Application: Travel Wise
Author: Yan Peng
2024.1

1. Data Definitions V2

Data Item	Definition	Usage and User Privileges	Sub Data & Main Info (Raw data, metadata, supporting data)
Users	Registered individuals on the platform with personal accounts	Users can utilize all the features in the app, including 1. Users can create, edit, and delete their blog posts, and comments, and interact with other users' content. 2. Users can read and search any travel posts. 3. Users can express their appreciation for blog posts by liking them. 4. Users can save posts to their favorites list for easy access and future reference. 5. Users can follow other users' posts. 6. Users can post comments on blog posts to share feedback and reply to comments. 7. Users can interact with the AI assistant to receive personalized travel suggestions and plans. 8. Users can update and delete their accounts. 9. Users can view other author's profiles.	• User ID • Username • Email • Password (hashed) • Image: varchar (255) • City • Nickname • Fullname • Interest
Posts	Articles shared by users.	Allows users to share and read any blog posts. Users can manage their posts, including creating new ones, and deleting, and updating existing ones.	• Post ID • Title • Description: long text • Image: varchar (255) • Category • Creation Date: datetime • UserId: Foreign Key to id in Users table.
Search	The tool within the blog that allows users to find content.	Enables users to search posts by all, by region, by title, and by author.	Search Query. Filters (All, Region, Title, Author)

Comments	Responses or remarks that users can post in reaction to a blog post.	Engages the community by allowing users to make comments and reply to comments.	• CommentId • Description • CreatedAt • parentCommentId: Foreign key to comments Id. • postId: Foreign Key to Posts. • userId: Foreign Key to Users.
Likes	Serves as a simple feedback mechanism to show approval or support for content within the platform.	An interactive feature that allows users to express their appreciation for a blog post.	Like ID Post ID (Foreign Key to Posts) User ID (Foreign Key to Users)
Relationships	A subscription-based feature enabling users to stay updated on other users' posts.	After clicking the follow button, the user gains the ability to follow all posts authored by the selected user and view all posts from followed users in the personal service settings.	• Follower ID • followerUserId (Foreign Key to Users being followed) • FollowedUserid (Foreign Key to Users following)
Favorites	A personal collection feature that allows users to save posts to their favorites list.	Clicking the favorite icon enables users to save the post to their favorites list, where they can later view and manage these posts.	• Favorite ID • Post ID (Foreign Key to Posts) • User ID (Foreign Key to Users)
Theme	A subject matter tag used to classify blog posts.	Aids in content discovery by grouping posts by common topics such as 'Adventure', 'Beach', 'Family', etc.	• TagId • name: theme name • Supporting Data: Posts tagged with the theme
Region	A geographical categorization used to organize blog content.	Users can filter and discover blog content relevant to specific parts of the world, such as Asia, Europe, the Middle East, the Caribbean, or other regions they are interested in or planning to visit.	• Users can filter blog content relevant to specific parts of the world by utilizing the "category" column in the Posts table to query the database. • Supporting Data: Posts categorized with the region.
Travel-Info	A compilation of data and tools related to travel.	Provides users with real-time weather updates, a local search feature similar to Yelp for discovering services and places, and detailed country information to assist with travel planning.	• City Name • Weather Information • Local Services and Places • Country Information
AI-Chatbot	An intelligent digital assistant that aids users in their travel-related queries and tasks.	Offers automated responses to user inquiries, personalized travel suggestions, and itinerary planning assistance enhancing the overall travel experience.	• User Queries • Automated Responses • Personalized Suggestions • Itinerary Plans

2. Functional requirements V2

Priority Level	Description
1	Must have
2	Desired
3	Opportunistic

1. **User Login -- Priority: 1**
   - **1.1** Users enter their username and password into the provided login form.
   - **1.2** The system checks the entered username against its database to see if it exists. If the username doesn't exist, the system informs the user that the provided username is incorrect.
   - **1.3** The system compares the user's hashed password with the stored password. If the hashes match, the login is considered successful; otherwise, it's rejected and displays an error message to users.

2. **User Registration -- Priority: 1**
   - **2.1** Users provide information based on the registration form. If any input field does not meet the requirements, the system rejects the user and displays an error message.
   - **2.2** The registration system can validate user input as they type and show related error messages to ensure everything is accurate before submitting the form.
   - **2.3** If the registration is successful, users will be taken to the login page.

3. **User Profile Editing -- Priority: 1**
   - **3.1** Users have the ability to update their personal profile information, including email, city, nickname, full name, and interests. When updating the profile, the system will validate the data to ensure that only clean and appropriate data is sent to the database.
   - **3.2** Users can update profile images and will validate the type of uploaded file as well.
   - **3.3** Users can delete their accounts within the travel platform.

4. **User Profile Viewing -- Priority: 1**
   - **4.1** Users who click another author's image in a single post page, can view other authors' profiles as well.
   - **4.2** A Registered User can view their profile in a personal setting.
   - **4.3** Registered users who click the username in their followed post lists can view the following user's profile as well.

5. **Blog Post Creation -- Priority: 1**
    - **5.1** Registered users can create posts.
      To do so, they must accurately fill in all the fields, select the correct image type file, select a category, and choose one or more themes by checking the appropriate checkboxes.
    - **5.2** If the post belongs to the user, edit, and delete buttons will appear under the image. Otherwise, those buttons will not be displayed.
    - **5.3** The create post system will validate user input to ensure accuracy before submitting posts.

6. **Blog Post Editing -- Priority: 1**
    - **6.1** Registered users can edit their published posts. When editing, the form will first display all information from the previous post, allowing users to directly update the relevant content as needed.
    - **6.2** Registered users can edit post images as well.
    - **6.3** If the uploaded file is not an image, the system will not allow publication.
    - **6.4** The update post system will validate user all input to ensure accuracy before submitting posts.

7. **Browse the Posts by Theme or Region -- Priority: 1**
    - **7.1** Offer users the capability to browse travel-related posts organized by specific themes, including adventure, beach, family travel, etc., along with regions such as Asia, Europe, America, and more.

8. **Search Functionality -- Priority: 1**
    - **8.1** Users can easily find posts by searching through various criteria such as by all (any keywords), by region, by title, or by author.

9. **Country Information Functionality -- Priority: 3**
    - **9.1** Users input their country name to retrieve and display detailed country information, including language, currency, region, population, and other relevant data.

10. **Weather Information Update -- Priority: 3**
    - **10.1** Users input a city name to retrieve and display real-time weather information and a one-week forecast so that users can pack appropriately, and plan activities suited to the weather.

11. **Users' Interaction -- Priority: 2**
    - **11.1** Implement features for users to like, comment on, follow, and add favorite posts.

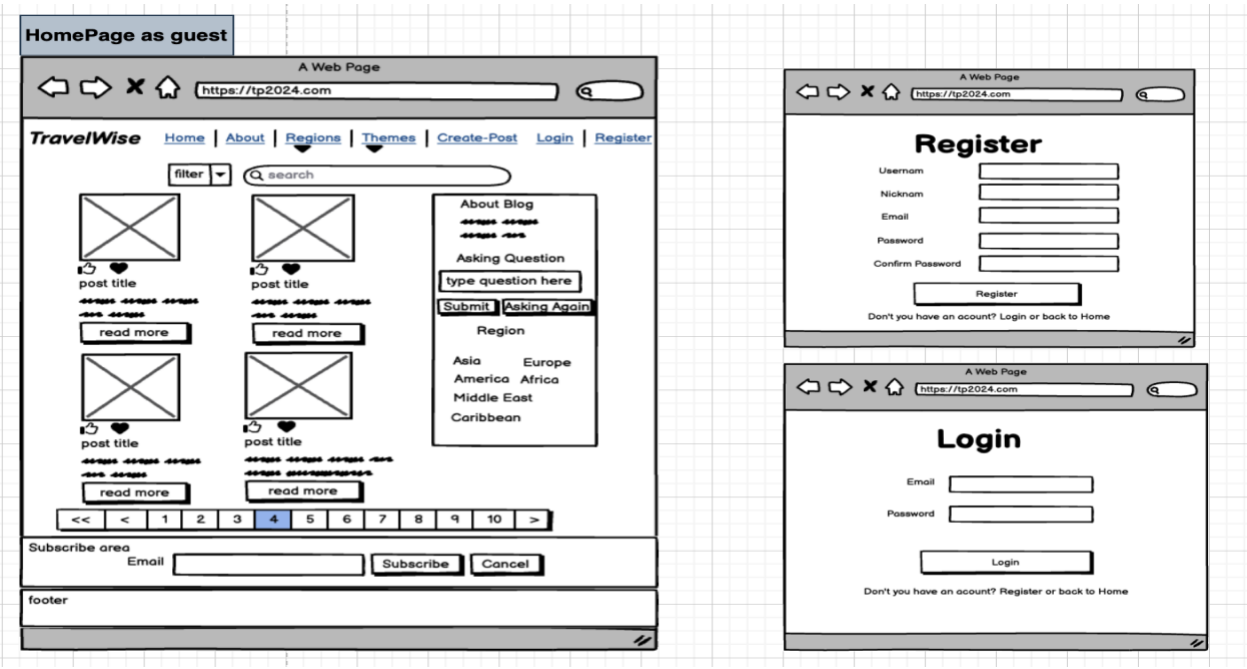
12. **View Personalized Content** -- **Priority: 2**
    - **12.1** Registered users can access and view their lists of published posts, with options to edit and delete their posts directly.
    - **12.2** Users should be able to view their lists of favorite posts in "*MY Favorites List"* in a dropdown menu.
    - **12.3** Users can view posts authored by users they are following in the "*Followed Users Posts"* dropdown menu.

13. **Manage Personalized Content** -- **Priority: 2**
    - **13.1** Registered users can manage their posts with the action to edit and delete directly from "*My Posts List"* in the dropdown menu.
    - **13.2** Users should be able to manage their lists of favorite posts with the action to delete from "My Favorites List".
14. **Yelp-like Search for Nearby Places** -- **Priority: 3**
    - **14.1** Users should be able to search for nearby places such as restaurants, cafes, and bars, and display the results sorted by ratings and category.

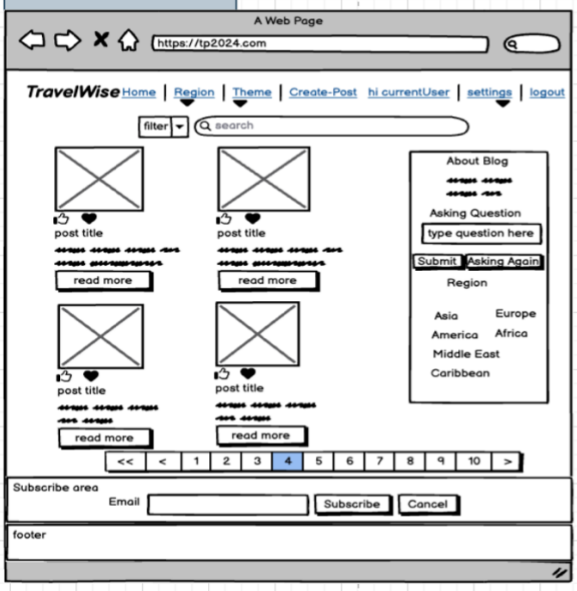
15. **AI-Powered Travel Chatbot** -- **Priority: 3**
    - **15.1** Users can ask any travel-related questions or seek assistance with their plans, receiving prompt automated responses for quick information and real-time support in travel planning.

3. UI Mockups and UX Flows

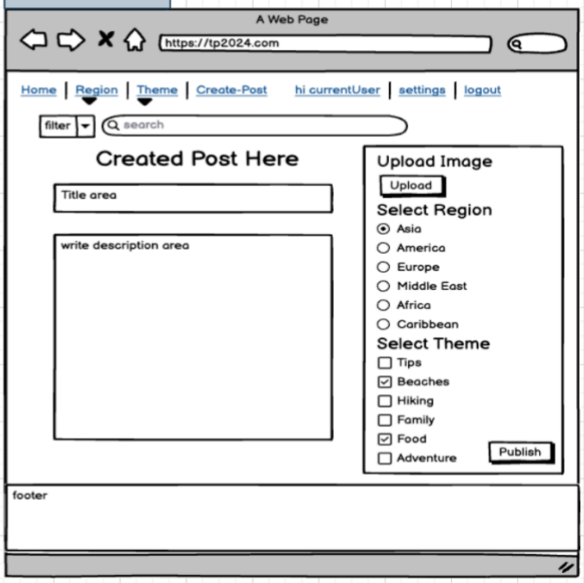
3.1 UI Mockups



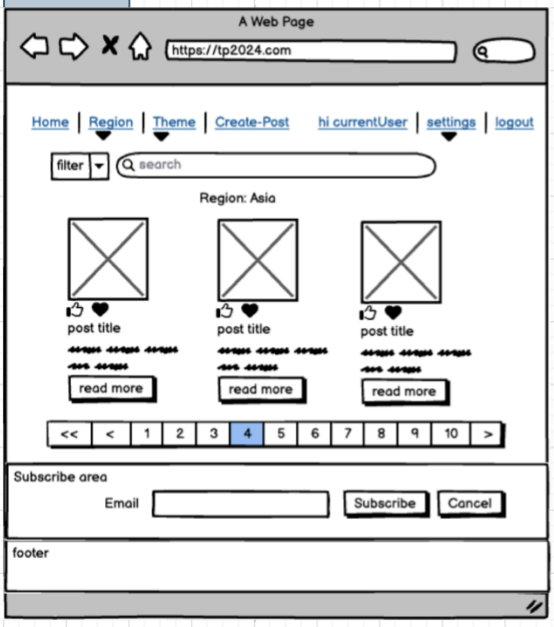
Logged in Home Page



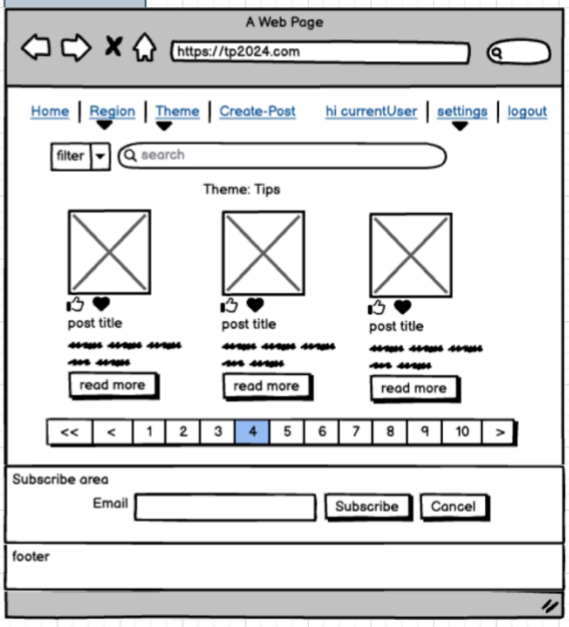
Creat Post page



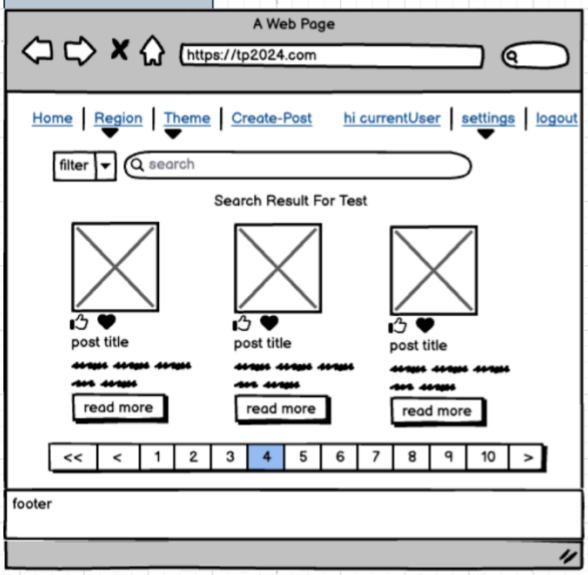
Region Page



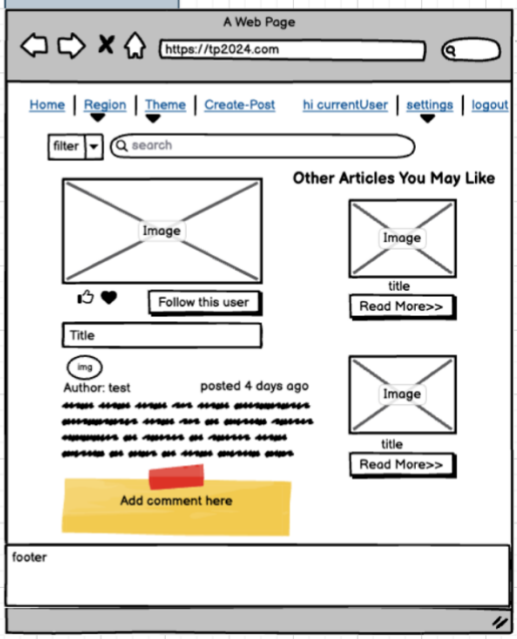
Theme Page



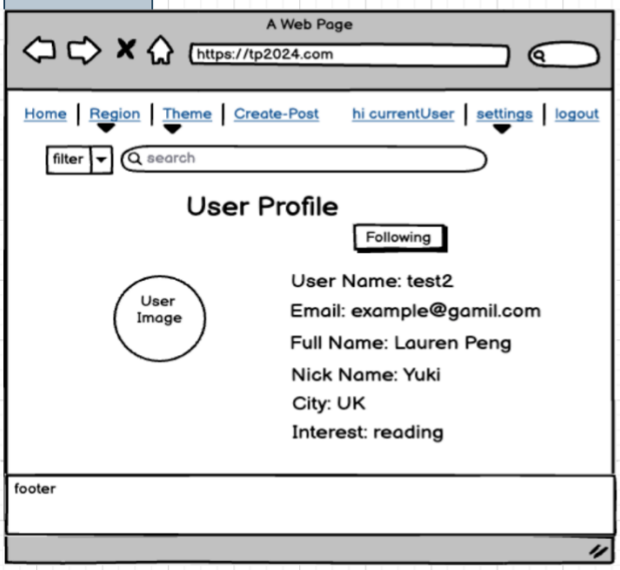
Search Result Page



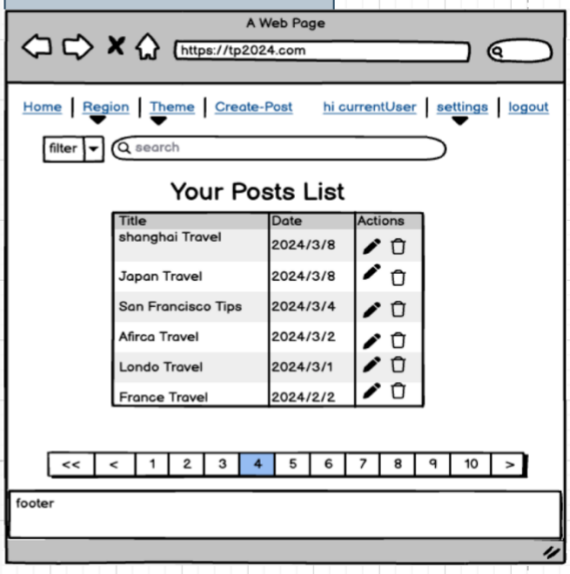
Single Post page

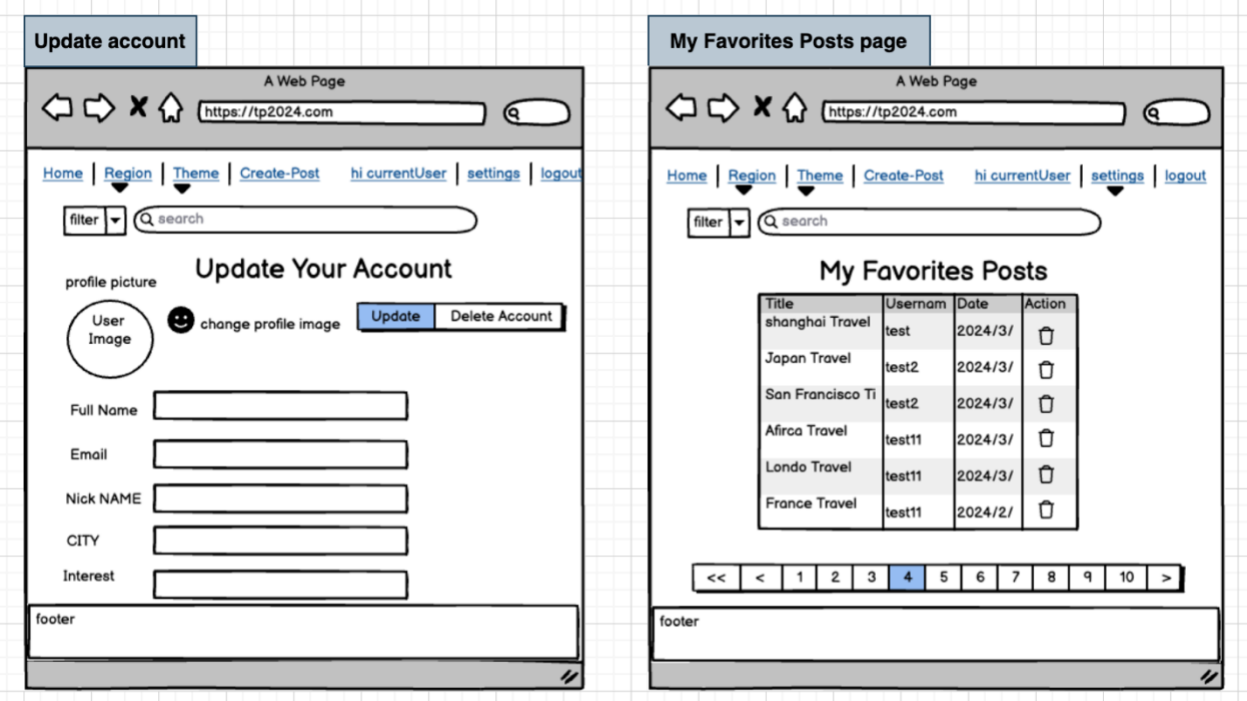


User Profile

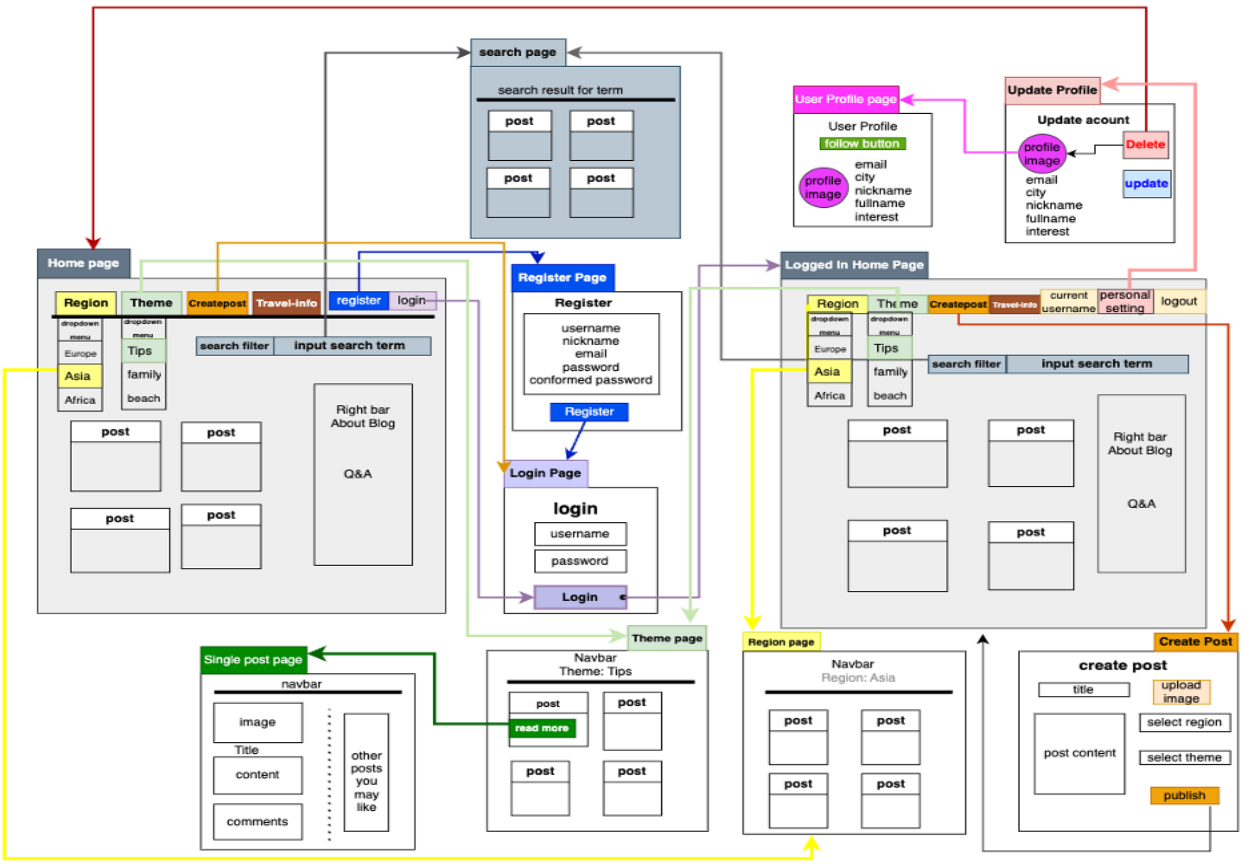


User own published all posts lists





3.2 Prototype UX Flows



3.3 Insights from Testing

The testing results of the travel blog website prototype have provided valuable insights into its user experience:

- **Useful:** The website effectively fulfills its purpose of providing useful travel information and tips, catering to the needs of travel enthusiasts.
- **Usability:** We found the website highly usable, with intuitive navigation and easy access to content.
- **Findable:** Information on the website is easy to find, indicating users can locate desired content efficiently. In addition, the search feature includes filters that allow users to narrow down their search results quickly, which makes it easier to find the desired content.
- **Desirability:** While the website's rough layout shows potential, the current design appears simple and not perfect. Further refinement is needed to enhance its desirability and create a stronger connection with users.
- **Credibility**: The fact that all posts are real stories from different users adds to the transparency and honesty of the content, which can enhance credibility. Additionally, user comments and feedback can further contribute to the credibility of the website, as they provide social proof and demonstrate active engagement with the audience.
- **Accessibility:** Accessibility is also a key focus, with the website being designed to be inclusive and usable by all, regardless of their abilities.

4. High-level Architecture, Database Organization

4.1 DB organization:

1. users		
Column	**Data Type**	**Description**
id (PK)	INT	Primary Key for user
img	Varchar(255)	Users' profile picture as image path
email	Varchar(255)	User's email
username	Varchar(45)	User register username, use it to login later.
password	Varchar(64)	User's hashed password
nickname	Varchar(45)	User's nick name
fullname	Varchar(45)	User's full name
interest	Varchar(255)	User's hobby or preference
city	Varchar(45)	User's city

2. posts		
Column	**Data Type**	**Description**
id (PK)	INT	Primary Key for post
title	Varchar(255)	Title of posts
desc	LONGTEXT	Content of posts
img	Varchar(255)	Image name of posts as string
thumbnail	Varchar(255)	Reduced size of image for posts
date	DATETIME	The posts created time
cat	Varchar(45)	Post's category,
uid (FK)	INT	a userId, a foreigner key referencing users.

3. comments table		
Column	**Data Type**	**Description**
id (PK)	INT	Primary Key for comment
desc	LONGTEXT	description
createdAt	DATETIME	comments created time
userId (FK)	INT	a foreign key referencing users.id
postId (FK)	INT	a foreign key referencing posts.id
parentCommentId (FK)	INT	a foreigner key referencing comment.id

4. favorites table		
Column	**Data Type**	**Description**
fid	INT	Primary Key for favorite (favorite id)
userId	INT	a foreign key referencing users.id
postId	INT	a foreign key referencing posts.id

5. likes table		
Column	**Data Type**	**Description**
id (PK)	INT	Primary Key for like
userId (FK)	INT	a foreign key referencing users.id
postId (FK)	INT	a foreign key referencing posts.id

6. relationships table		
Column	**Data Type**	**Description**
id (PK)	INT	Primary Key for relationship
followerUserId (FK)	INT	indicate yourself, current user. a foreign key referencing user.id
followedUserId (FK)	INT	indicate the person you are following. A foreign key referencing users.id

7. tags		
Column	**Data Type**	**Description**
tagId (PK)	INT	Primary Key for tag
name	Varchar(45)	indicate the theme name

8. posttags		
Column	**Data Type**	**Description**
pid (FK)	INT	means post id, a foreign key referencing post
tid (FK)	INT	means tag id, a foreign key referencing tag

9. subscriptions		
Column	**Data Type**	**Description**
id (PK)	INT	means post id, a foreign key referencing post
created_at	Data Time	subscriptions time
email	Varchar (255)	Users' email for subscribing.

4.2 Add/Delete/Search Architecture

Add Operations:

Users: New users can be added to the Users table.
Posts: New posts can be added to the Posts table.
Likes: Likes can be added to the Likes table.
Comments: New comments can be added to the comments table.
Favorites: The user's favorite posts can be added to the favorites table.
Relationships: Follower can be added to the relationship table.
Subscriptions: New subscribers can be added to the subscriptions table.

Search Operations:

Posts: Search for users' posts from the posts table and search by title, by author, by region, and by all.

Delete Operations:

Users: Users can be deleted from the Users table using their user ID.
Posts: Posts can be deleted from the posts table using the post ID and user ID.
Favorites: The user's favorite post can be removed from the favorites table using the user ID and post ID.
Likes: Likes can be deleted from the likes table using the user ID and post ID.
Relationship: The follower relationship can be deleted from the relationship using the followerUser ID and followedUser ID.
Subscriptions: Cancel subscriptions can use the delete operation.

Display Operations:

Users: Display user profile.
Posts: Display all posts or display search result posts.
Comments: Display all comments for each post.
Likes: Display the likes count for each post.
Favorites: Display a list of favorite posts for a user.
Relationship: Display the following status.

4.3 APIs

The application's architecture includes several RESTful APIs that allow for communication between the front and backend services and integration with third-party API services.
Own Backend & Frontend APIs include,
Users, Posts, Likes, Comments, Relationships, Favorites, Search, Authentication.

	HTTP Method	Request	Responses 200 (or 500 {message: "err..."}	Route Function
Post API	POST	title, desc, img, thumbnail, cat, date, selectedTheme	{message}	addPost
	DELETE	userId, postId	{message}	deletePost
	PUT	title, desc, img, thumbnail, cat, selectedTheme	{message}	updatePost
	GET	postId	{title, desc, img, date, uid, cat, tags, username}	getPost for getting single post.
	GET	id	{id, title, desc, img, thumbnail, date, uid, cat}	getPosts

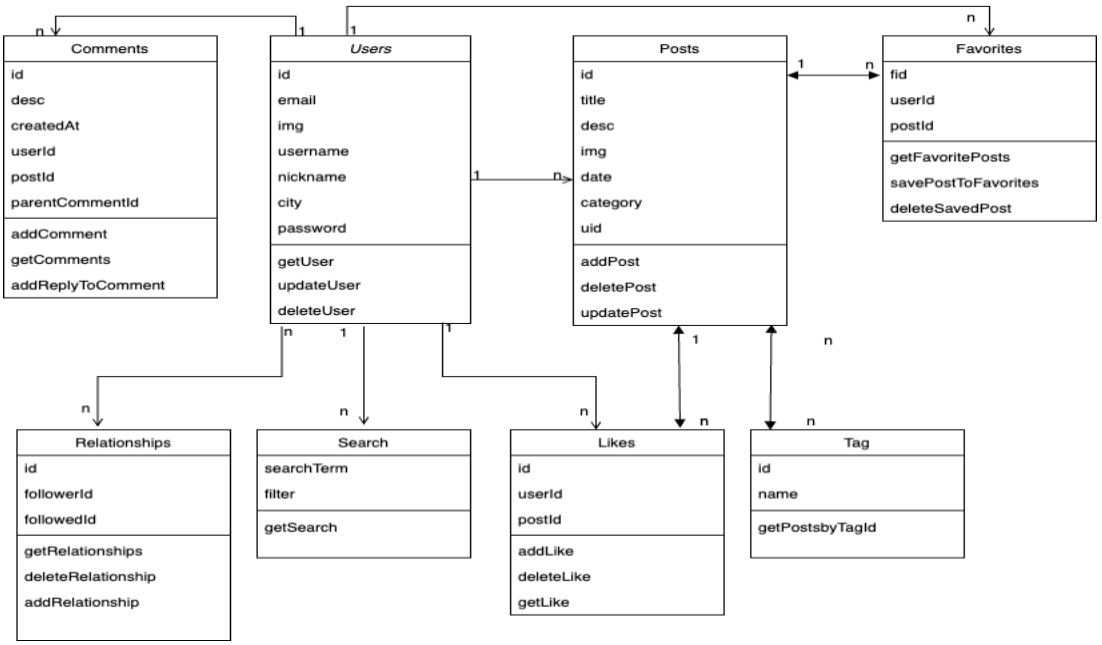
Users API	GET	userId	{id, username, email, city, img, nickname} (user object)	getUser
	PUT	userId	{message}	updateUser
	DELETE	userId	{message}	deleteUser
	POST	username, email, password, nickname	{message}	createUser
Likes API	GET	postId	list of userId	getLikes
	POST	userId, postId	{message}	addLike
	DELETE	userId, postId	{message}	deleteLike
Comments API	GET	postId	list of objects	getComments
	POST	desc, postId	{message}	addComment
	POST	userId, desc, parentCommentId, postId	{message}	addReplyComment
Favorites API	GET	userId	{data: [{post}, {post}], totalPages}	getFavoritePosts
	POST	userId, postId	{message}	savePostToFavorites
	DELETE	userId, postId	{message}	deleteSavedPost
Authentication API	POST	username, password	{message}	login
	POST		{message}	logout
	POST	email, username, password, nickname	{message}	register
Relationship API	GET	followedUserId	{data: [{post}, {post}], totalPages}	getFollowerPosts
	POST	followedUserId, followerUserId	{message}	addRelationship
	DELETE	followedUserId, followerUserId	{message}	deleteRelationship
Search API	GET	searchTerm, filter	{data: [{post}, {post}], total, totalPages}	getSearch

3rd Party API:

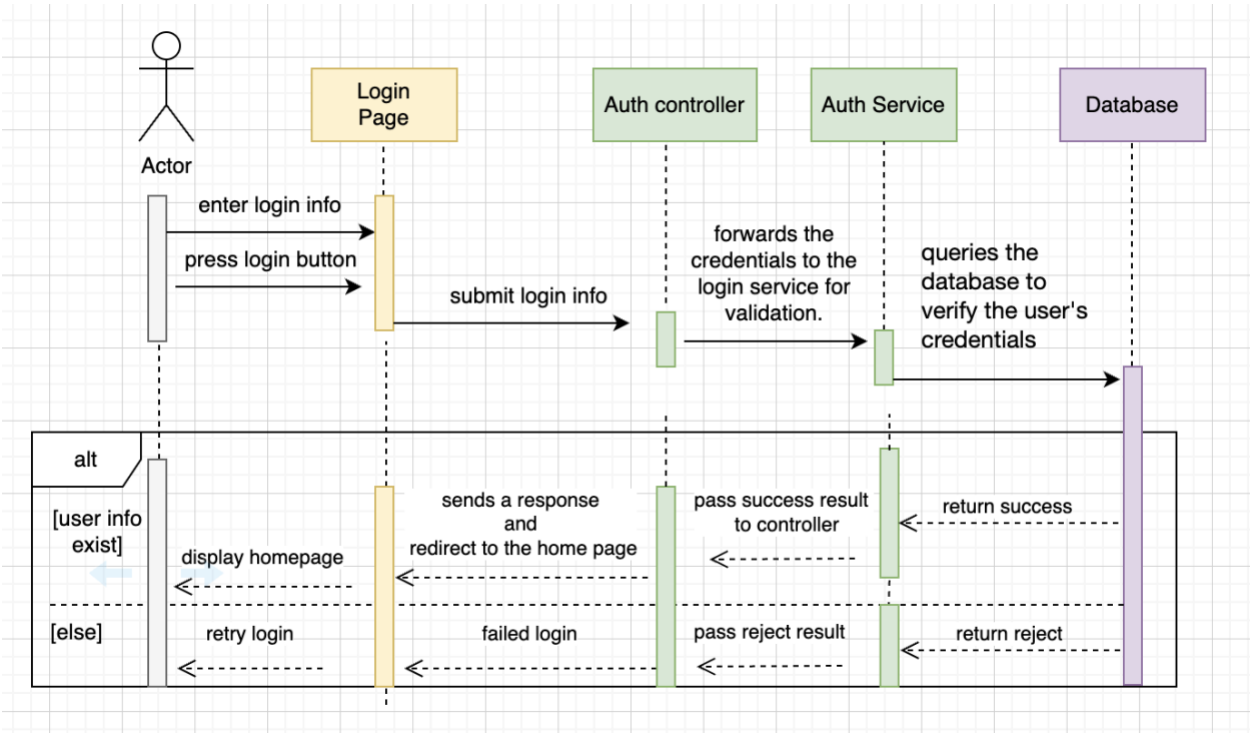
- **OpenWeather API** for real-time weather information.
- **Yelp Business API** for accessing business reviews and details.
- **OpenAI API** for leveraging AI-driven content generation and processing.
- **Geo API** for implementing geolocation services.
- **Country API** for retrieving detailed country-specific data.
- **Server-Side Events Module**: GitHub: mpetazzoni/sse.js.

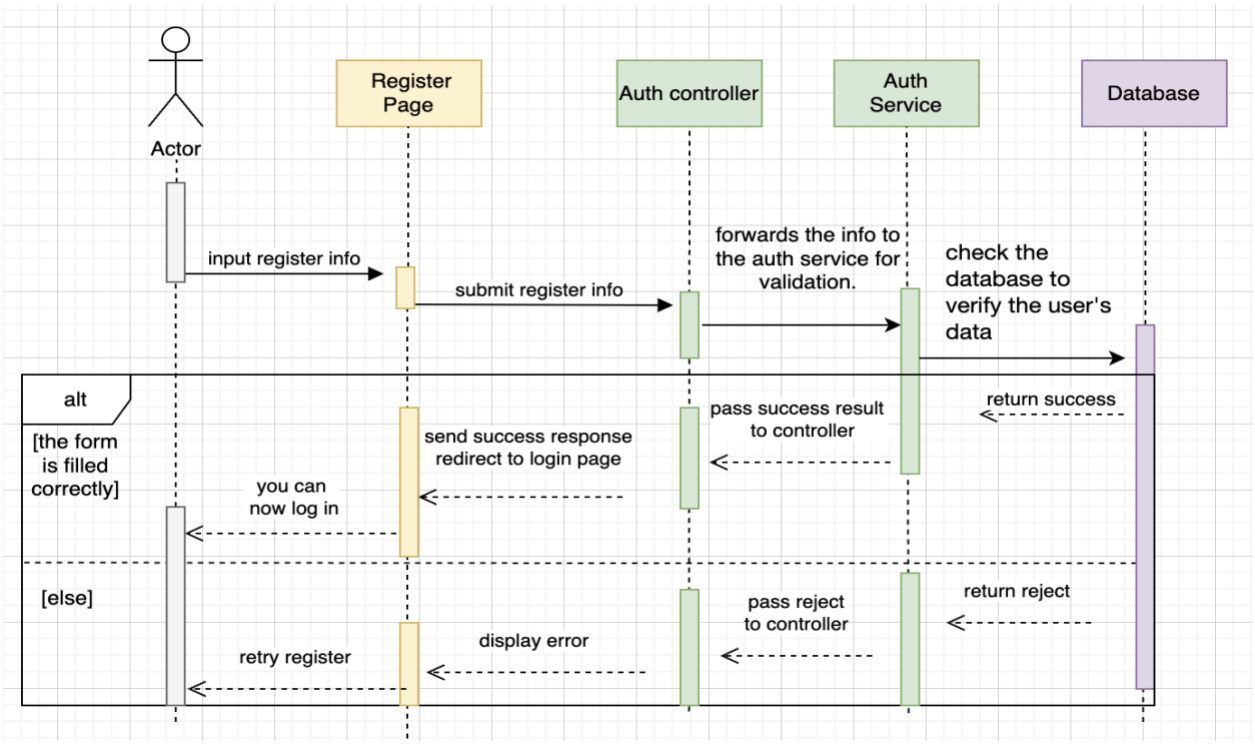
5. High-Level UML Diagrams

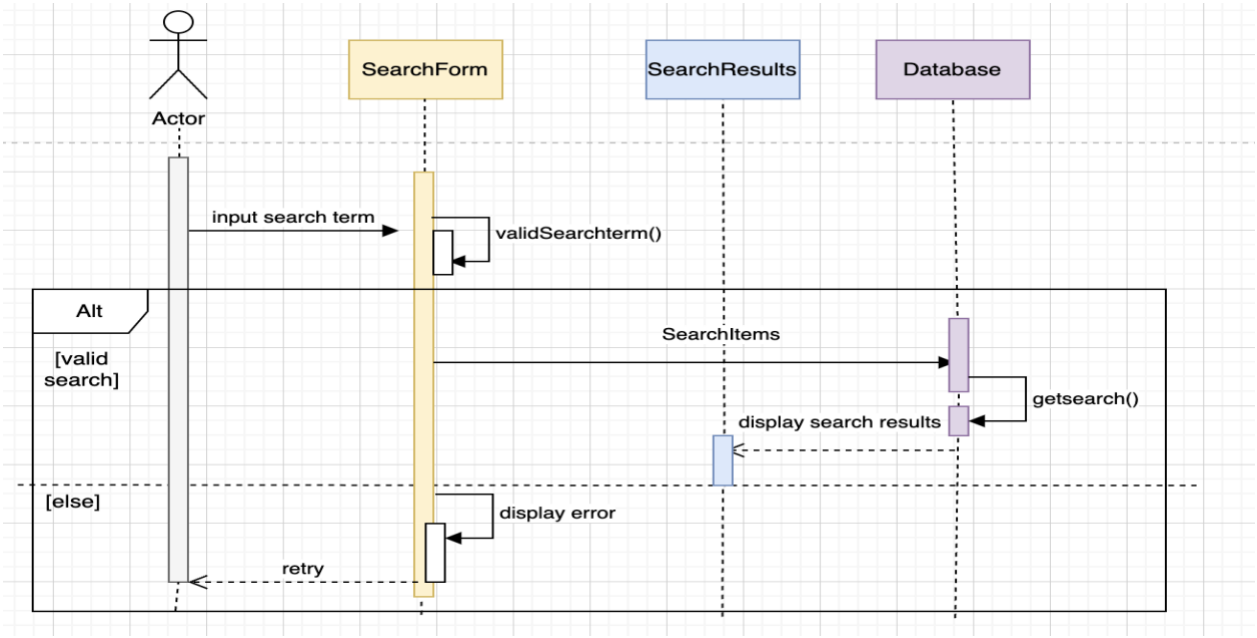
5.1 High-level UML Class diagrams

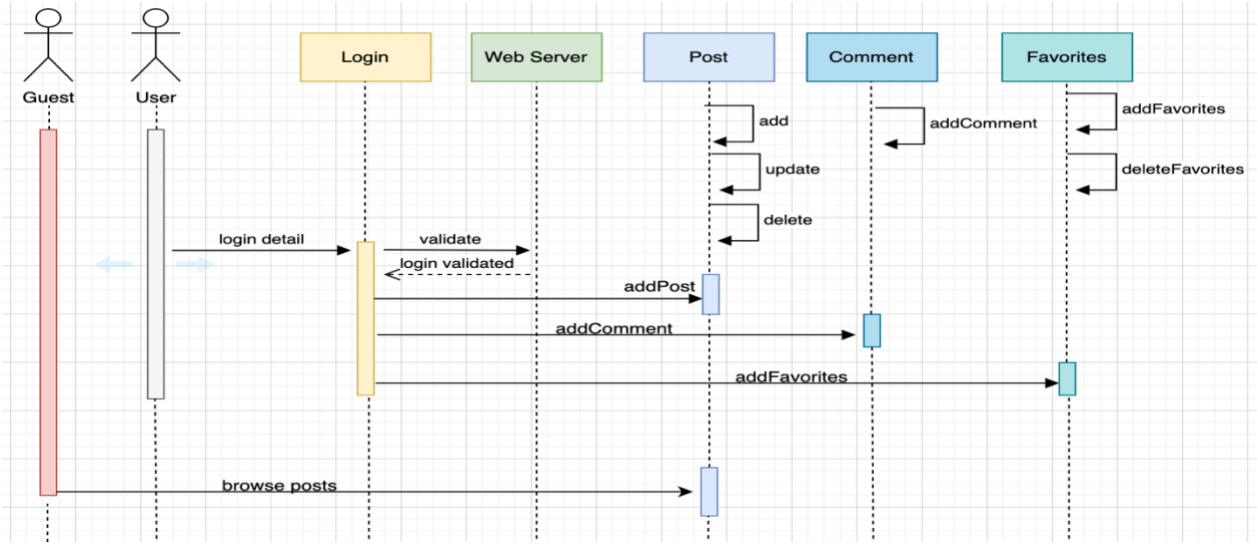


5.2 High-level Sequence diagrams




Actor
Register Page
Auth controller
Auth Service
Database
input register info
submit register info
forwards the info to the auth service for validation.
check the database to verify the user's data
alt
[the form is filled correctly]
send success response redirect to login page
pass success result to controller
return success
you can now log in
[else]
pass reject to controller
return reject
display error
retry register


Actor
SearchForm
SearchResults
Database
input search term
validSearchterm()
Alt
[valid search]
SearchItems
getsearch()
display search results
[else]
display error
retry


Guest
User
Login
Web Server
Post
Comment
Favorites
add
addComment
addFavorites
update
deleteFavorites
delete
login detail
validate
login validated
addPost
addComment
addFavorites
browse posts