Software Requirements Specification

Version 2

Touristor

Team 5

Member Name	Member Roll #	Primary Responsibility
Ayishm Aziz	15L-4140	Managing Product and Business Users
Mirah Saqib	15L-4080	API handling and integration
Midhat Asif	15L-4282	Front-end design and implementation
Eman Ijaz	15L-4267	Implementing Booking System
Tehreem Talat	15L-4143	Database formulation and Integration
Hafsa Batool	15L-4127	Back-end implementation

Table of Contents

Revision History

Name(s)	Date	Reason(s) For Change(s)	Version
Appendix C	20-11-18	Few mistakes and updation.	2

1. Introduction

1.1 Product

Touristor is an online travelling agent website which lets you lookup a destination for your holidays or trip across countries and cities. It also facilitates the search for hotels, restaurants, attractions and flights for the desired destination as well as provides the means for booking them online from home. You can view ratings and also give ratings, view rates and costs and make a payment online. You can design an online trip schedule to facilitate your tour.

1.2 Scope

This software is directed for business users who own a hotel, restaurant or flight agency. They can provide the online discovery of their business and booking or purchasing to extend their business. It will facilitate the business users as well as their customers, the travelers [4].
More specifically, this system will facilitate the communication and interaction between users and service providers (hotel, restaurant or flight agency owners) from across the world to aid both. Authenticated means of payment. Business users can manage the information on their products (hotels, restaurants or flights), edit, add or remove them. The client users can make reservations, bookings or give ratings to the products. The system contains a relational database containing a list of business users, client users, products and services, ratings and comments.

1.3 Business Goals

The Touristor is here to facilitate the users with fast, easy and reliable procedure for planning their trips. It also helps the hotel, restaurant and fight agency owner to advertise their services and enlarge their organization's scope. The goals are to provide the customer/user the finest environment to plan their trips, giving them the best possible choices and safe reliable booking procedure for their desired place. At the same time providing sponsored placements, business advantage, instant booking, ads for restaurants, awards and recognitions and promotion tools for the business holders or service providers. We will have a database server storing data of hundreds of major hotels, restaurants around the world as well as thousands of flight by various airline companies. Above all, we hope to provide a comfortable user experience along with the best pricing available.

1.4 Document Conventions

Following conventions have been used in this project.

For Main Headings
Format : Times
Size : 14
Face : Bold

For Explanation
Format : Calibri
Size : 12
Face : Normal

Business User	The user who can add, update and delete Hotel's or Restaurant's information.
Client User	The user who can search, book, give reviews about different places.
SRS	Software Requirements Specification.
Actors	Entities that provide input data or receive the output result.
DB	Database.
UC	Use Case.
Xampp	Server Name.
Postman	Tool for API developers.

More acronyms and technical jargons are annotated and included in the glossary.

1.5 References

[1] IEEE Software Engineering Standards Committee, "IEEE Recommended Practice for Software Requirements Specifications", October 20, 1998.
[2] Valuecoders - Hire Dedicated Software Development Team. "10 Top Web Development Frameworks,"July, 2017. Internet: https://www.valuecoders.com/blog/technology-and-apps/10-top-web-development-frameworks-businesses/
[3] Sass-lang. "Sass: Syntactically Awesome Style Sheets." Internet: https://sass-lang.com/
[4] Xintian-Cai. "Usability Analysis of TripAdvisor," August, 2013. Available:https://prezi.com/3tynat13npyf/usability-analysis-of-tripadvisor/
[5] Creative-Bloq. Rupert G., Tony C., Tim P. "9 security tips to protect your website from hackers," May,2018. Available: https://www.creativebloq.com/web-design/website-security-tips-protect-your-site-7122853
[6] Laravel. Taylor O. "Database: Getting Started - Laravel - The PHP Framework For Web Artisans."Avaialble: https://laravel.com/docs/5.7/database
[7] Sarah G. "Software Requirements Specification – Amazing Lunch Indiactor", April, 2011.

2. Overall Description

2.1 Product Features

Touristor will provide searching functionality for all the places to visit around the world, the hotels and restaurants on your desired location and flight schedules to your destination. The website will also enable the user to book flights and rooms according to their needs and other specifications. The user can choose from different options of hotels or restaurants and find about their services, prices and reviews using the website [1].

2.2 User Classes and Characteristics

The two main user classes include: Business Users and Client Users

2.2.1 Business Users

The term *business users* are opted for the users who own or manage any restaurant, hotel or flight agency. Business user is the one who has registered to this website and has added his/her products (hotels, restaurant or flight agency services). They can add, remove and update/edit their products and services. They will have a special profile with access to an interface designed for business users.

2.2.2 Client Users

The term *client users* is opted for the users who have registered as a normal user on the website and avail the services like searching for destination places, restaurants, hotels and flights and the service for making a reservation from internet. Their interests include travelling for holidays, tours, business trips and others. This type of users can look up products, view the ratings and reviews on them, get contact and location information, give reviews and ratings, make reservations and bookings and make a travel plan. They will have a profile of a normal user/client user.

2.3 Operating Environment

The desired system is designed as an interactive web application that runs on browsers. The business and client user must register through providing authentication to avail the most of the provided services. Profiles for users are private so one cannot search or lookup other users registered on the site. Only the business user can control or manage the information on the products while only the client users can give reviews or comments on the product.

2.4 Design and Implementation Constraints

The latest web technologies will be used in the development process. AngularJS and Laravel will be used in front-end and back-end development respectively. Xampp, Sublime Text, Postman, Apache, Sass [3], MySQL, Website Speed Test, SourceTree, Chrome Developer tools and slack are the tools that will be used during the developmental phase [5] [2]. For designing, the MVC architecture will be used. The website will be compatible with all latest browsers on all operating systems e.g. Chrome, Firefox, Opera, etc. The Database will be managed using Laravel [6], where MySQL will be

used. The English language will be the default language of our website so we are limited to one language for now. The communication protocol HTTPS along with TCP will be used.
Our website will be safe from SQL Injection and XSS attacks. Using HTTPS will add an encryption layer of SSL which will handle the security matters. Validation will be done on both sides and passwords will be checked and teach user to enter strong passwords.
They could be memory limitations as the data from cities over the world will be quite large.
The parallel booking of many users from across the globe will be an issue as well.

2.5 Assumptions and Dependencies

It is assumed that all users know how to use a mouse and type with a keyboard and all of the can read and write. Similarly, it is assumed that the user will be familiar with internet browsers since it is a web based application [7]. Thus it is also assumed that a user will have a computer or laptop with a proper internet connection while using the website. It is also assumed that business users have only added or will add the hotels, restaurants that exist in real life and all the information is correct. There are no hotels, restaurants on the website which do not exist as traveler cannot search and visit locations that don’t exist.

3. Functional Requirements

Register User (Business User)

<table>
<tr><td><b>Identifier</b></td><td colspan="2">Register Business User</td></tr>
<tr><td><b>Purpose</b></td><td colspan="2">Register a Business User into the database</td></tr>
<tr><td><b>Priority</b></td><td colspan="2">High</td></tr>
<tr><td><b>Actors</b></td><td colspan="2">Business User</td></tr>
<tr><td><b>Pre-conditions</b></td><td colspan="2">Must have a valid email ID and phone number</td></tr>
<tr><td><b>Post-conditions</b></td><td colspan="2">None</td></tr>
<tr><th colspan="3">Typical Course of Action</th></tr>
<tr><th>S#</th><th>Actor Action</th><th>System Response</th></tr>
<tr><td><b>1</b></td><td>Open the homepage of the website</td><td>HTML response – page loaded onto the screen.</td></tr>
<tr><td><b>2</b></td><td>Press the ‘Sign Up’ button</td><td>Signup panel opens and prompts the user to enter the following information: Email, Phone Number, CNIC, Full Name, Location Country, Location City, and Password.</td></tr>
<tr><td><b>3</b></td><td>User enters the required information</td><td>Prompt with ‘Successful Registration’ notice and re-directs to homepage while keeping the user logged in.</td></tr>
<tr><th colspan="3">Alternate Course of Action</th></tr>
<tr><th>S#</th><th>Actor Action</th><th>System Response</th></tr>
<tr><td><b>2a</b></td><td>Press ‘Sign In’ button</td><td>Sign in panel opens and prompts with the required fields
Email
Password
And displays a link to re-direct to registration page
Not a registered user? Sign Up here!</td></tr>
<tr><td><b>2b</b></td><td>User presses on the ‘Sign Up’ option</td><td>Signup panel opens and prompts the user to enter the following information: Email, Phone Number, CNIC, Full Name, Location Country, Location City and Password.</td></tr>
<tr><td><b>2b.1</b></td><td>User enters the required information</td><td>Sign in panel opens and prompts with the required fields
Email</td></tr>
</table>

		Password

Table 1: UC-1

Register User (Client User)

Identifier	Register User(Traveler)	
Purpose	Register a normal user in database with their information	
Priority	High	
Actors	Customer/Traveler	
Pre-conditions	None	
Post-conditions	None	
Typical Course of Action		
S#	**Actor Action**	**System Response**
1	Open the Home Page of website	HTML response – page loaded onto the screen.
2	Press the "Sign Up" button	System prompts to ask if the user wants to sign up as a "Business user" or "traveler".
3	User selects "traveler" option.	System prompts the user to enter the following information: Name, Email, Phone, Password, Credit card information, Current location (country and city).
4	The user enters the required information and clicks register.	System fetches the information and adds into the database and prompts by saying "Registered successfully" and logs in the user and opens his main page.
Alternate Course of Action		
S#	**Actor Action**	**System Response**
4a	The user entered the incorrect information and clicks register.	The system validates the information and prompts the user that he has entered incorrect information.
4a.1	User enters the information again and clicks register.	System fetches the information and adds into the database and prompts by saying "Registered successfully" and logs in the user and opens his main page.
4b	The user missed a required field while entering information and clicks register.	System prompts the user "This field cannot be left empty!"
4b.1	User enters the correct information and clicks register.	System fetches the information and adds into the database and prompts by

		saying "Registered successfully" and logs in the user and opens his main page.
3a	The user accidently selects the business user type.	System opens the signup page for business user which has a label saying "Not a business user? Sign up as a traveler here!"
3a.1	User clicks on "Sign Up as traveler here!" link.	System prompts the user to enter the following information: Name, Email, Phone, Password, Credit card information, Current location (country and city).
3a.2	The user enters the required information and clicks register.	System fetches the information and adds into the database and prompts by saying "Registered successfully" and logs in the user and opens his main page.

Table 2: UC-2

Login User

Identifier	UC-Login User	
Purpose	Login a user trying to book.	
Priority	High	
Actors	Business User, Customer/Traveler	
Pre-conditions	User should have registered once before login.	
Post-conditions	None	
Typical Course of Action		
S#	**Actor Action**	**System Response**
1	User opens Home Page of website and clicks "Login" button.	System displays a popup with fields email and password.
2	User enters email and password.	System verifies the user and directs the user to the Home Page respective of what type of user it was (Business or Traveler).
Alternate Course of Action		
S#	**Actor Action**	**System Response**
2a	User enters incorrect email or password.	System prompts the user by an error message "Incorrect email or password!"
2a.1	User enters correct information.	System logs in user successfully.

2b	User left a field empty.	System prompts the user with an error message saying "You cannot leave this field empty!"
2b.1	User enters correct information.	System logs in user successfully.
2c	User enters incorrect email 3 times.	System sends an email notification to the user's email informing about the account activity.

Table 3: UC-3

Add Hotel

Identifier	UC-Add Hotel	
Purpose	Enable business users to add Hotels on Touristor	
Priority	Medium	
Actors	Business User	
Pre-conditions	The business user should be logged in.	
Post-conditions	None	
Typical Course of Action		
S#	**Actor Action**	**System Response**
1	User selects the "Add Hotel" option from the menu.	The system opens a page where the user is asked to enter the information about the Hotel, like: Name, Location (Country and city), Phone, Rooms, etc.
2	User enters the information in all the fields and clicks "Add" button.	System verifies the information and adds the Hotel information in the database and prompts the user "Hotel added successfully".
Alternate Course of Action		
S#	**Actor Action**	**System Response**
2a	User enters the information in fields but leaves one or more required fields empty.	System prompts the user with an error message saying "You cannot leave this field empty!"
2a.1	User enters all correct information and clicks "Add" button.	System verifies the information and adds the Hotel information in the database and prompts the user "Hotel added successfully".
2b	User enters the incorrect information in one or more required fields and clicks "Add" button.	System verifies the information and prompts the user with an error message saying "You entered wrong information!"

2b.1	User enters the correct information in all the fields and clicks "Add" button.	System verifies the information and adds the Hotel information in the database and prompts the user "Hotel added successfully".

Table 4: UC-4

Add Restaurant

Identifier	UC-Add Restaurant		
Purpose	Enable business users to add Restaurants on Touristor		
Priority	Medium		
Actors	Business User		
Pre-conditions	The business user should be logged in.		
Post-conditions	None		
Typical Course of Action			
S#	**Actor Action**	**System Response**	
1	User selects the "Add Restaurant" option from the menu.	The system opens a page where the user is asked to enter the information about the Restaurant, like: Name, Location (Country and city), Phone, Deals, menu, etc.	
2	User enters the information in all the fields and clicks "Add" button.	System verifies the information and adds the Restaurant information in the database and prompts the user "Restaurant added successfully".	
Alternate Course of Action			
S#	**Actor Action**	**System Response**	
2a	User enters the information in fields but leaves one or more required fields empty.	System prompts the user with an error message saying "You cannot leave this field empty!"	
2a.1	User enters all correct information and clicks "Add" button.	System verifies the information and adds the Restaurant information in the database and prompts the user "Restaurant added successfully".	
2b	User enters the incorrect information in one or more required fields and clicks "Add" button.	System verifies the information and prompts the user with an error message saying "You entered wrong information!"	
2b.1	User enters the correct information in all	System verifies the information and	

	the fields and clicks “Add” button.	adds the Restaurant information in the database and prompts the user “Restaurant added successfully”.

Table 5: UC-5

Search Places, Hotels, Restaurants, Flights

Identifier	UC- Search Places, Hotels, Restaurants, Flights		
Purpose	User can search all the desired destinations to choose.		
Priority	Medium		
Actors	Customer/Traveler		
Pre-conditions	User should be on the page displaying Search bar.		
Post-conditions	None		
Typical Course of Action			
S#	**Actor Action**	**System Response**	
1	User selects a filter (such as hotel, restaurant, place or flight) on the search bar.	System then displays a field for location in the search bar which the user can fill.	
2	User enters his/her desired destination location (Country, City).	System then displays a timeline field for user’s arrival and departure dates (must for flight searches only, optional for hotel).	
3	User enters the expected date of arrival and date of departure (in case of flight or hotel) and click “Search”.	System matches the query in database according to the given specifications and retrieves all relevant results and displays to the user.	
4	User clicks on the preferred result.	System retrieves all information regarding the place or hotel or restaurant or flight selected by the user and displays on screen.	
Alternate Course of Action			
S#	**Actor Action**	**System Response**	
1a	User selects the “place” filter in the search bar and clicks “Search”.	System retrieves all *places* matching the query and displays the results.	
1a.1	User clicks on the preferred result.	System retrieves all information regarding the place or hotel or restaurant or flight selected by the user and displays on screen.	
2a	User does not fill the location field and clicks “Search”.	System displays results according to only the filter (hotel, restaurant, place, flight).	
2a.1	User clicks on the preferred result.	System retrieves all information	

		regarding the place or hotel or restaurant or flight selected by the user and displays on screen.
3a	User clicks “Search” without adding dates in case of *flight* filter.	System prompts the user to fill the time field and does not display anything.
3a.1	User enters the expected date of arrival and date of departure and click “Search”.	System matches the query in database according to the given specifications and retrieves all relevant results and displays to the user.
3a.2	User clicks on the preferred result.	System retrieves all information regarding the flight selected by the user and displays on screen.
3b	User clicks “Search” without adding dates in case of *hotel* filter.	System displays results for all the hotels in the specified location.
3b.1	User clicks on the preferred result.	System retrieves all information regarding the hotel selected by the user and displays on screen.

Table 6: UC-6

Book Restaurant

Identifier	UC- Book Restaurant	
Purpose	User can book any desired restaurant.	
Priority	Medium	
Actors	Customer/Traveler	
Pre-conditions	User must be logged in and on the desired Restaurant’s information page.	
Post-conditions	…	
Typical Course of Action		
S#	**Actor Action**	**System Response**
1	User clicks on “Book Restaurant” button.	System displays panel with editable fields already containing the information like: name, phone number, email, CNIC number that were entered by the user during registration.
2	User can change any information he /she wants and clicks “Continue”.	System opens new page which displays fields to be filled by the user like: date, time, number of people and meal type (dinner, lunch, occasion etc.)
3	User enters all required information and	System displays “Booking Confirmed

	clicks on “Confirm Booking” button.	Successfully” on prompt, log record in database, sends user confirmation email and redirect user to main page.
Alternate Course of Action		
S#	**Actor Action**	**System Response**
2a	User accidentally missed to enter some required field.	System prompts with “You cannot leave this field empty!” message.
2a.1	User properly enters all required fields and clicks on “Confirm Booking” button.	System displays “Booking confirmed successfully” on prompt, log record in database, sends user confirmation email and redirect user to main page.
2b	User properly enters all required fields and clicks on “Confirm Booking” button.	System verifies information, displays “Booking unsuccessful due to non-availability!” on prompt.
2c	User can click “Cancel Booking” button.	System redirects user to Restaurant page.

Table 7: UC-7

Book Hotel

Identifier	UC- Book hotel	
Purpose	User can book any desired hotel.	
Priority	Medium	
Actors	Customer/Traveler	
Pre-conditions	User must be logged in and present on the Hotel information page.	
Post-conditions	None	
Typical Course of Action		
S#	**Actor Action**	**System Response**
1	User can click on “Book Hotel” button.	System displays panel with editable fields already containing the information like: name, phone number, email, CNIC number that were entered by the user during registration.
2	User can change any information he /she wants and clicks “Continue”.	System opens new page which displays fields to be filled by the user like: stay duration and number of accommodations.

3	User enters all required information and clicks on "Confirm Booking" button.	System displays "Booking confirmed successfully" prompt, log record in database, sends user confirmation email and redirect user to payment page.
Alternate Course of Action		
S#	**Actor Action**	**System Response**
2a	User accidentally missed to enter some required field.	System prompts with "You cannot leave this field empty!" message.
2a.1	User properly enters all required fields and clicks on "Confirm Booking" button.	System displays "Booking confirmed successfully" on prompt, log record in database, sends user confirmation email and redirect user to payment page.
2c	User properly enters all required fields and clicks on "Confirm Booking" button.	System processes information, displays "Booking unsuccessful due to non-availability of room" on prompt.
2d	User can click "Cancel Booking" button.	System redirects user to Hotel page.

Table 8: UC-8

Book Flight

Identifier	UC- book flight	
Purpose	User can book flight for travelling to desired destination.	
Priority	Medium	
Actors	Customer/Traveler	
Pre-conditions	User must be logged in.	
Post-conditions	…	
Typical Course of Action		
S#	**Actor Action**	**System Response**
1	User can click on "Book Flight" button from the menu.	System displays panel with editable fields already containing the information like: name, phone number, email, CNIC number that were entered by the user during registration.
2	User can change any information he /she wants and clicks "Continue".	System opens new page which displays panel and asks for following information: Desired flight, timings, source location, destination location, payment method

		and credit card information.
3	User enters all required information and clicks on "Confirm Flight" button.	System verifies information, logs record in database, displays prompt "Flight confirmed successfully" and redirect user to payment page. System asks following information: Enter amount, card number, check payment method.
3.	User enters all payment information and click "Proceed" button.	System validates information, log record in database, sends confirmation email and displays "Payment done successfully. Verify through email".
Alternate Course of Action		
S#	**Actor Action**	**System Response**
2a	User accidentally missed to enter required field.	System prompts with "You cannot leave this field empty!" message.
2a.1	User properly enters all required fields and clicks on "Confirm Flight" button.	System verifies information, logs record in database, displays "Flight confirmed successfully" on prompt and redirect user to payment page.
2b	User properly enters all required fields and clicks on "Confirm flight" button.	System verifies information and displays "Booking unsuccessful due to non-availability of flight" on prompt.
2c	User can click "Cancel Booking" button.	System redirects user to main page.
3a	User accidentally missed to enter required payment field.	System prompts with "You cannot leave this field empty!" message.
3a.1	User enters all payment information and click "Proceed" button.	System validates information, log record in database, sends confirmation email and displays" Payment done successfully. Verify through email".

Table 9: UC-9

Give a Review

Identifier	UC- Give a Review
Purpose	Enable customer(traveller) to enter reviews about any place, restaurant or hotel on Touristor

Priority	Medium	
Actors	Customer/Traveler	
Pre-conditions	The customer should be logged in.	
Post-conditions	…	
Typical Course of Action		
S#	**Actor Action**	**System Response**
1	User selects the “Enter a Review” option from the menu.	The website shows a dropdown sub-menu for the user to choose any one option from: • Review Places • Review Hotels • Review Restaurants
2	The user selects to review places or to review hotels or to review restaurants from the dropdown sub-menu.	The system opens a page displaying restaurants, hotels or places according to the user’s selected choice. Here, the user is required to choose any restaurant, hotel or a place. User can scroll or type the name of selected option in the customized search bar.
3	The user selects his desired place, restaurant or hotel from the web page to review.	System then opens the web page of the selected restaurant, hotel or place, with a review section below. Here the user is required to enter his/her name along with the review.
4	The user then enters his/her name in the review section along with the review in the review box; and then presses enter.	The system then adds the review information of that customer for that place, restaurant or hotel in the database and displays message to the user: “Thank you for giving your feedback!”

Alternate Course of Action		
S#	**Actor Action**	**System Response**
2a	The user had already booked a place, hotel or a restaurant for a specific time period.	After that time span is over, the system notifies the user: “How was your experience? Would you like to give a review for this place/hotel/restaurant?”
2a.1	The user ignores the notification.	System doesn’t do anything.
2b	User clicks on the notification.	System then opens the web page of the restaurant, hotel or place, in accordance with the prompted notification, with a review section below. Here the user is required to enter his/her name along with the review.
2b.1	The user then enters his/her name in the review section along with the review in the review box; and then presses enter.	The system then adds the review information of that customer for that place, restaurant or hotel in the database and displays message to the user: “Thank you for giving your feedback!”

Table 10: UC-10

Delete Hotel

Identifier	UC-Delete Hotel
Purpose	Enable business users to delete Hotels from Touristor
Priority	Medium
Actors	Business User
Pre-conditions	The business user should be logged in.
Post-conditions	None

Typical Course of Action		
S#	**Actor Action**	**System Response**
1	User selects the “Delete Hotel” option from the menu.	The system provides user with a list of all of his/her hotels with their names and delete buttons in front of each name.
2	User clicks “Delete” button in front of the hotel he wants to delete.	System deletes the Hotel from database and prompts the user “Hotel has been removed successfully”.

Table 11: UC-11

Delete Restaurant

Identifier	UC-Delete Restaurant
Purpose	Enable business users to delete Restaurants from Touristor
Priority	Medium
Actors	Business User
Pre-conditions	The business user should be logged in.
Post-conditions	None

Typical Course of Action		
S#	**Actor Action**	**System Response**
1	User selects the “Delete Restaurant” option from the menu.	The system provides user with a list of all of his/her restaurants with their names and delete buttons in front of each name.
2	User clicks “Delete” button in front of the restaurant he wants to delete.	System deletes the Restaurant from database and prompts the user “Hotel has been removed successfully”.

Table 12: UC-12

Update Hotel Features

Identifier	UC-Update Hotel Features
Purpose	Enable business users to edit Hotels on Touristor
Priority	Medium
Actors	Business User
Pre-conditions	The business user should be logged in.

Post-conditions	None	
Typical Course of Action		
S#	**Actor Action**	**System Response**
1	User selects the "Edit Hotel Info" option from the menu.	The system provides user with a list of all of his/her hotels.
2	User selects the hotel from the list and clicks "Edit" button.	The system opens a page with the specified hotel information and where the user can edit the information he wants about the Hotel by changing the existing info that is editable.
3	User edits the information in all the fields and clicks "Update" button.	System verifies the information and adds the Hotel information in the database and prompts the user "Hotel information edited successfully".
Alternate Course of Action		
S#	**Actor Action**	**System Response**
3b	User enters the incorrect information in one or more and clicks "Update" button.	System verifies the information and prompts the user with an error message saying "You entered wrong information!"
3b.1	User enters the correct information and clicks "Update" button.	System verifies the information and adds the Hotel information in the database and prompts the user "Hotel information edited successfully".

Table 13: UC-13

Update Restaurant Features

Identifier	UC-Update Restaurant Features	
Purpose	Enable business users to edit Restaurants on Touristor	
Priority	Medium	
Actors	Business User	
Pre-conditions	The business user should be logged in.	
Post-conditions	None	
Typical Course of Action		
S#	**Actor Action**	**System Response**
1	User selects the "Edit Restaurant Info" option from the menu.	The system provides user with a list of all of his/her Restaurants.
2	User selects the Restaurant from the list	The system opens a page with the

	and clicks “Edit” button.	specified Restaurant information and where the user can edit the information he wants about the Restaurant by changing the existing info that is editable.
3	User edits the information in all the fields and clicks “Update” button.	System verifies the information and adds the Restaurant information in the database and prompts the user “Restaurant information edited successfully”.
Alternate Course of Action		
S#	**Actor Action**	**System Response**
3b	User enters the incorrect information in one or more and clicks “Update” button.	System verifies the information and prompts the user with an error message saying “You entered wrong information!”
3b.1	User enters the correct information and clicks “Update” button.	System verifies the information and adds the Restaurant information in the database and prompts the user “Restaurant information edited successfully”.

Table 14: UC-14

4. Nonfunctional Requirements

4.1 Performance Requirements

- The search must give results within 2-3 seconds.
- Website must be able to handle booking of at least 1000 users per second.
- The database must be updated within 0.5-1 second in case of register, add, delete, etc.

4.2 Security Requirements

- Privacy and security is compromised if the website is vulnerable to SQL injection or XSS attacks. For secure website, it must be immune to such attacks. Website must implement SSL so that security requirements are met.

- Furthermore, user must be encouraged to use strong passwords.
- User must be authenticated before using the website.
- The data related to users must be kept confidential.

4.3 Usability Requirements

- Easy Navigation: Website provides clear text links, search tool, simple searches and sitemap to make navigation on the site easy for the users.
- Useful Content: Clear objective, organization of content, regularly updated, useful links which makes it easy for user to locate a search for a flight, hotel, car,
- Good Search Engines: utilizes Clear text with keywords, clear text links.
- Efficiency of use: goals are easy to accomplish quickly and with few or no user errors
- Intuitiveness: the interface is easy to learn and navigate; buttons, headings, and help/error messages are simple to understand
- Low perceived workload: the interface appears easy to use, rather than intimidating, demanding and frustrating

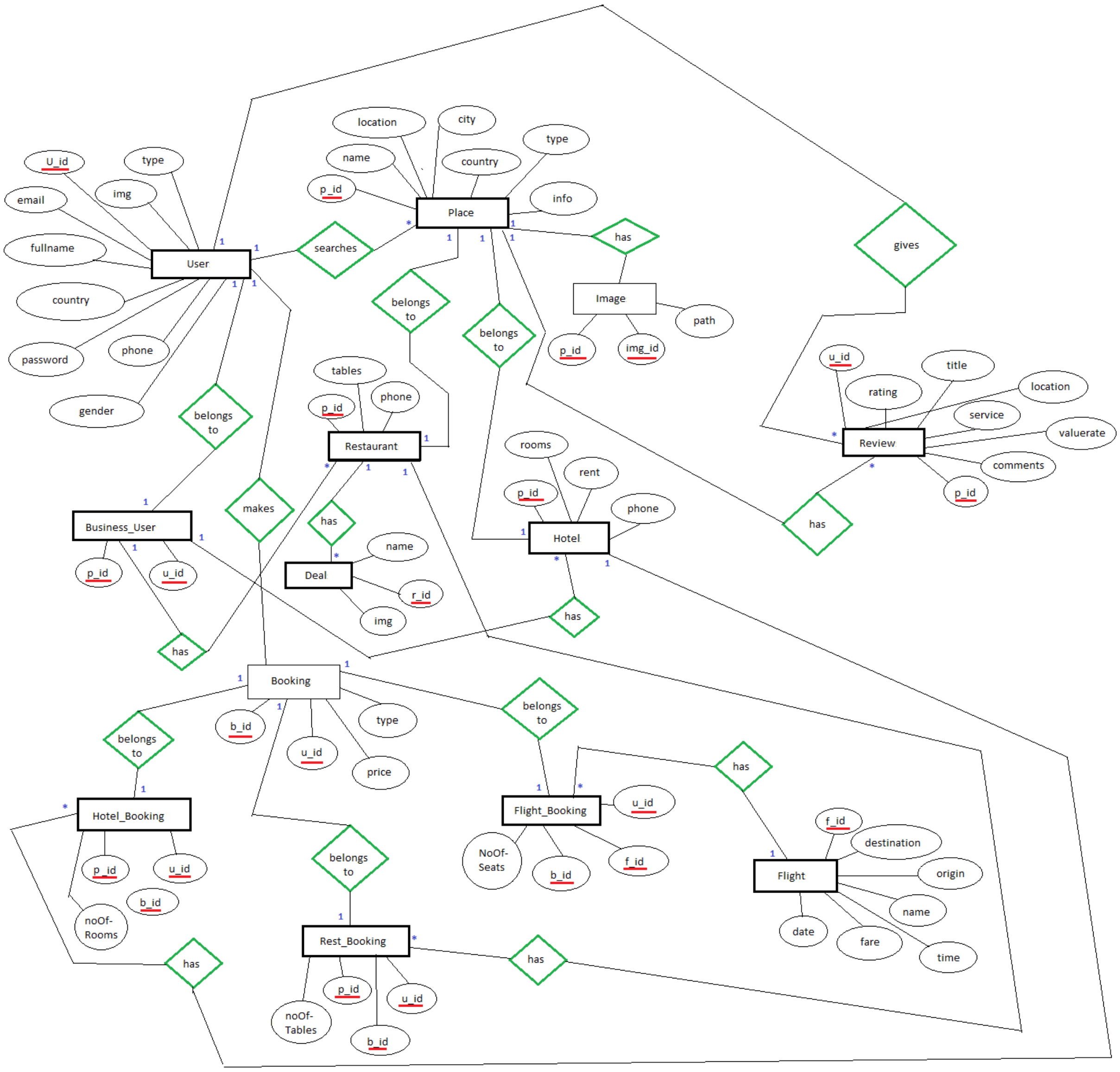
5. Other Requirements

The Touristor should already have integrated the API of different airline companies regarding their flight timetables. This would allow Tourists to dynamically update the flight schedules of different airlines for the customers to conveniently pick up a suitable flight. Also, this would free the admins of Touristor and the flight agents the hassle of manually updating the schedules.

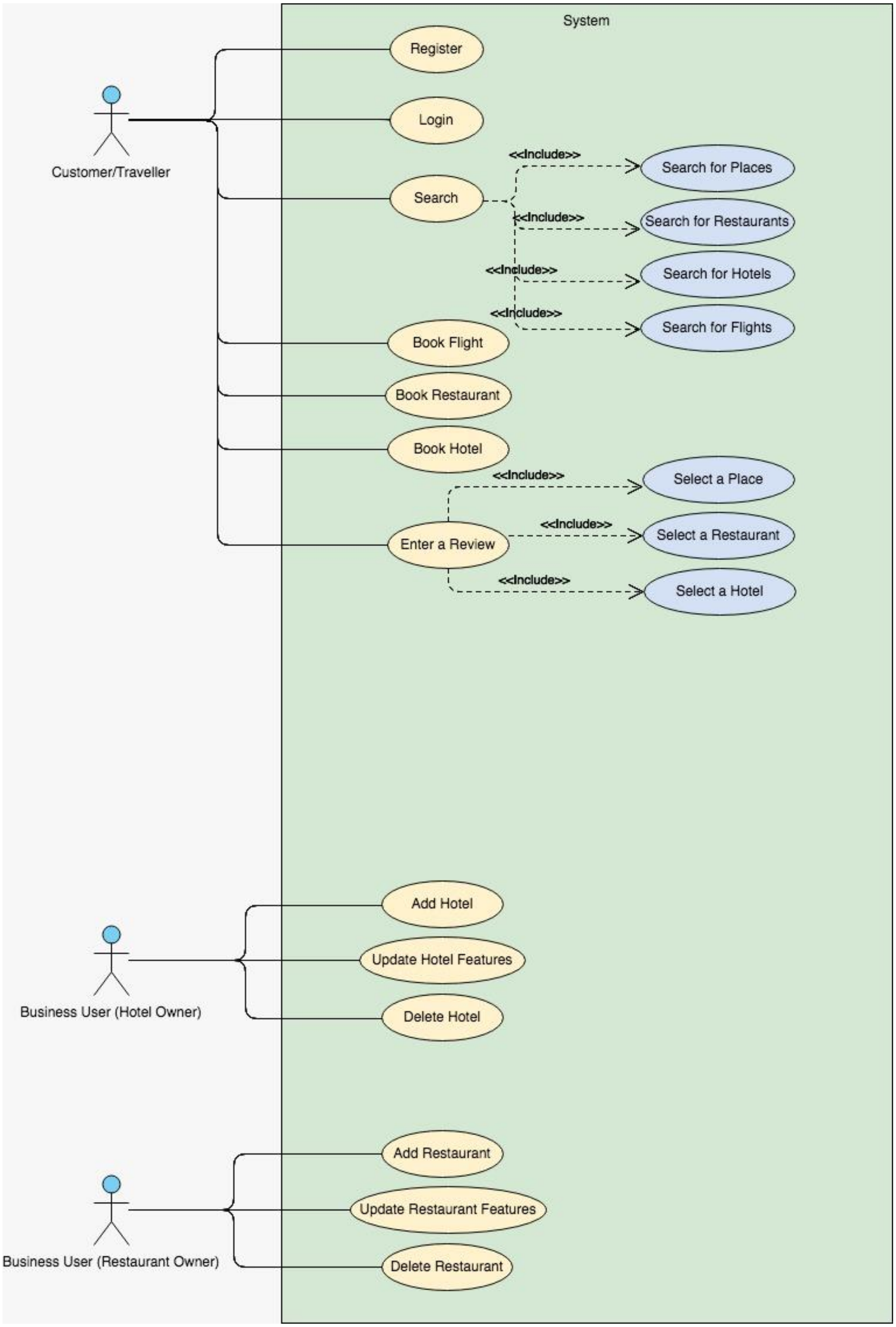
Appendix A: Glossary

Term	Definition
Xampp	Xampp is a free and open-source cross-platform web server solution package developed by Apache Friends, consisting mainly of the Apache HTTP server, MariaDB database and interpreters for scripts written in PHP and Perl programming languages.
Sublime Text	Sublime Text is a proprietary cross-platform source code editor with a Python application programming interface(API).
Postman	Postman is a tool chain for API developers to share, test, document and monitor APIs.
Apache	Apache is a free and open-source cross-platform web server.
Sass	Syntactically awesome style sheets (Sass) is a style sheet language.
MySQL	MySQL is an open-source relational database management system (RDBMS).
Website Speed Test	It is online Speed testing software for a website.
SourceTree	SourceTree is a free Git client for Windows and Mac.
Chrome Development Tools	Chrome Development Tools is a set of web developer tools built directly into Google chrome browser.
Slack	An online platform for sharing information with your team.
XSS	Cross-Site scripting attacks are a type of injection, in which malicious scripts are injected into otherwise benign and trusted websites.
SQL Injection	It is a code injection technique that might destroy your database.

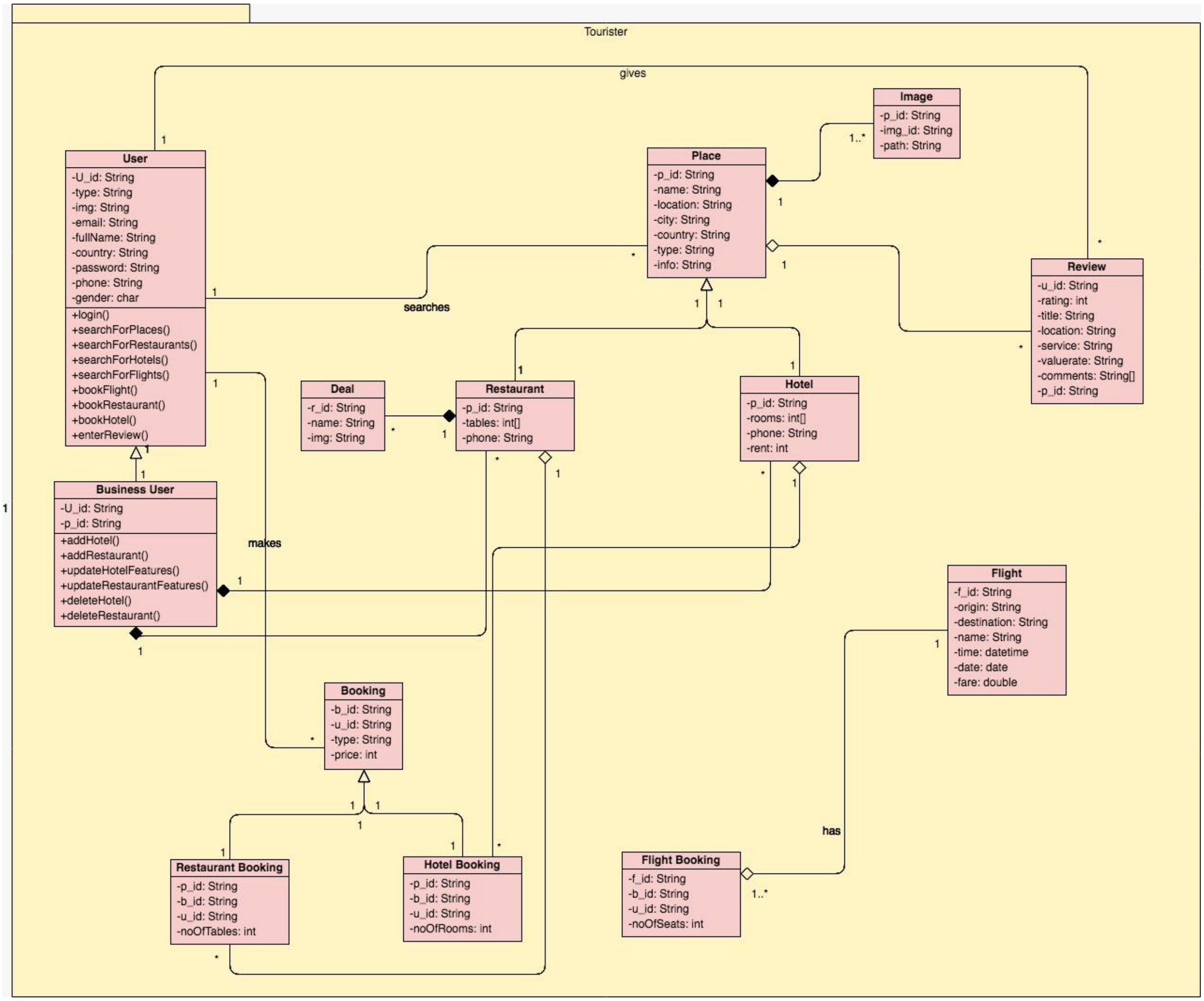
Appendix B: Analysis Models



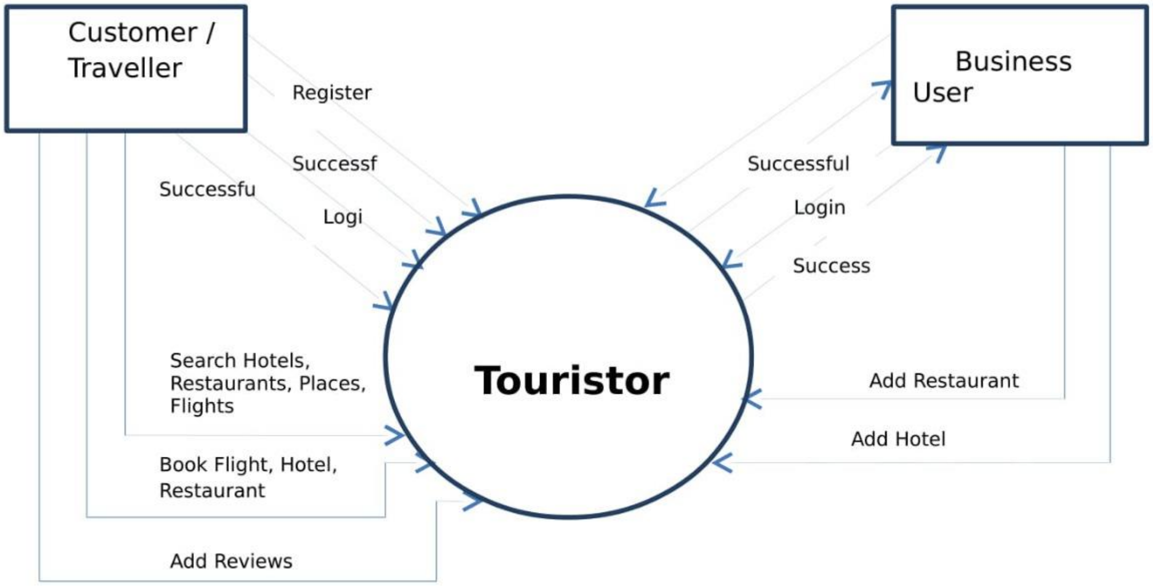
Entity Relationship Diagram



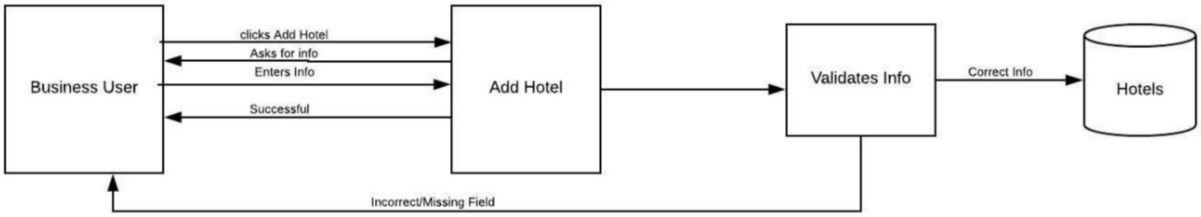
Use Case Diagram

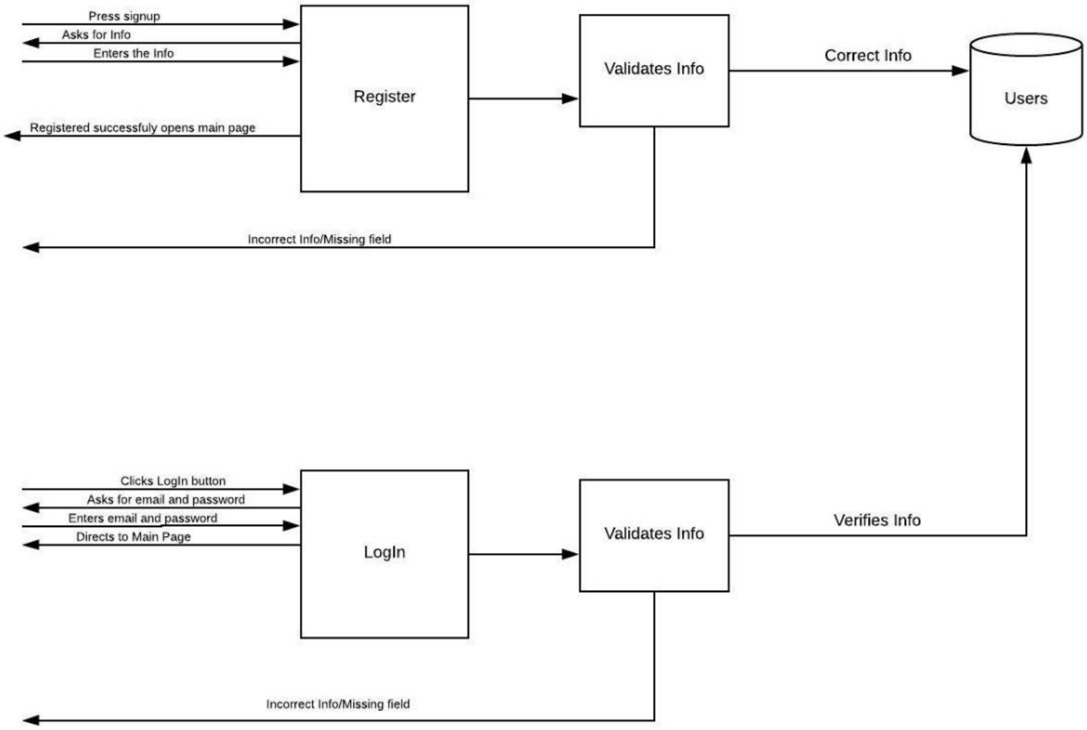


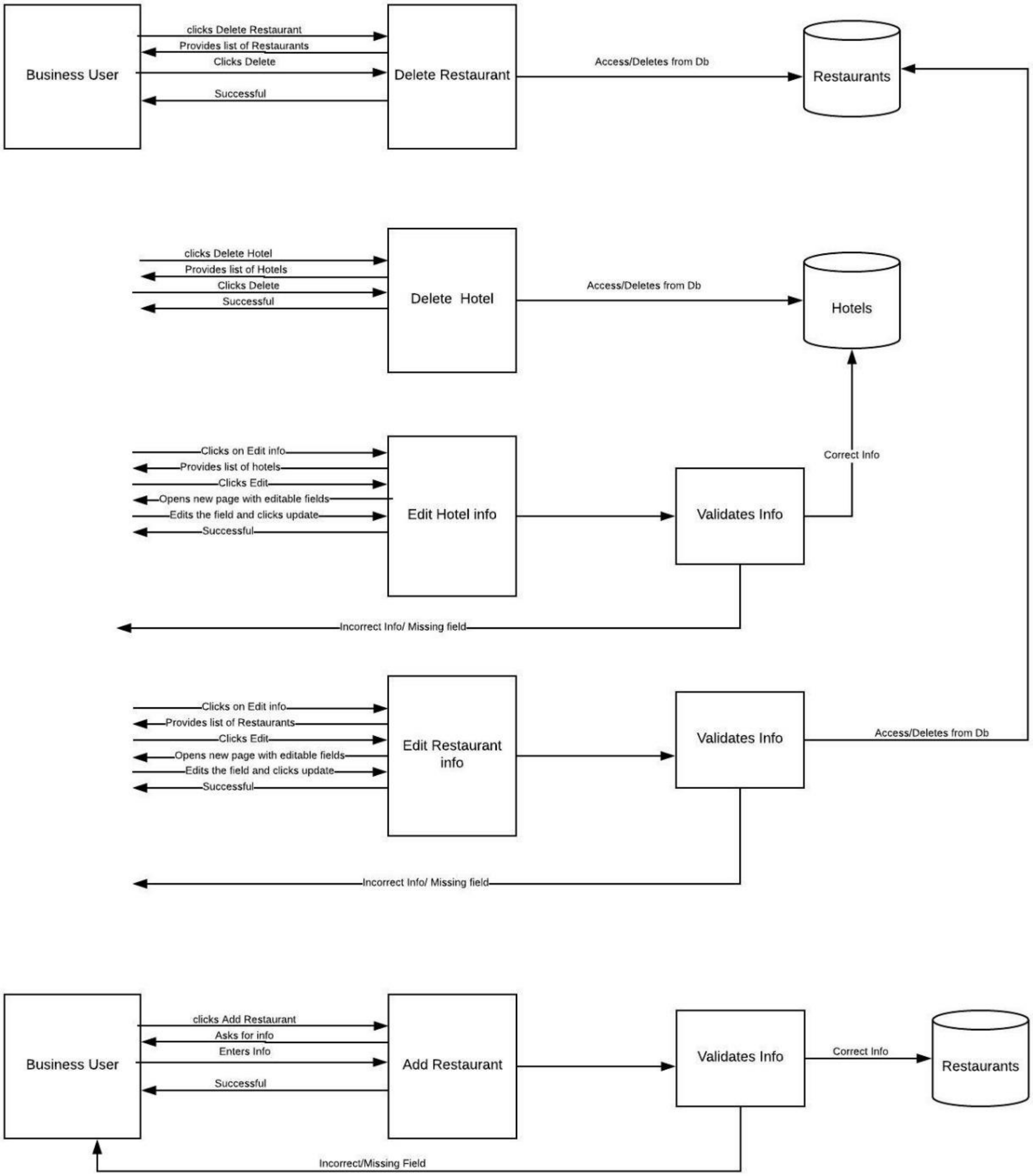
Class Diagram

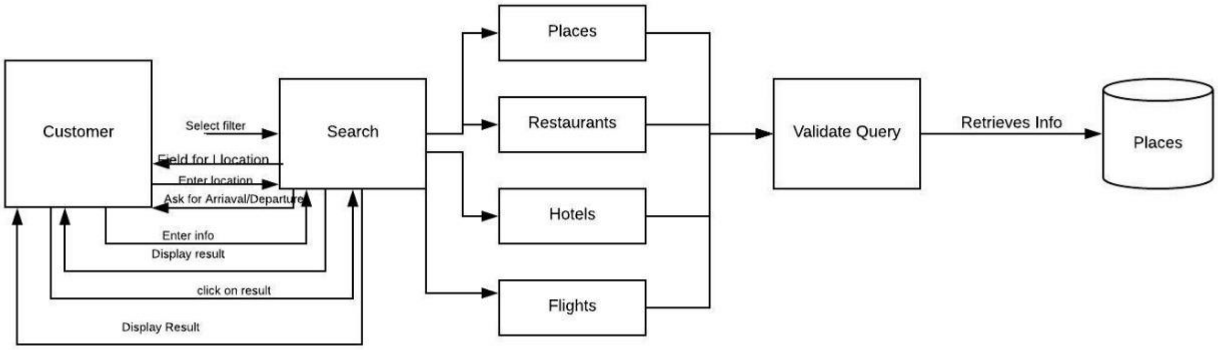


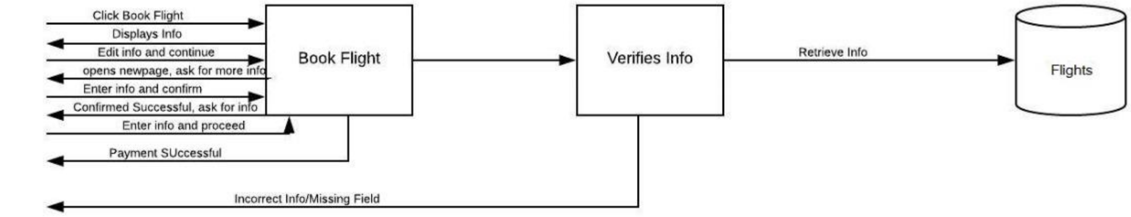
Contextual Level Data Flow Diagram

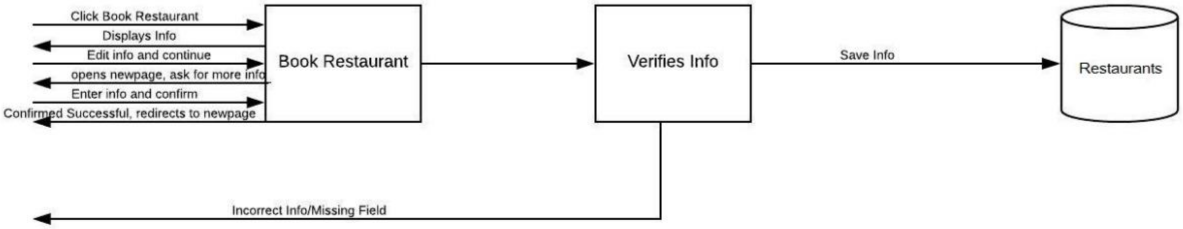

Business User
clicks Add Hotel
Asks for info
Enters Info
Successful
Add Hotel
Validates Info
Correct Info
Hotels
Incorrect/Missing Field

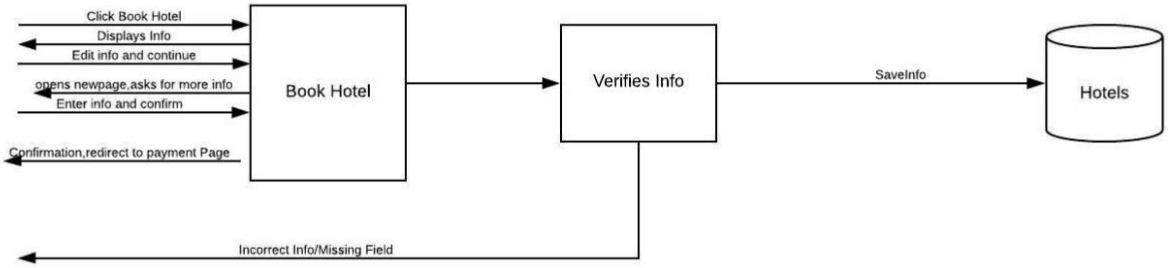

Press signup
Asks for Info
Enters the Info
Registered successfuly opens main page
Register
Validates Info
Correct Info
Users
Incorrect Info/Missing field
Clicks LogIn button
Asks for email and password
Enters email and password
Directs to Main Page
LogIn
Validates Info
Verifies Info
Incorrect Info/Missing field

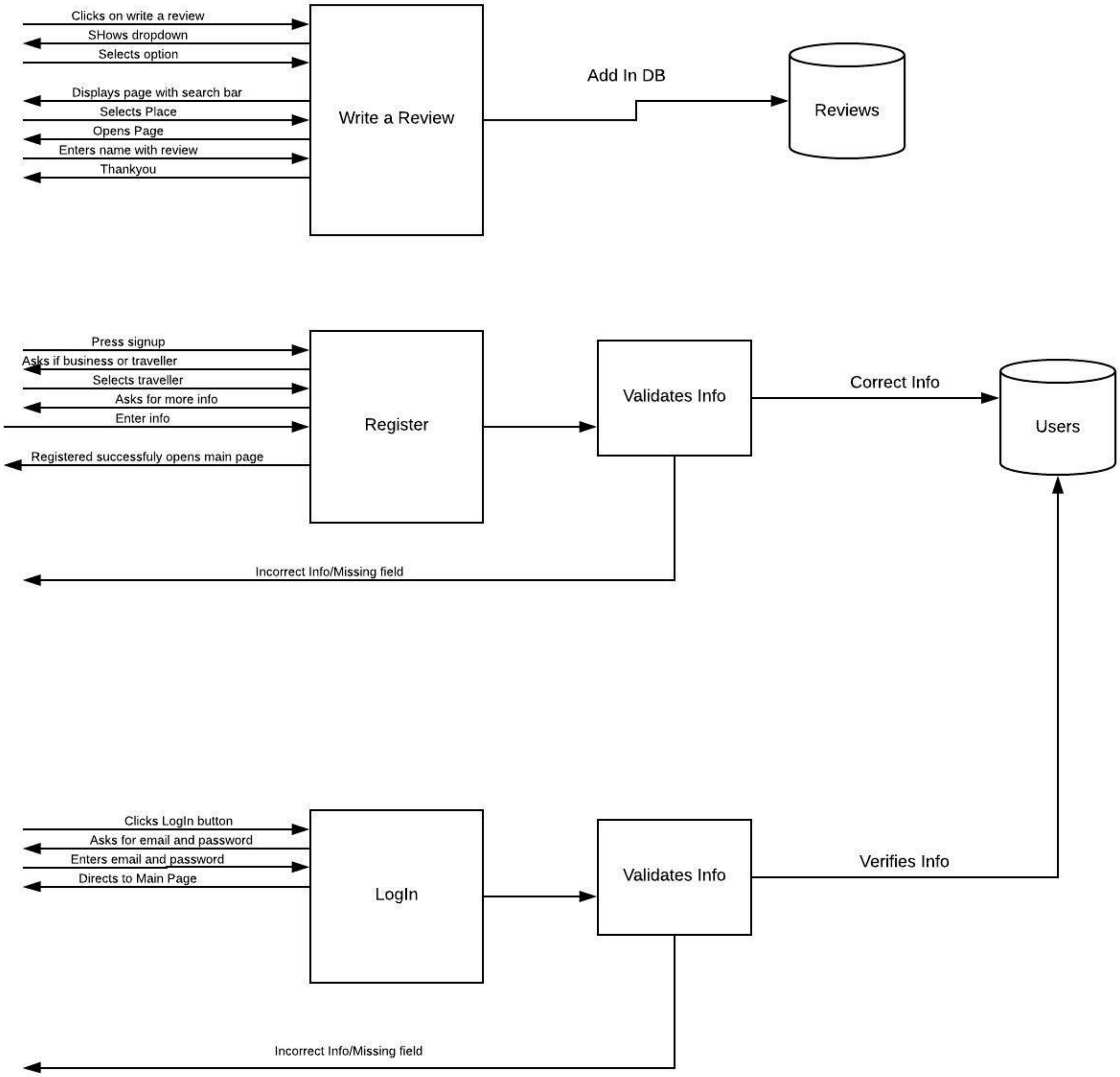

Business User
clicks Delete Restaurant
Provides list of Restaurants
Clicks Delete
Successful
Delete Restaurant
Access/Deletes from Db
Restaurants
clicks Delete Hotel
Provides list of Hotels
Clicks Delete
Successful
Delete Hotel
Access/Deletes from Db
Hotels
Clicks on Edit info
Provides list of hotels
Clicks Edit
Opens new page with editable fields
Edits the field and clicks update
Successful
Edit Hotel info
Validates Info
Correct Info
Incorrect Info/ Missing field
Clicks on Edit info
Provides list of Restaurants
Clicks Edit
Opens new page with editable fields
Edits the field and clicks update
Successful
Edit Restaurant info
Validates Info
Access/Deletes from Db
Incorrect Info/ Missing field
Business User
clicks Add Restaurant
Asks for info
Enters Info
Successful
Add Restaurant
Validates Info
Correct Info
Restaurants
Incorrect/Missing Field


Customer
Select filter
Field for Llocation
Enter location
Ask for Arriaval/Departure
Enter info
Display result
click on result
Display Result
Search
Places
Restaurants
Hotels
Flights
Validate Query
Retrieves Info
Places


Click Book Flight
Displays Info
Edit info and continue
opens newpage, ask for more info
Enter info and confirm
Confirmed Successful, ask for info
Enter info and proceed
Payment SUccessful
Book Flight
Verifies Info
Retrieve Info
Flights
Incorrect Info/Missing Field

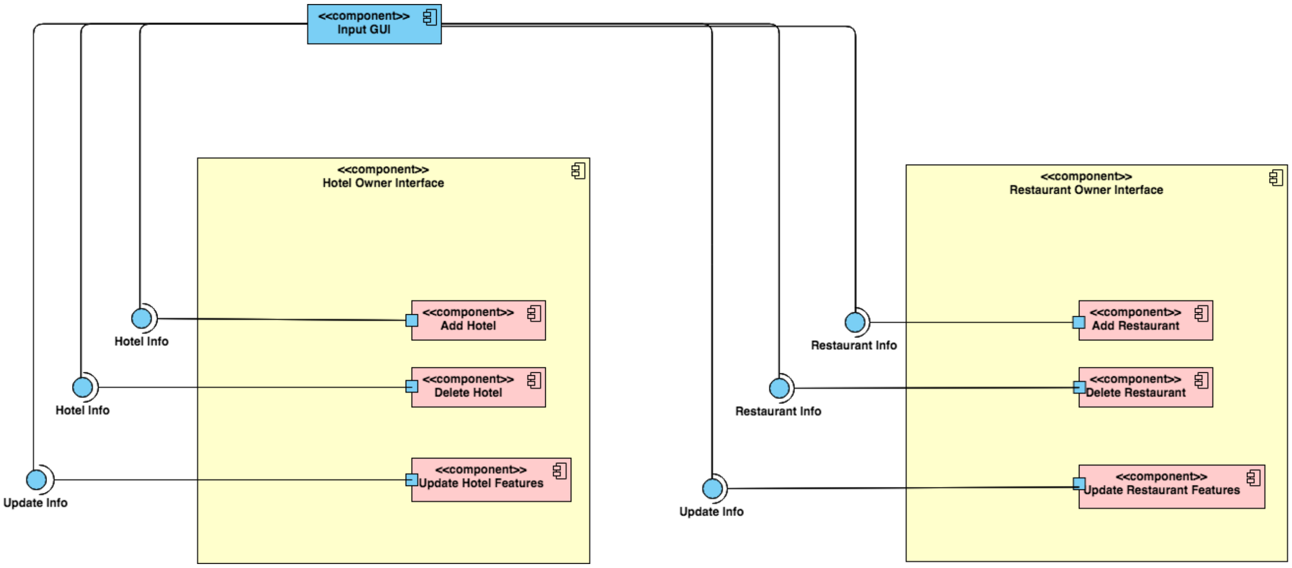

Click Book Restaurant
Displays Info
Edit info and continue
opens newpage, ask for more info
Enter info and confirm
Confirmed Successful, redirects to newpage
Book Restaurant
Verifies Info
Save Info
Restaurants
Incorrect Info/Missing Field

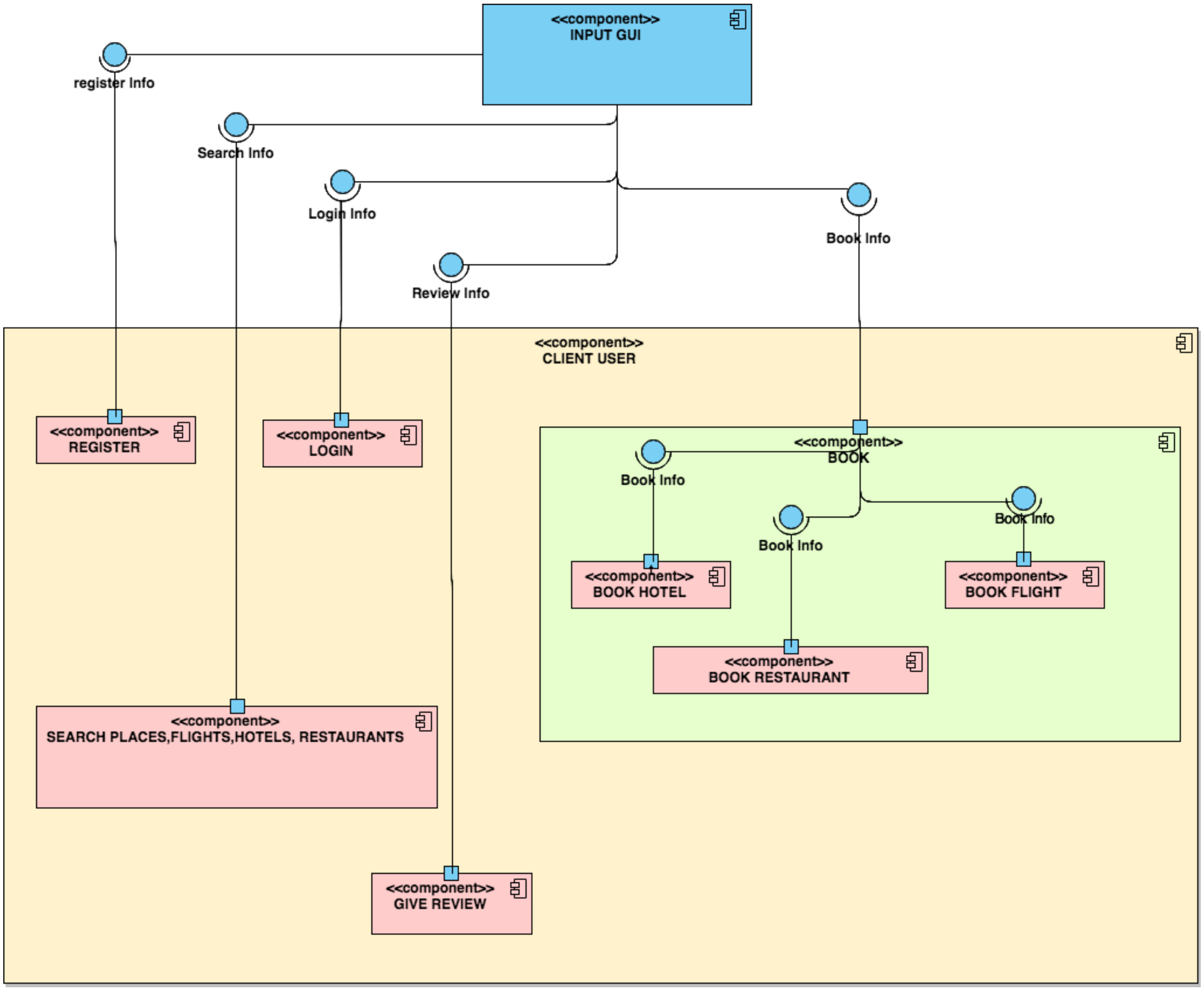

Click Book Hotel
Displays Info
Edit info and continue
opens newpage,asks for more info
Enter info and confirm
Confirmation,redirect to payment Page
Book Hotel
Verifies Info
SaveInfo
Hotels
Incorrect Info/Missing Field



Level 1 Data Flow Diagram

Appendix C: Design Models




<<component>>
INPUT GUI
register Info
Search Info
Login Info
Review Info
Book Info
<<component>>
CLIENT USER
<<component>>
REGISTER
<<component>>
LOGIN
<<component>>
BOOK
Book Info
Book Info
Book Info
<<component>>
BOOK HOTEL
<<component>>
BOOK FLIGHT
<<component>>
BOOK RESTAURANT
<<component>>
SEARCH PLACES,FLIGHTS,HOTELS, RESTAURANTS
<<component>>
GIVE REVIEW