Design & Modeling
Let’s Tour Pakistan

Deliverable 2 Submitted in Partial
Fulfillment of Assessment

for

Final Year Project

Prepared By

BSEF19M012 - Iqra Sarwar
BSEF19M027 - Mahnoor

Supervisor

Madiha Khalid

December 2022

Faculty of Computing &
Information Technology

CONTENTS

1. Introduction

Second deliverable of our project contains software design and modeling. In the previous deliverable, analysis of the system is completed. So we understand the current situation of the problem domain. Now we are ready to strive for a solution for the problem domain by using an object-oriented approach.Following artifacts are included in this deliverable document.

1. Use case description
2. Use case diagram refined
3. Domain Model
4. Sequence Diagram
5. Collaboration Diagram
6. Operation Contracts
7. Design Class Diagram
8. Data Model

2. UseCase Descriptions

A use case document is text that captures the detailed functionality of a use case. Descriptions of all use cases are written down.

2.1 UC_ExploreSite

Brief description

Home page is presented to the user when the user opens the application. Users can explore different features of the site. They can traverse the hierarchy of spots. By traversing this hierarchy they will be able to explore spots of specific cities and available facilities for them to avail. Users can also explore the services provided by the service providers. They will be able to get to know about pricing and availability of services. Users will also be able to read different blogs shared on the site. In order to avail these services or share their own blogs they will have to sign up to the site.

Preconditions

Users have access to the internet and a device with a web browser.

Basic flow

1. Users open the website.
2. Users traverse the hierarchy of spots and view different services.
3. Users view different blogs shared on the site.
4. Users decide either to sign up on site or leave without signing up.

Alternate flows

If the site is not available at any moment or the server is down, users might not be able to visit it.

Post conditions

Users have successfully visited the site.

2.2 UC_loginSignUp

Brief description

In order to avail packages, book services, write blogs and use other features of the website, users must be logged in. If the user's account does not exist already he signs up to the site by providing the personal information and the user role. If a user has an existing account, he logins and continues using the site.

Preconditions

To login, users must have signed up before.
To sign up, the user must have an email and knows his role as either tourist or service provider.

Basic flow

1. User opens the application and home page is presented
2. If a user has no account on site, he chooses to sign up.
3. User enters email, password, username, role(tourist or service provider)
4. If user is a tourist he can sign up successfully by specifying all above fields
5. If the user is a service provider, a contract is presented to the user, he agrees and signs the agreement.
6. Captcha verification is carried out before signup.
7. Users successfully sign up to the site.
8. If a user already has an account, he can log in by providing credentials.

Alternate flows

1. While signing up, the user enters an email that is already in use.
2. User enters a weak password to sign up
3. Service provider don't agree with contract and wants to negotiate
4. User forgets password and cannot login
5. User enters wrong email or password

Post conditions

User successfully signup as a tourist or service provider
User login to site successfully.

2.3. UC_AvailServices

Brief description

Users can avail services after they sign up for the site and login to it. User open the application and traverse in hierarchy to the spot they want to visit. User explore service they want to avail. Users select a service and request for booking. User pays for service online or chooses to pay later. Service is reserved for the user and the user leaves the website. Users can later view the services reserved for them.

Preconditions

user must have an account on site and user is logged in before availing the services.

Basic flow

1. User opens the website and finds the intended service.

2. User selects the service and books that service.
3. User chooses to pay online for the selected service.
4. User avails the service and leaves the site.

Alternate flows

1. Requested services are not available for booking. Users will not be able to avail the service.
2. Users want to pay for the service online but might run out of credit. Users can choose to pay later or avail service later.
3. Users can view the services they have availed.
4. Users can request to cancel the booking of a service within the specified cancellation duration defined by a service provider for that service..

Post conditions

service shall be reserved for the particular user.

2.4 UC_Help&Support

Brief description

Users interact with different features of the site and may come up with a problem. The possible problems are not being able to book a service, unable to get desired service, issues with the service provider services, to request for refund, problem with some feature, etc,. He visits the help and support section to contact the support agent. If a user finds a blog containing spam or false information, he can report it. On the other hand, if he finds a service provider or a specific service as a scam, he will report it as well which will then be managed by the admin.

Preconditions

User must be able to access the website
The support agent (Admin) needs to have an active internet connection and must be able to view the complaints or help requests submitted by the user.

Basic flow

1. User head over to the help and support section
2. User reads FAQs to find the solution to his problem
3. If FAQs does not help, user opens the request form to submit a complaint or ask for assistance
4. User submits the support request by entering his role and related information about the help he needs
5. Support agent (eg Admin) receives request for help and respond to user on provided email
6. User communicates with the agent until the problem is solved or concluded.

Alternate flows

1. Complaint form fails to save and submit

Post conditions

User successfully receives help and guidance related to problems from the support agent.

2.5. UC_OnlinePayment

Brief description

User books a service and chooses to pay for it online. User selects the appropriate payment method and provides the required information. Payment is verified and service is reserved.

Preconditions

The user must have an account to pay online.
The user must have booked the service.

Basic flow

1. User selects to pay online for a booked service.
2. User selects a payment method from the available choices.
3. User provides the information that is required to process the payment.
4. User confirms the transaction, and the payment will be deducted from the user's account.

Alternate flows

1. Payment method that the user wants to avail isn’t available.User will not be able to pay online.
2. User provides invalid or incorrect information. Payment will be terminated.
3. User provides correct information but doesn't have enough balance to pay for it.

Post conditions

The user has either paid successfully and service is reserved or payment is terminated.

2.6 UC_BlogPosts

Brief description

A blog can be shared by both tourists and service providers who want to share some information. A tourist can write a blog sharing his experience with a service provider at some tourist spots in some cities. It can be some recommendations or image-based blogs targeting some tour areas. A service provider can also write a blog showcasing their services or share some beautiful spot's information with images.

Preconditions

Users must be logged in to the site before adding, updating or deleting a blog.

Basic flow

1. User visits the Blogspot section
2. User reads the blog(s)
3. User selects a category, enters the title, description, images, and posts it
4. User may edit or delete it if he wants

Alternate flows

1. Blog contains inappropriate content such as false or useless information and admin deletes it
2. Blog is empty and is unable to post
3. Content is outdated and is reported as irrelevant by other users or requested by Admin to update

Post conditions

Blog is published successfully and appears on the site. Blog authors and other users are able to view this blog.

2.7. UC_ReviewsAndComments

Brief description

Users can post their reviews and comments on services and blogs shared on the website. Users might find a service helpful and want to share experience with other users or users might have found the service is spam and wants to alert other users.User might want to talk to the service provider about the service. In all of the above cases users can post comments and reviews on the services.

Similarly, users can share their reviews about the blogs shared on the website. Users might want to add something to the blog writer's idea or might want to share the updated information about the blog's topic. Users might also want to comment on the authenticity of the blog by providing their reviews.

Preconditions

The user must have an account and log in to it.

Basic flow

1. The user reads or visits a blog and wants to share something.
2. The user avail or explore services.
3. The user posts a review on a service or comments on a blog.
4. The user can delete or update his existing comments on a blog
5. The user can delete, update or read existing comments on a blog

Alternate flows

User might have posted something offensive, and the admin deletes that.

Post conditions

The user has successfully posted a review or comment about any service or blog.

2.8 UC_Manage_Services

Brief description

A service provider can add the services both individually and as a package. The service information can be updated or deleted by the service provider at any time. Users can add any type of service their company can provide such as tour guides, transportation, hotels, etc,. Tourists will then avail of these services according to their requirements.

Preconditions

1. Service providers must be registered users of platform
2. Service providers must have logged in to the system to manage the services they provide

Basic flow

1. Services provider login to the application
2. The Services provider adds the mandatory information for the service they are offering like price, service type, contact information and special discounts if any, etc.
3. Services provider and other users can view the added services
4. The service providers can edit, update and delete the information provided by him/her.

Alternate flows

1. The added service is identified to be a scam or fake and is removed by admins.

2. Service is irrelevant i.e not related to the platform

Post conditions

Service providers can perform CRUD operations on the services and these services are presented to tourists in a hierarchy of places and service types.
Users can view services, book them and connect with service providers.

2.9. UC_ViewBooking

Brief description

Service providers offer different services on the website. users book their services. So, service providers have booking records for their services. service providers can choose to view bookings for a particular service. All the current and past bookings record will be available to the service provider.

Preconditions

User must be a service provider.

Basic flow

1. User opens the application.
2. User selects a service to view bookings.
3. All bookings are presented to the service provider.
4. Service providers can view past bookings and the current bookings.
5. Service providers can’t modify this record.

Alternate flows

1. Service providers must have added some services. If there is no service added service provider will not be able to view bookings.
2. There are no bookings for any particular service provided by the service provider.

Post conditions

The service provider has successfully viewed the bookings history.

2.10 UC_Packages&Discounts

Brief description

A service provider can add special packages that are exclusively for privileged users. He can add or remove a discount from the services at any time.

Preconditions

1. The service provider must be a registered user of platform
2. The service provider must have logged in to the system to add services packages

Basic flow

1. Services provider logins to the site
2. A Service provider adds the mandatory information for the package of services such as discount and users status for which a discounted package is available.

Alternate flows

User is banned from the site and not allowed to add any service package
The added services package is identified to be a scam or fake and is removed by admins.

Post conditions

A discounted package of services is successfully added and is visible to all privileged users

2.11 UC_ServicePlans

Brief description

The website provides in-app plans to the tourists which offers some special privileges to them according to plan type such as basic, premium etc,.
Tourist becomes a privileged user by purchasing the service plan and can avail the exclusive deals and discounts on service packages.

Preconditions

Tourists must be a registered user on the site and be logged in before availing a specific plan.

Basic flow

1. User signup and logins to the site.
2. User selects an appropriate plan for him.
3. User pays online for the selected plan.
4. Users avail the incentives.

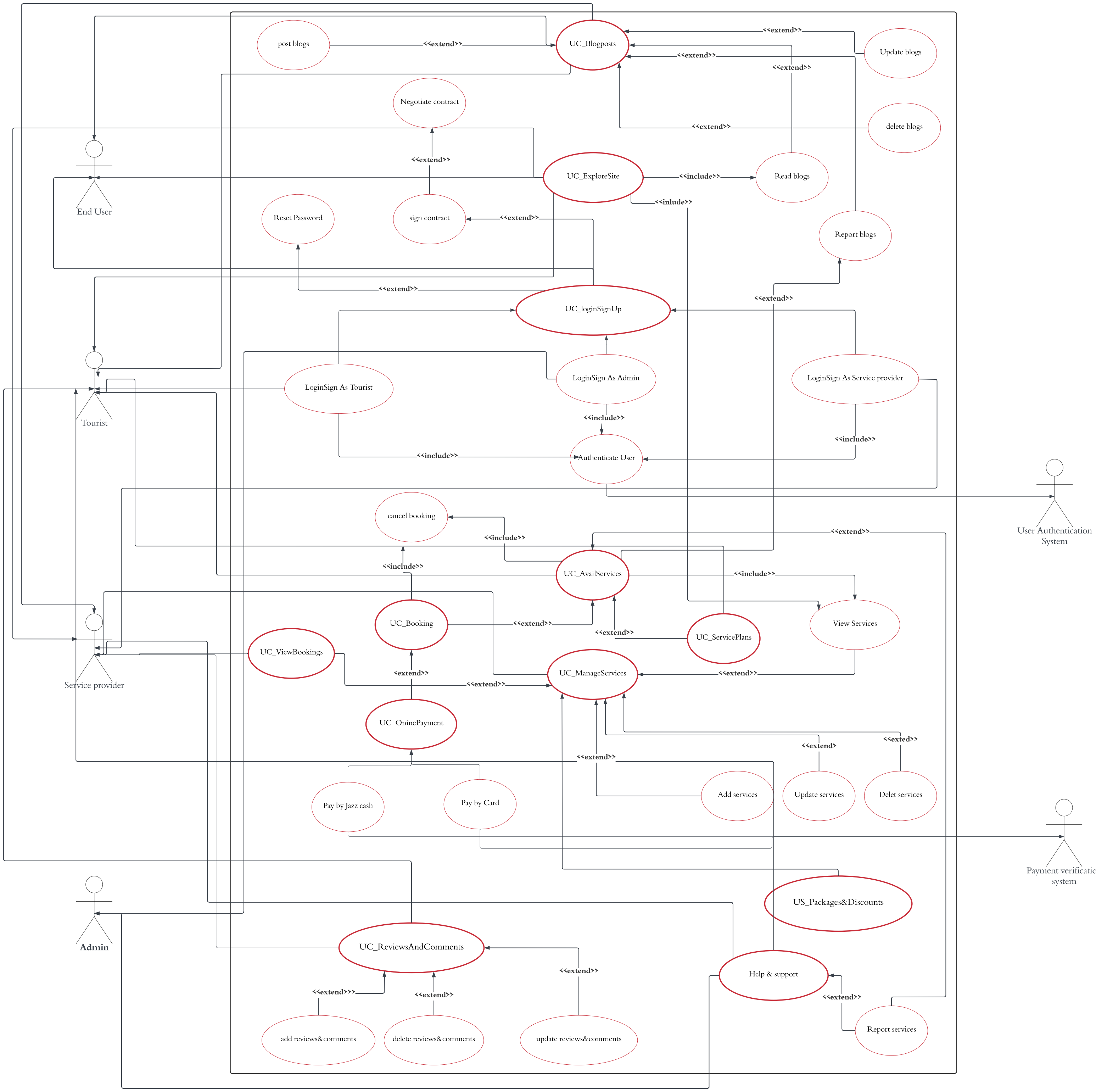
Alternate flows

1. Payment becomes unsuccessful due to any reason
2. User is not login and cannot continue to purchase a plan

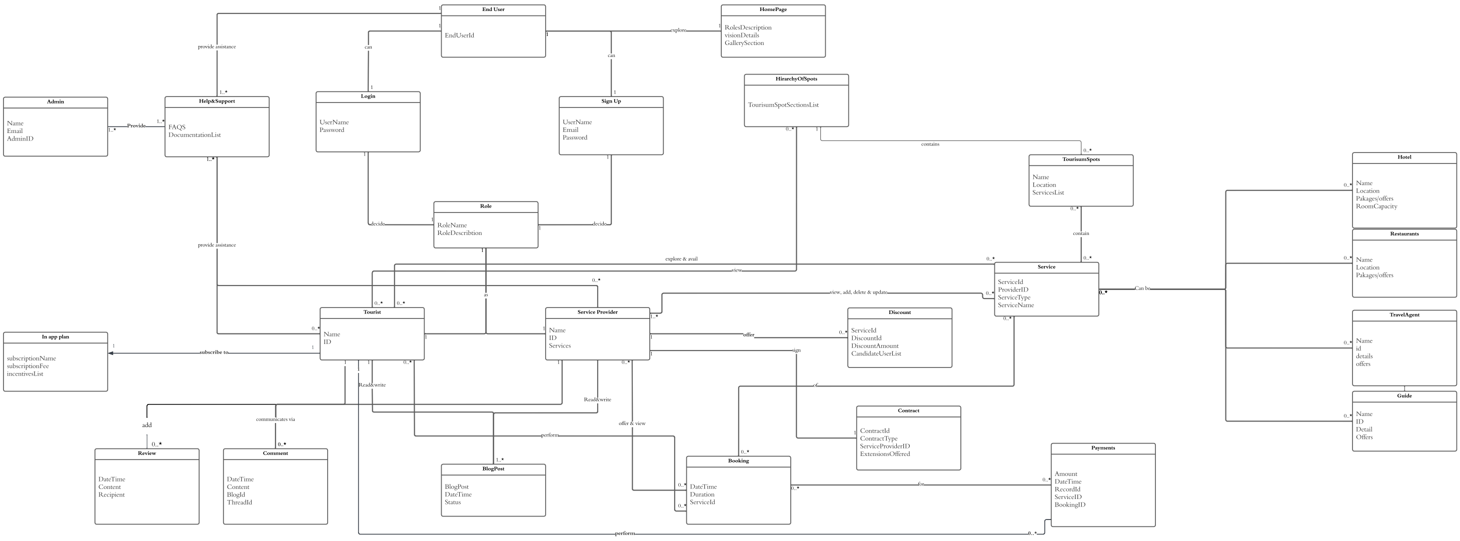
Post conditions

User’s status changes as he becomes a privileged user and can avail the incentives on the site

3. Analysis Level Use Case Diagram

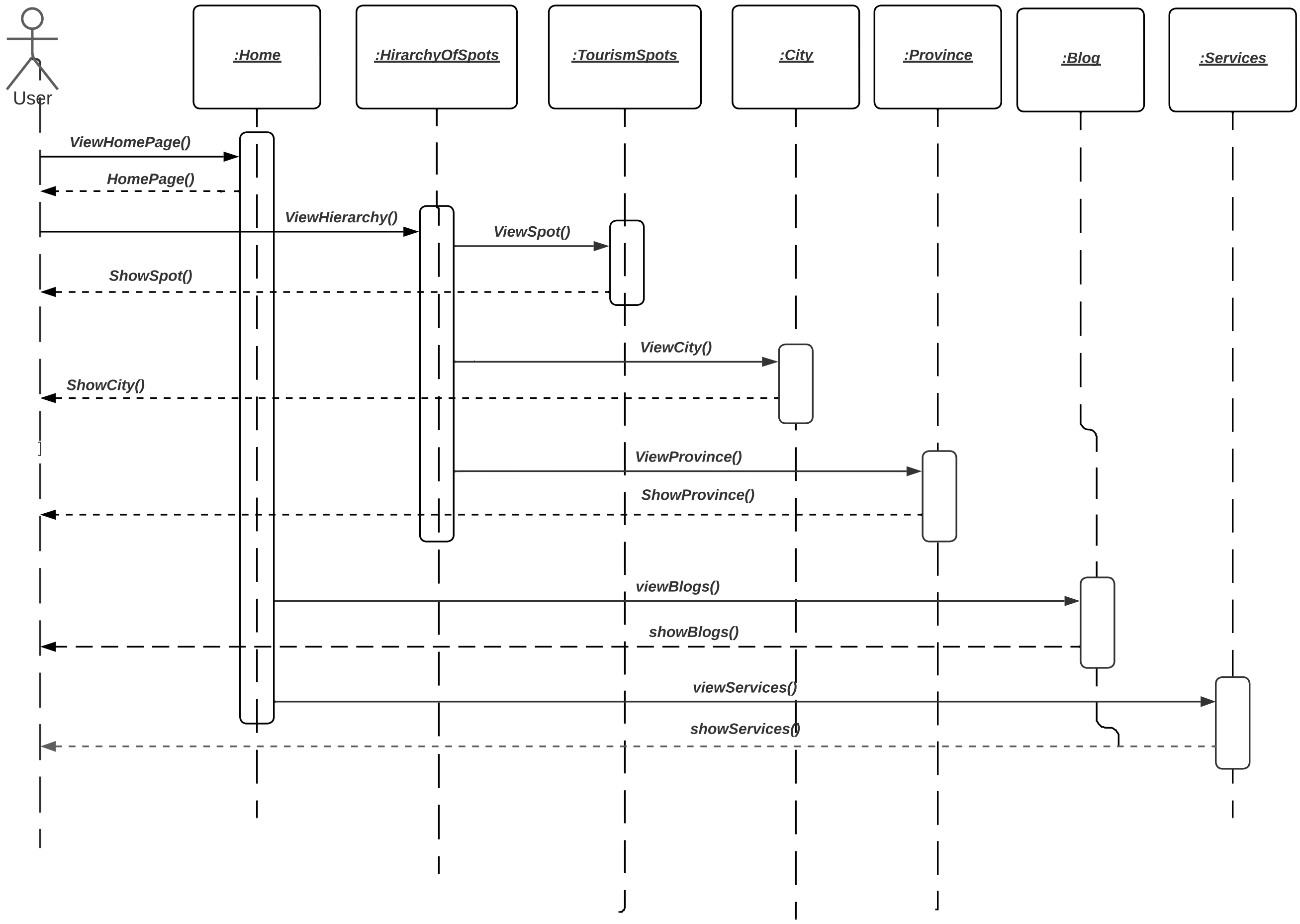


4. DOMAIN MODEL

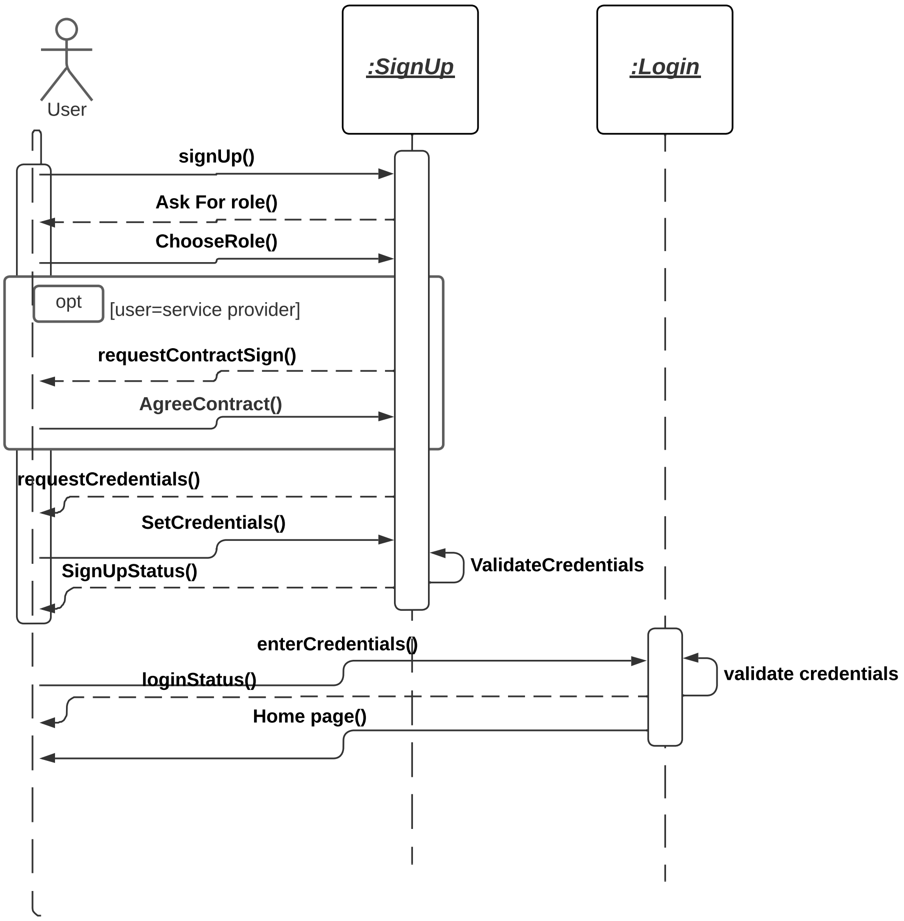


5. Sequence Diagrams

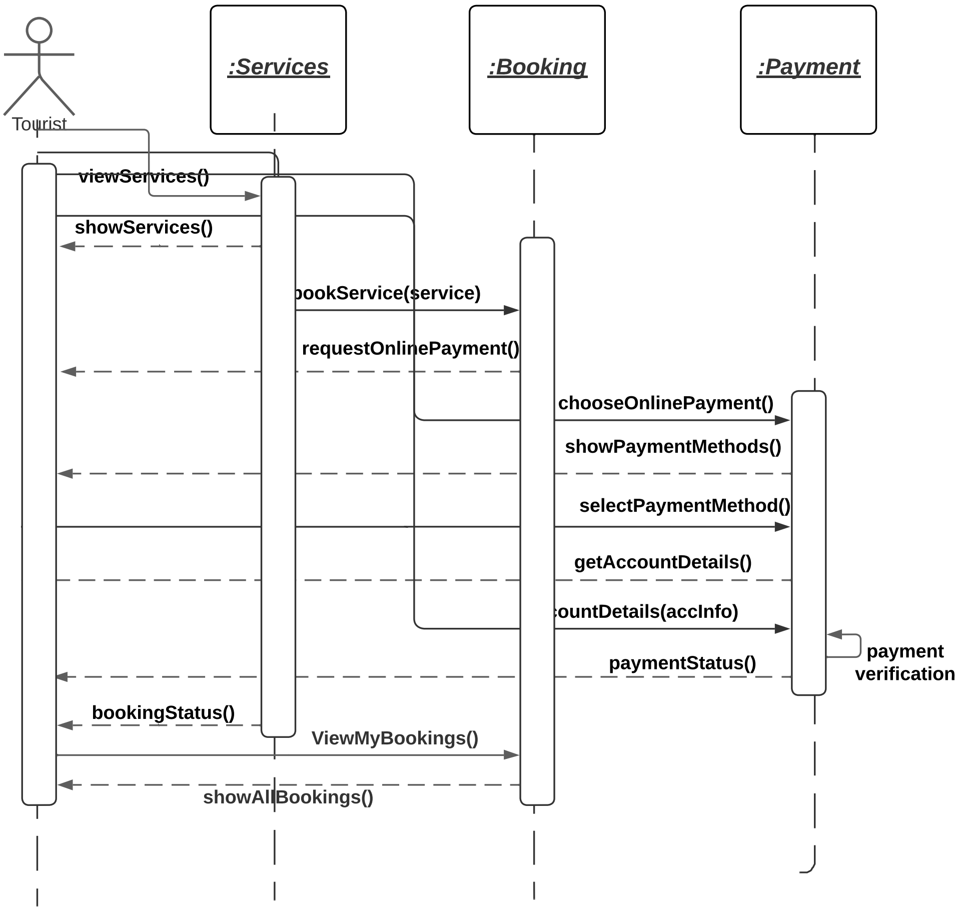
UC_ExploreSite



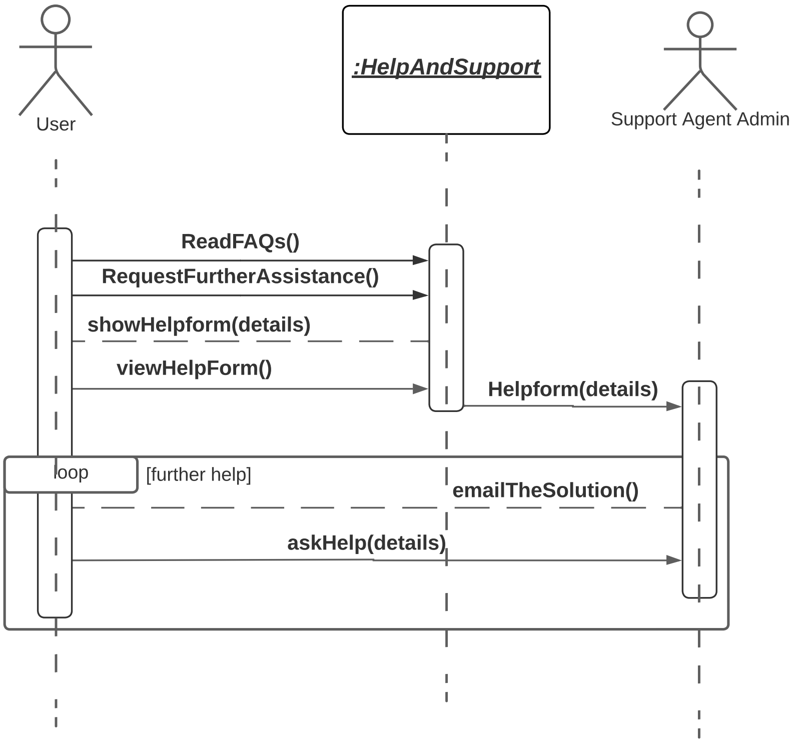
UC_LoginSignUp



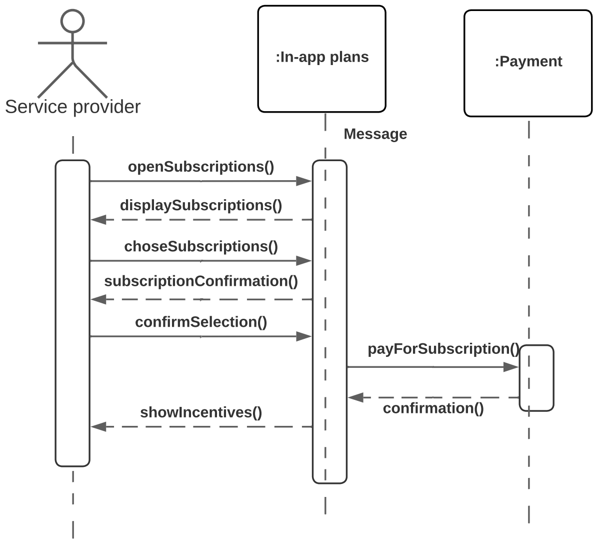
UC_AvailServices



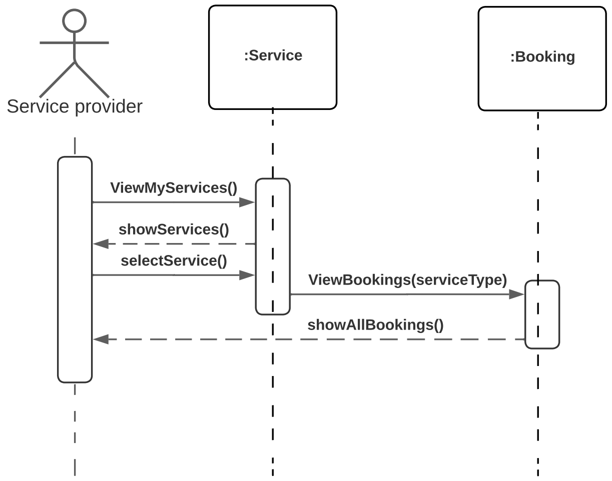
UC_Help&Support



UC_servicePlans

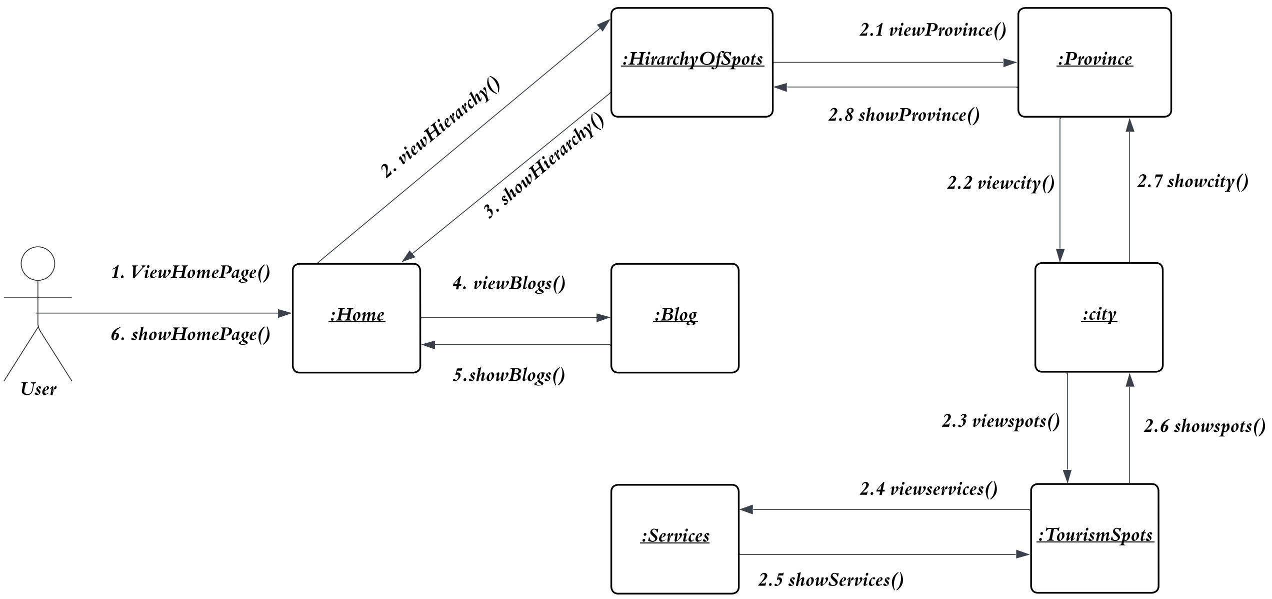


UC_viewBoookings

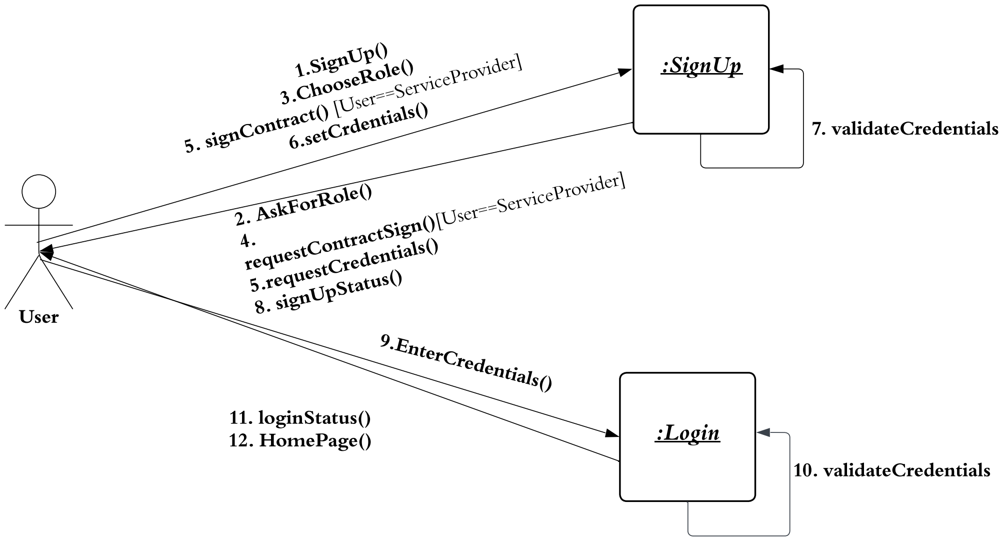


6. Collaboration Diagrams

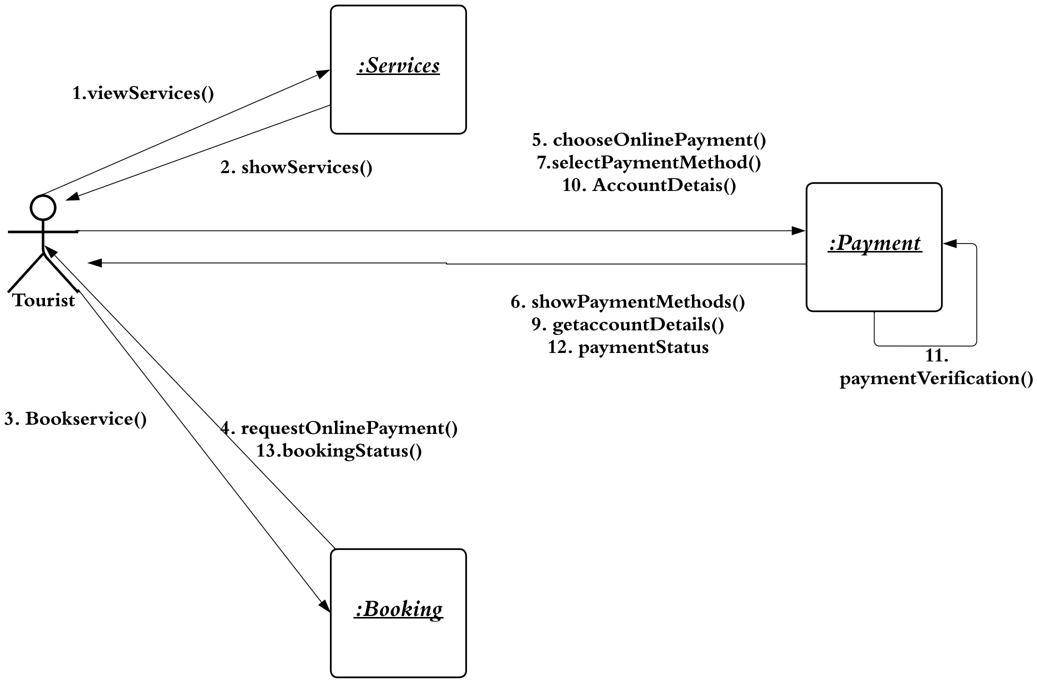
UC_ExploreSite



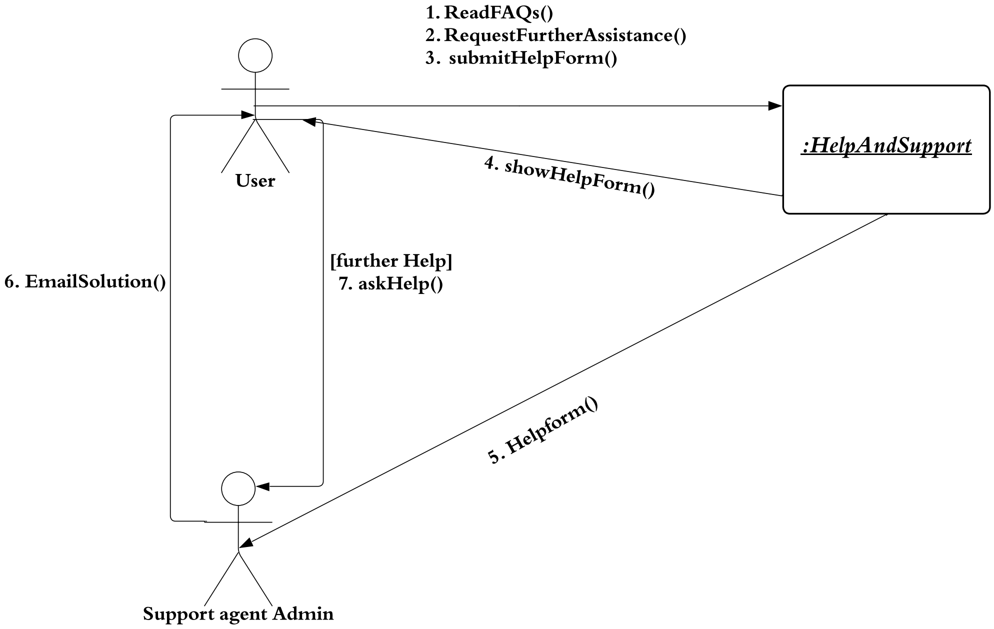
UC_LoginSignUp



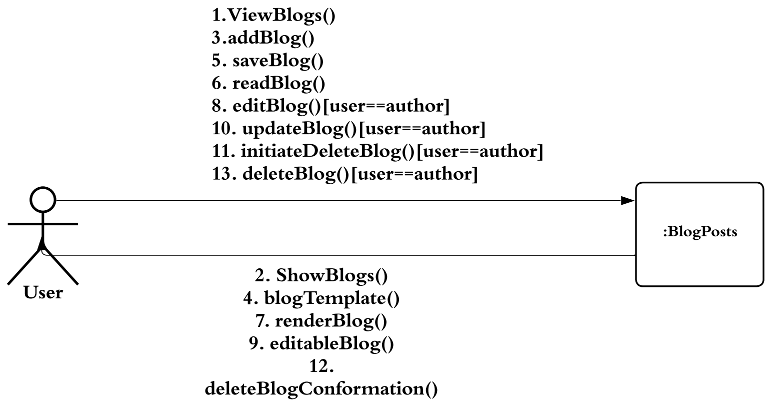
UC_AvailServices



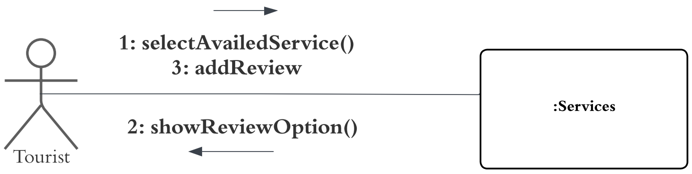
UC_Help&Support

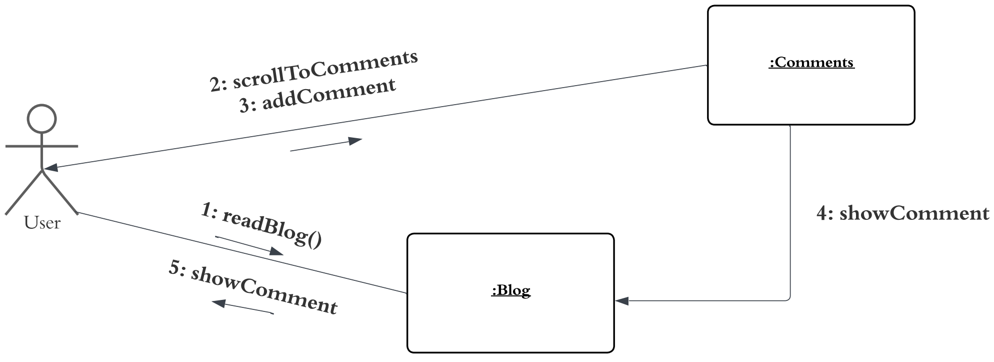


UC_BlogPosts

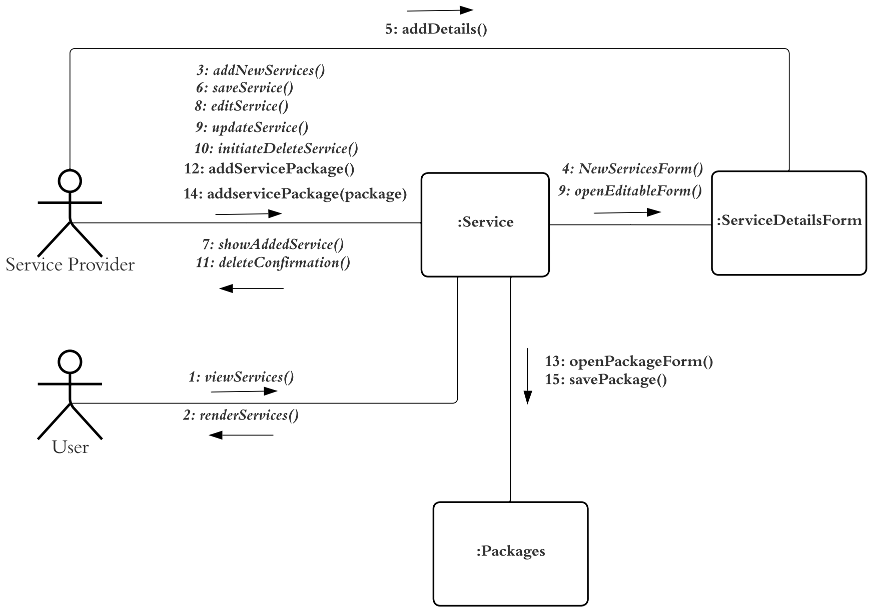


UC_Reviews&Comments

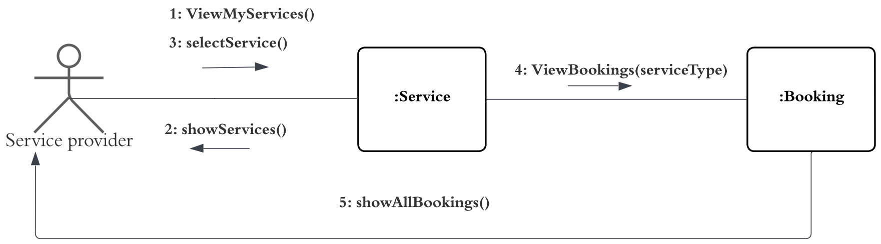




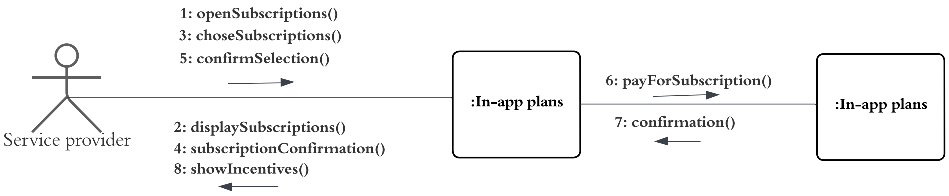
UC_ManageServices



UC_viewBoookings



UC_servicePlans



7. Operational Contracts

UC_ExploreSite

OC_1:

Operation: ViewHomePage() {static}

Cross References: UC_ExploreSite

PreConditions:
User opened the application and is exploring the home page.

PostConditions:

showHomePage() was called which returns all the blogs, and interface hierarchy and renders it to the screen.(No associations needed as the function is a static one).

OC_2:

Operation: ViewBlogs() {static}

Cross References: UC_ExploreSite

PreConditions:
User opened the application and is exploring the home page.

PostConditions:

- showBlogs() was called which returns all the blogs and renders them to the screen.
- Blogs are displayed on the home page.(No associations needed as the function is a static one).

OC_3:

Operation: ViewHierarchy() {static}

Cross References: UC_ExploreSite

PreConditions:
User opened the application and is exploring the home page.

PostConditions:

showHierarchy() was called which returned all the hierarchy of provinces, cities, tourism spots and services associated with them. This is rendered on the screen as a part of the Home Page.(No associations needed as the function is a static one).

UC_LoginSignUp

OC_1:

Operation: ChooseRole()

Cross References: UC_LoginSignUp

PreConditions:
The user must be an eligible person having complete knowledge of his role.

PostCondition:

- An instance “role” of UserRole was created.
- The role was identified as either tourist or service provider.
- The instance role’s string attribute has been assigned a value.

OC_2:

Operation: signUp()

Cross References: UC_LoginSignUp

PreConditions:
The user must be an eligible person having complete knowledge of his role.User must agree to provide all required information and sign a contract in case he is a service provider.

PostConditions:

- User signed up successfully as a tourist or service provider and was able to complete his profile.
- An instance of the class corresponding to the user role was created.
- The user has assigned a unique username and password.
- Password and username attributes of Signup class were assigned a value.

OC_3:

Operation: login()

Cross References: UC_LoginSignUp

PreConditions: User must have an account.

PostConditions:

- Users logged in as tourists or service providers.
- If a user was a service provider, he/she can offer his/her services as a list “Services” of Service class instances is created.

- If the user was a tourist, he/she is able to avail services.
- If the user was a tourist, the user has a TouristID.
- The service provider has an instance of Contract that he/she has signed with an application.
- If the user is a service provider, the user has a ServiceProviderID. Tourist/service provider has an instance of HomePage.
- Through that instance the user is able to delete, update, view, add content such as blogs.

OC_4:

Operation: ValidateCredentials()

Cross References: UC_LoginSignUp

PreConditions:

User has opened the application and wants to login/signup, hence provides the requested credentials.

PostConditions:

- The user was proved to be a human and declared safe for further login/signup operation.
- The user was proved to be the genuine owner of the account.
- It was declared that the user is not a scam.

OC_5:

Operation: SignContract()

Cross References: UC_LoginSignUp

PreConditions: A registered, authenticated service provider is signing up to the application.

PostConditions:

- An instance of Contracts contract was created.
- Contract was associated with the ServiceProvider Class for the contract to be signed by a service provider.
- ServiceProviderID in serviceProvider class became serviceId in Contracts class.

OC_7:

Operation: setCredentials()

Cross References: UC_LoginSignUp

PreConditions:

The user was requested to enter required information to sign up. users have entered valid information that is verified and required to be recorded permanently.

PostConditions:

- An instance of User respective to role was created.
- Credentials were set according to the provided data and stored in the database.

OC_6:

Operation: EnterCredentials()

Cross References: UC_LoginSignUp

PreConditions:

User has successfully signed up in the application. User wants to login to his existing account.

PostConditions:

Users provided the required credentials for login that can be verified for authenticity.

UC_BlogPosts

OC_1:

Operation: viewBlogs()

Cross References: UC_BlogPosts

PreConditions:

The user is exploring the site. Login or sign up is not mandatory.

PostCondition:

showBlogs() was called to initiate a fetch request for all the available blogs. Returned blogs are rendered to the screen.

OC_2:

Operation: addBlog()

Cross References: UC_BlogPosts

PreConditions:

User is a registered tourist or service provider and logged in to the site.

PostCondition:

- blogTemplate() returned a template of blog posts that users can use to add their blog.
- After the user has provided the required information saveBlog() was initiated to save it to the database.

OC_3:

Operation: saveBlog()

Cross References: UC_BlogPosts

PreConditions:
A blog with the accurate information is provided according to the template presented to the user.

PostCondition:

The blog was saved to the blogs collection in the database and rendered on the home page when users explore the site.

OC_4:

Operation: readBlog()

Cross References: UC_BlogPosts

PreConditions:
All blogs are presented to the user, and the user chooses a particular blog to read.

PostCondition:

The selected blog was presented to the user to read with all the relevant information about the author and related comments.

OC_5:

Operation: editBlog()

Cross References: UC_BlogPosts

PreConditions:

- The user is logged in as a tourist or service provider.
- The user is the author of any blog post.
- The user wants to update his blog post.

PostCondition:

editableBlog() returned an editable blog to the user.

User edited the returned blog and requested to update its record.

OC_6:

Operation: updateBlog()

Cross References: UC_BlogPosts

PreConditions:
The user has edited his authored blog and wants to update its record.

PostCondition:

- Update request was sent to update the blog record in the database.
- Record was updated in the database and the updated blog is rendered with the collection of blogs onwards.

OC_7:

Operation: initiateDeleteBlog()

Cross References: UC_BlogPosts

PreConditions:

- The user is logged in as a tourist or service provider.
- The user is the author of any blog post.
- The user wants to delete his blog post.

PostCondition:

- Delete request was initiated to delete the blog record in the database.
- A confirmation result was generated for the user to confirm deletion of the blog.

OC_8:

Operation: deleteBlog()

Cross References: UC_BlogPosts

PreConditions:

The user confirms that he wants to delete the blog.

PostCondition:

- Delete request was sent to delete the blog record in the database.
- Record was deleted in the database and the deleted blog is never rendered again from the site as it is no longer present in the database.

UC_Help&Support

OC_1:

Operation: ReadFAQS(){static}

Cross References: UC_Help&Support

PreConditions:

An authenticated registered user has been logged in to the site and wants to read FAQS.

PostCondition:

All available FAQS were presented to the user to seek guidance from them.

OC_2:

Operation: RequireFurtherAssistance() {static}

Cross References: UC_Help&Support

PreConditions:

An authenticated registered user has been logged in to the system and needs assistance regarding the site other than FAQS.

PostCondition:

showHelpform() was called to display a form that the user can fill in with the appropriate information to seek assistance from admins.

OC_3:

Operation: submitHelpForm()

Cross References: UC_Help&Support

PreConditions:

An authenticated registered user has been logged in to the system and is seeking assistance from admins.

PostCondition:

- The information provided by the user was submitted to the admins for further processing.
- Admins can reach out to the user via email to provide assistance.

UC_AvailServices

OC_1:

Operation: viewSerivces()

Cross References: UC_AvailServices

PreConditions:

User is exploring the site and wants to view the services associated with a particular tourism spot.

PostCondition:

All services related to a particular spot were rendered for the user. users can select from those to book.

OC_2:

Operation: bookSerivce()

Cross References: UC_AvailServices

PreConditions:

The user is registered as a tourist and logged in to the site. User wants to book a service that is available.

PostCondition:

- A Booking instance b was created.
- b was associated with the Service selected for booking.
- ServiceID in services became ServiceID in Bookings.
- The user is prompted to choose a payment method for online payment or pay onsite.

UC_OnlinePayments

OC_1:

Operation: showPaymentMethods()

Cross References: UC_OnlinePayments

PreConditions:

- The user is logged in as a tourist.
- The user is booking a service.

- The user chooses to pay online.

PostCondition:

- All available payment methods were rendered on screen for the user.
- The user can select one of them to pay.

OC_2:

Operation: selectPaymentMethod()

Cross References: UC_OnlinePayments

PreConditions:

- A registered, authenticated, and logged in tourism is booking a service.
- He wants to pay for it online and has multiple methods to pay online.

PostCondition:

- The user has selected one of the available online payment methods to pay online.
- Users can provide account details according to the chosen payment method.

OC_3:

Operation: accountDetails()

Cross References: UC_OnlinePayments

PreConditions:

- Tourist is booking a service and is selected to pay online via a particular method.
- The user is prompted to provide details according to the selected method by getAccountDetails().

PostCondition:

- Users have filled in the required information including personal details and account details for online payment.
- Information was submitted for verification of the payment details.

OC_4:

Operation: paymentVerification()

Cross References: UC_OnlinePayments

PreConditions:

User is paying online to book a service and has provided all of the required information according to the selected payment method.

PostCondition:

- The provided information is verified and ensured to be a valid account with enough balance to pay for service.
- when authenticated, Payment instance p was created.
- p was also associated with Bookings
- Id in Bookings became booking Id in Payments.
- The payment was processed and the payment status method was dispatched to the user.

UC_ReviewsAndComments

OC_1:

Operation: addComment()

Cross References: UC_ReviewsAndComments

PreConditions: The user is a registered tourist or service provider and logged in to the site.

PostCondition:

- A new comment object was initiated referring to a specific blog.
- After the user has provided the required information saveComment() was initiated to save it to the database.

OC_2:

Operation: saveComment()

Cross References: UC_ReviewsAndComments

PreConditions:

- The user writes a comment in an editable comment version on a blog

PostCondition:

- The Comment was saved to the Comments collection in the database and rendered on the home page when users explore the site.

OC_3:

Operation: updateComment()

Cross References: UC_ReviewsAndComments

PreConditions:

- The user has edited his own comment in an editable comment version of the comment and wants to update its record..

PostCondition:

- Update request was sent to update the Comment record in the database.
- Record was updated in the database and the updated Comment was rendered with the collection of Comments onwards.

OC_4:

Operation: deleteComment()

Cross References: UC_ReviewsAndComments

PreConditions:

- The user confirms that he wants to delete the Comment.

PostCondition:

- Delete request was sent to delete the comment in the database.
- Record was deleted in the database and the deleted Comment is never rendered again from the site as it is no longer present in the database.

OC_5:

Operation: addComment()

Cross References: UC_ReviewsAndComments

PreConditions:

- The tourist is logged in to the site and has availed the service on which he wants to add review.

PostCondition:

- A new Review object was initiated referring to a specific blog.
- After the tourist has provided the required information saveComment() was initiated to save it to the database.

OC_6:

Operation: saveComment()

Cross References: UC_ReviewsAndComments

PreConditions:

- The tourist writes a Review
- in an editable Review version on a blog

PostCondition:

- The Review was saved to the Reviews collection in the database and rendered on the home page when tourists explore the site.

OC_7:

Operation: updateComment()

Cross References: UC_ReviewsAndComments

PreConditions:

- The tourist has edited his own
- Review in an editable Review version of the Review and wants to update its

record.

PostCondition:

Update request was sent to update the Review record in the database.

Record was updated in the database and the updated Review is rendered with the collection of Reviews onwards.

OC_8:

Operation: deleteComment()

Cross References: UC_ReviewsAndComments

PreConditions:

- The tourist has availed the service whose review he wants to delete added by him.
- Tourist confirms that he wants to delete the Review.

PostCondition:

- Delete request was sent to delete the Review in the database
- Record was deleted in the database and the deleted Review is never rendered again from the site as it is no longer present in the database.

UC_BOOKINGS

OC_1:

Operation: ViewBookings(serviceType)

Cross References: UC_Bookings

PreConditions:

- Service provider has successfully logged in to the site.
- Service provider gets to see all his services with ViewMyServices()
- Service provider selects a service to see bookings

PostCondition:

All bookings related to a serviceType are rendered on site UI.

UC_Manage_Services

OC 1:

Operation: AddService()

Cross References: UC_Manage_Services

PreConditions :

The user is a registered service provider

User has successfully logged in to the site.

PostConditions:

- servicesForm() returned a form to add a new service of any category.
- The added services was saved by saveService() and is added to the database which is then displayed on site by showAddedService()

OC 2:

Operation: DeleteService()

Cross References: UC_Manage_Services

PreConditions :

Service provider has successfully logged into the site and owns a service he/she wants to delete.

PostConditions:

- Delete request was sent to delete the service record in the database.
- Service was deleted in the database and the deleted blog was never rendered again from the site as it was no longer present in the database.
- The service was never rendered again in the UI hierarchy.

OC 3:

Operation: editService()

Cross References: UC_Manage_Services

PreConditions :

Service provider has successfully logged into the application.

Service provider is the owner of the service

PostConditions:

- editableBlog() returned an editable blog to the user.
- User edited the returned blog to change information/attributes of a service and requested to update its record.

OC 3:

Operation: UpdateService()

Cross References: UC_Manage_Services

PreConditions :

- Service provider has successfully logged into the application
- User is the owner of the service he/she wants to update.

PostConditions:

- Update request was sent to update the service record in the database.
- Record was updated in the database and the updated service object is rendered in the services list.

UC_Packages&Discounts

OC 1:

Operation: addPackages()

Cross References: US_Packages&Discounts

PreConditions : The user is a registered service provider

User has successfully logged in to the site.

PostConditions:

showPackageForm() returned a form to add a new services package.

OC 2:

Operation: savePackage()

Cross References: UC_ServicePlans

PreConditions :

- The user is a registered service provider.
- User has successfully logged in to the site.
- User has entered package details in add package form

PostConditions:

- The added services were saved by savePackage() and are added to the database.
- New instance of a package is created.

UC_ServicePlans

OC 1:

Operation: choseSubscriptions()

Cross References: UC_ServicePlans

PreConditions:

- User is a tourist and is a registered service provider.
- User has successfully logged in to the site.
- A subscription plan is displayed to the user

PostConditions:

- A subscription plan is selected for a tourist

OC 2:

Operation: payForSubscription()

Cross References: UC_ServicePlans

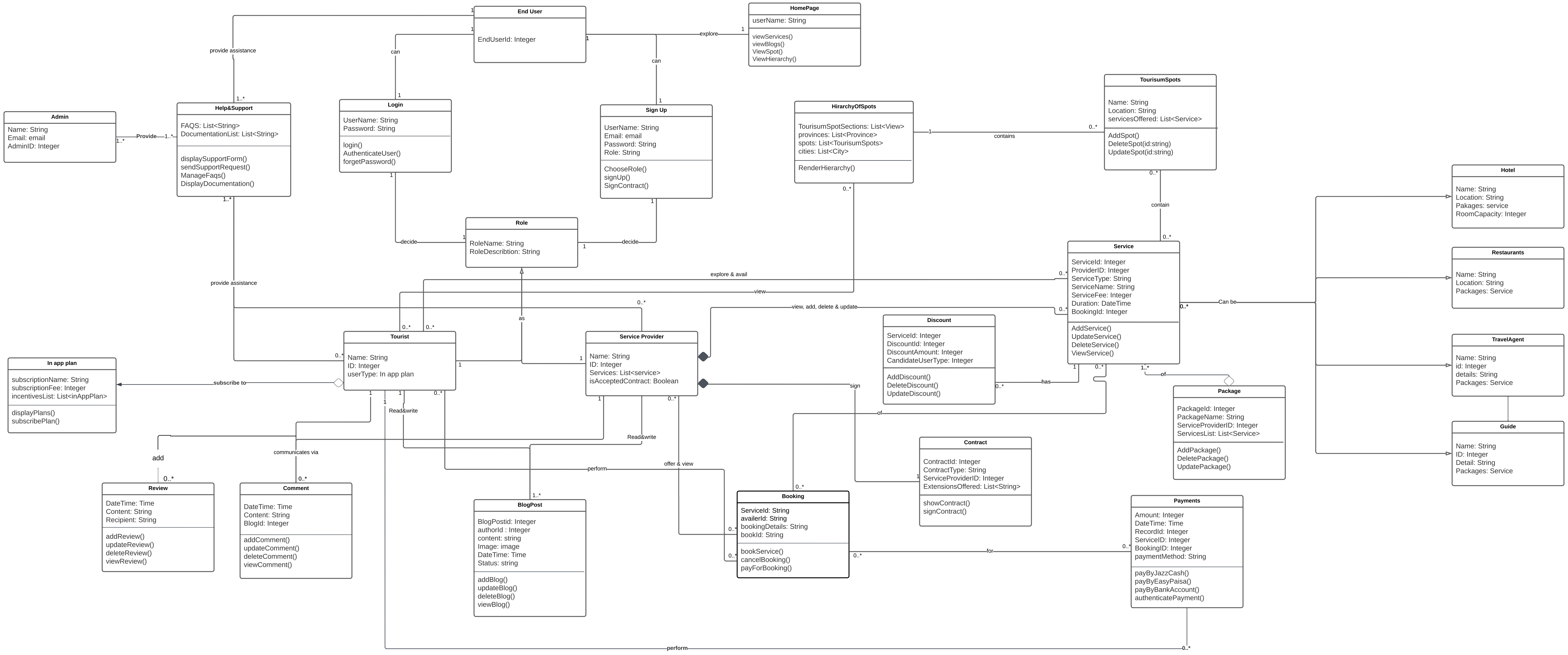
PreConditions:

- A subscription plan is selected by a tourist.

PostConditions:

- New instance of a subscription plan is created
- User has got a new subscription and is a privileged user.

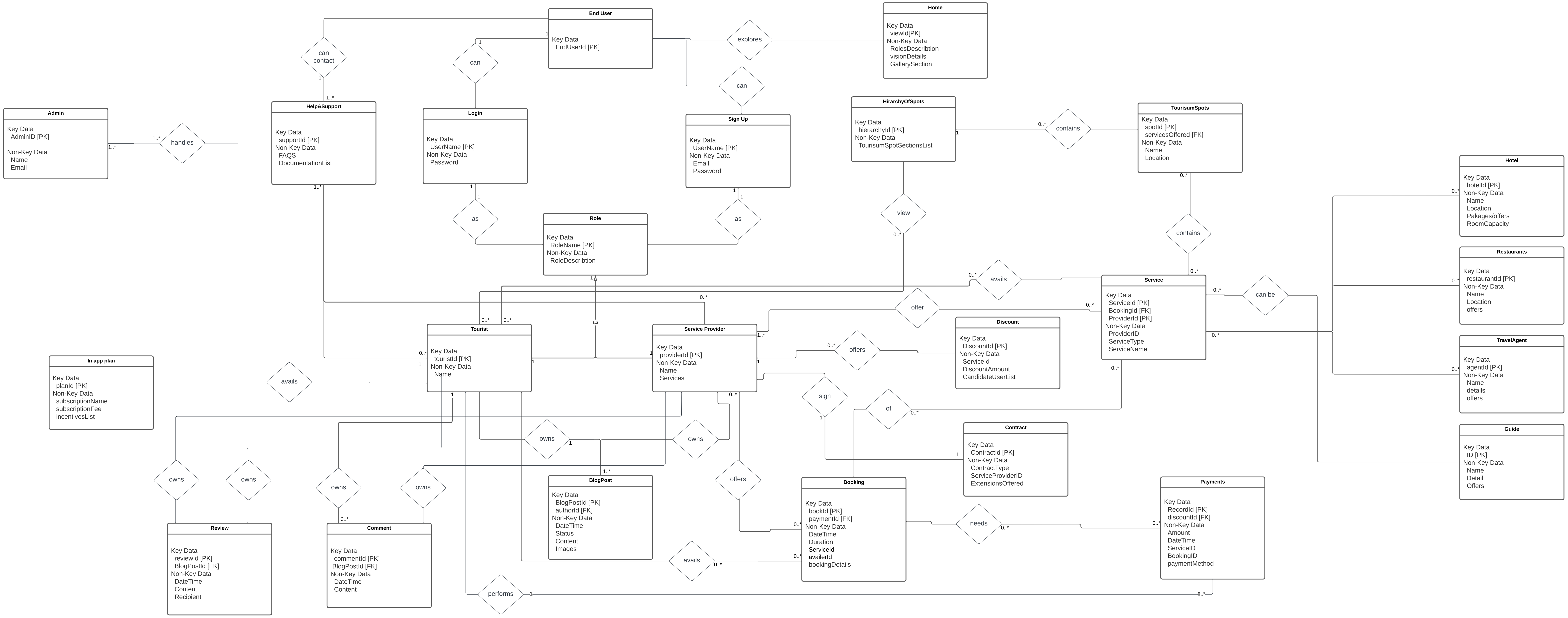
8. DESIGN CLASS DIAGRAM



9. Data Model

Entity Relationship Matrix

Entities	end user	admin	login	sign up	role	tourist	service provider	blog posts	in app plans	reviews	comments	booking	home page	discounts	hierarchy of spots	contracts	tourism spots	payment	se
end user		can be	can	can	is ass to	can be	can be			can view	can view		explores		can view	can view	can view		ca vi
admin			can																
login					as	as	as												
signup					as	as	as												
role						as	as												
tourist			can		have			owns	avails	owns	owns	avails	can view	avails	can view		can view	can perform	b vi
service provider			can		have			owns	avails	owns	owns	offer	can view	offer	can view	can sign	can view		o
blog posts	can view					by	by				have								
in app plans				on		avails	avails												
reviews						by	by												fo
comments						by	by	for											
booking														have				needs	o
home page								have	have						have	have	have		h
discounts						for												for	fo
hierarchy of spots													resides in				contains		
contracts				for			for												
tourism spots															resides in				co n
payment						by	to					on		contains					fo
service						for	by			have		provide		have			for	have	
restaurants						for	by			have		provide		have			for	have	ca
guides						for	by			have		provide		have			for	have	ca
hotel						for	by			have		provide		have			for	have	ca
travel agents						for	by			have		provide		have			for	have	ca
Help & support	availed by	by				for	for												



End User
Key Data
EndUserId [PK]

Home
Key Data
viewId[PK]
Non-Key Data
RolesDescribtion
visionDetails
GallarySection

Admin
Key Data
AdminID [PK]
Non-Key Data
Name
Email

Help&Support
Key Data
supportId [PK]
Non-Key Data
FAQS
DocumentationList

Login
Key Data
UserName [PK]
Non-Key Data
Password

Sign Up
Key Data
UserName [PK]
Non-Key Data
Email
Password

HirarchyOfSpots
Key Data
hierarchyId [PK]
Non-Key Data
TourisumSpotSectionsList

TourisumSpots
Key Data
spotId [PK]
servicesOffered [FK]
Non-Key Data
Name
Location

Role
Key Data
RoleName [PK]
Non-Key Data
RoleDescribtion

Hotel
Key Data
hotelId [PK]
Non-Key Data
Name
Location
Pakages/offers
RoomCapacity

Restaurants
Key Data
restaurantId [PK]
Non-Key Data
Name
Location
offers

Service
Key Data
ServiceId [PK]
BookingId [FK]
ProviderId [PK]
Non-Key Data
ProviderID
ServiceType
ServiceName

Tourist
Key Data
touristId [PK]
Non-Key Data
Name

Service Provider
Key Data
providerId [PK]
Non-Key Data
Name
Services

Discount
Key Data
DiscountId [PK]
Non-Key Data
ServiceId
DiscountAmount
CandidateUserList

TravelAgent
Key Data
agentId [PK]
Non-Key Data
Name
details
offers

In app plan
Key Data
planId [PK]
Non-Key Data
subscriptionName
subscriptionFee
incentivesList

Contract
Key Data
ContractId [PK]
Non-Key Data
ContractType
ServiceProviderID
ExtensionsOffered

Guide
Key Data
ID [PK]
Non-Key Data
Name
Detail
Offers

BlogPost
Key Data
BlogPostId [PK]
authorId [FK]
Non-Key Data
DateTime
Status
Content
Images

Booking
Key Data
bookId [PK]
paymentId [FK]
Non-Key Data
DateTime
Duration
ServiceId
availerId
bookingDetails

Payments
Key Data
RecordId [PK]
discountId [FK]
Non-Key Data
Amount
DateTime
ServiceID
BookingID
paymentMethod

Review
Key Data
reviewId [PK]
BlogPostId [FK]
Non-Key Data
DateTime
Content
Recipient

Comment
Key Data
commentId [PK]
BlogPostId [FK]
Non-Key Data
DateTime
Content