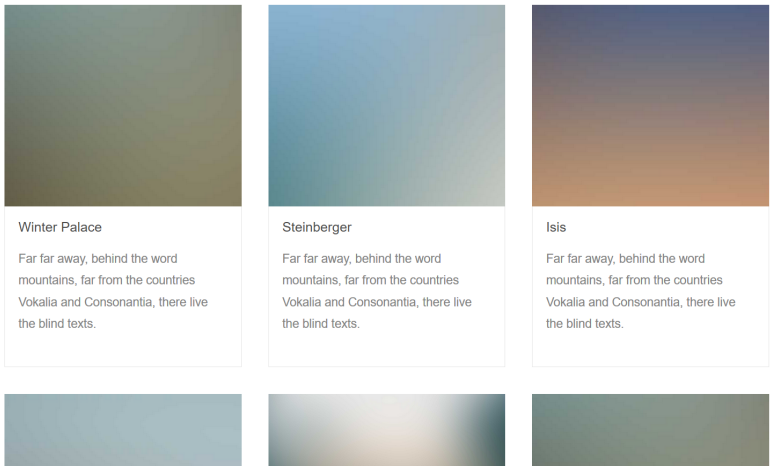

Figure 25: Hotels part 2

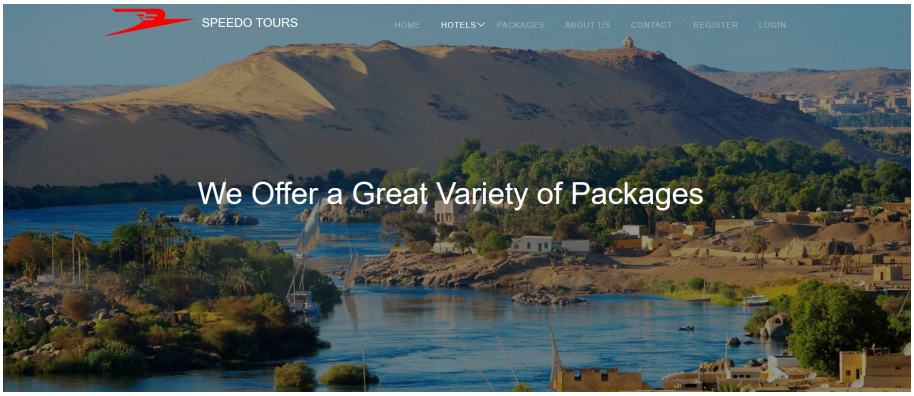

Figure 26: Packages part 1

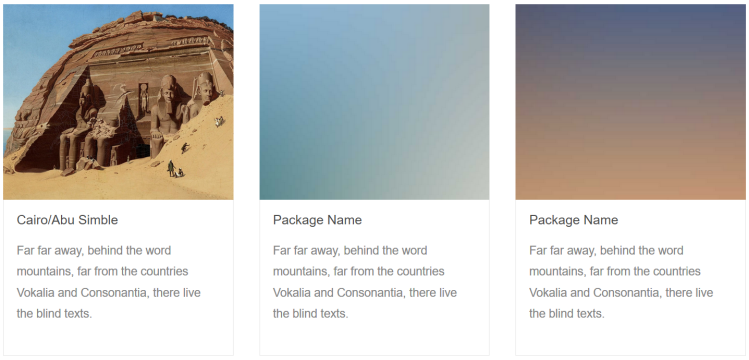

Figure 27: Packages part 2

Description

The Sofitel Winter Palace Hotel, also known as the Old Winter Palace Hotel, is a historic British colonial-era 5-star luxury resort hotel located on the banks of the River Nile in Luxor, Egypt, just south of Luxor Temple, with 86 rooms and 6 suites.

The hotel was built by the Upper Egypt Hotels Co, an enterprise founded in 1905 by Cairo hoteliers Charles Baehler and George Nungovich in collaboration with Thomas Cook & Son (Egypt). It was inaugurated on Saturday 19 January 1907, with a picnic at the Valley of the Kings followed by dinner at the hotel and speeches.[1] The architect was Leon Stienon, the Italian construction company G.GAROZZO & Figli Costruzioni in Cemento Armato, Sistema SIACCI brevettato. During World War I the hotel was temporarily closed to paying guests and employed as a hospice for convalescing soldiers. A regular guest at the hotel from 1907 on was George Herbert, 5th Earl of Carnarvon, better known simply as Lord Carnarvon. Carnarvon was the patron of Egyptologist Howard Carter, who in 1922 discovered the intact tomb of Tutankhamun. After the discovery was announced the Winter Palace played host to the international press corps and foreign visitors there to follow the story. Carter used the hotel's noticeboard to deliver occasional news and information on the discovery. In 1975 the complex was expanded with the construction of the New Winter Palace. The addition, classified as a 3-star hotel, was joined by corridors to the original. It was demolished in 2008. In 1996, the Pavillon, a 4-star annex with 116 rooms, was built in the rear garden of the Winter Palace, close to the swimming pool. The Pavillon shares many amenities with the Winter Palace, including the gardens, pools, tennis courts, terraces and restaurants. The hotel is owned by the Egyptian General Company

Figure 28: Single Hotel part 1

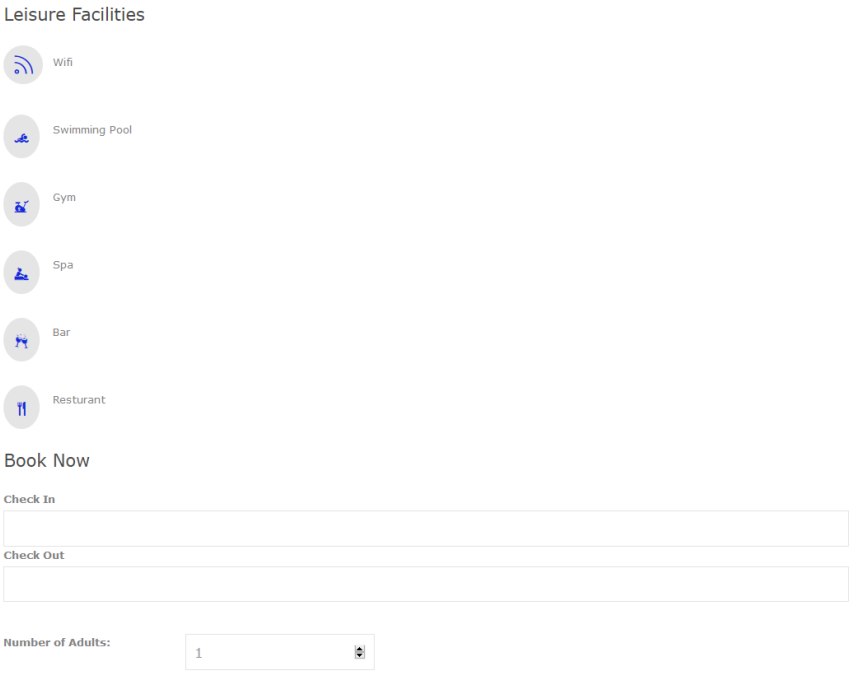

Figure 29: Single Hotel part 2

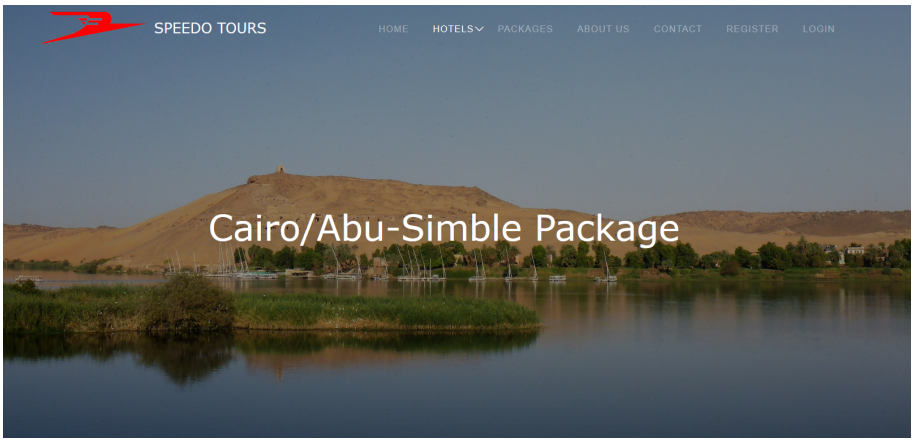

Figure 30: Single Package part 1

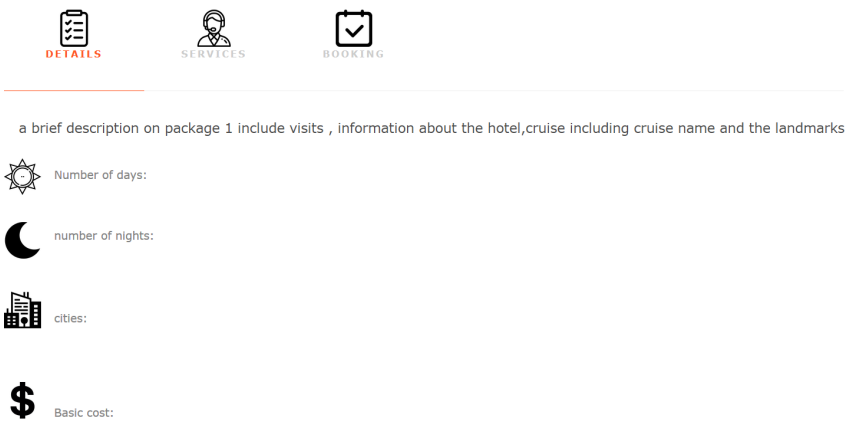

Figure 31: Single Package part 2

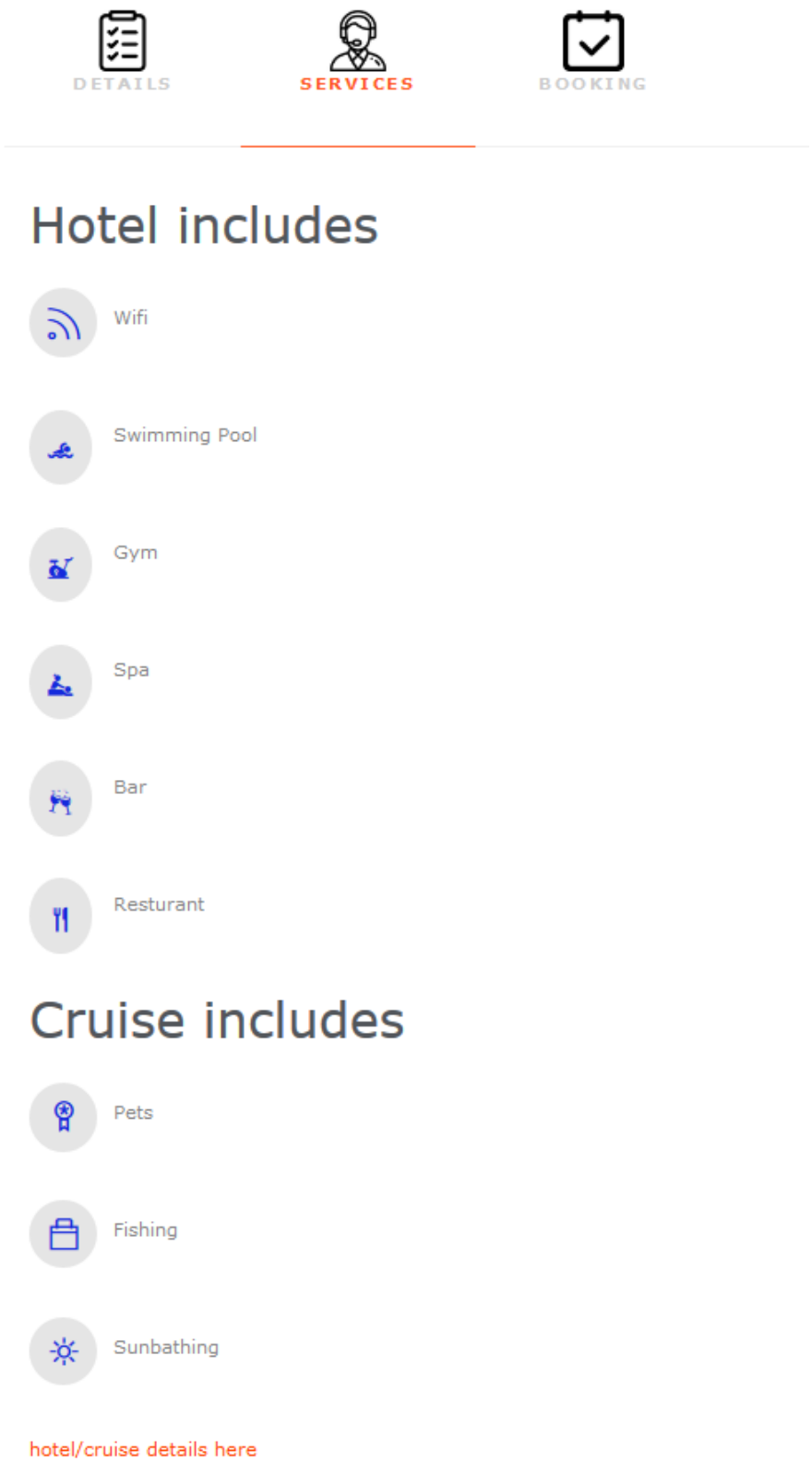

Figure 32: Single Package part 3

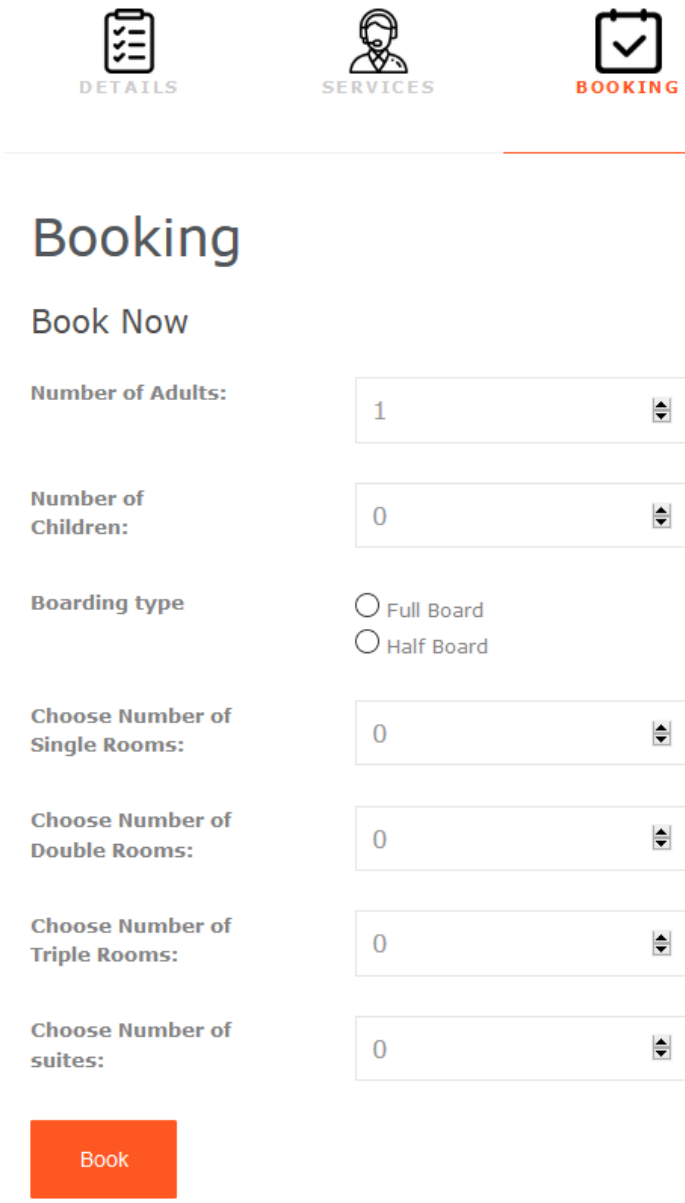

Figure 33: Single Package part 4

7 Requirements Matrix

Functional Requirement	Function Code	Finished
Guests shall login or register to the system. Also sign in with google option is available.	FR2,FR3	YES
Guests shall edit their profile from their perspective profile pages.	FR4	YES
Guests shall book any hotel/package	FR7,FR8	YES
Guests shall cancel desired reservation.	FR5,FR6	YES
User and Guest should send an enquiry.	FR10	YES
Support Operator shall reply to messages and enquiries from guests and users.	FR14	YES
Support Operator shall send news wire to subscribed emails on the system.	FR15	YES
Support Operator shall send emails to particular guests.	F17	YES
Admin shall edit every page's content	From FR18 To FR28	YES
Admin shall confirm books requested by guests.	FR30	YES
Admin shall add new pages to the website by adding hotels or packages.	FR18,FR24	YES
Admin can upload and edit gallery of images that are represented in web application's components	FR29	NO