SD^2: Software Design Document

The purpose of the software design document (SD^2) is to provide an overview of the system and to help people understand the system. This document can be used by a programmer as a guideline for implementing the design without needing to make significant engineering design decisions, and will be provided to the customer as a final deliverable.

Link to repo

https://github.com/Ike44/OnTheGO

1. Project Overview

   This section gives context for the reader so that they can understand the design that is being presented. It should describe the problem that your software attempts to address, identifying stakeholders and their stake in the system, and how your software will address the problem. This section gives context for the reader so that they can understand the design that is being presented. It should describe the problem that your software attempts to address, identifying stakeholders and their stake in the system, and how your software will address the problem. In this section provide the General Model including (Context Diagram and Use-case Diagram), user stories, product backlog as well as the main use-cases presenting the technical details based on requirement engineering.

   Our OnTheGO! application is a dynamic application designed for travelers seeking authentic travel insights and business owners looking to promote their services. The application addresses two key challenges which first is the difficulty travelers face in finding personalized, community-driven recommendations and the need for businesses to engage directly with a targeted audience. Modern travelers often struggle with generic, unauthentic platforms that lack peer-to-peer interaction and fail to offer seamless trip planning advice or realistic experiences . Also, small and medium-sized businesses lack accessible and user-friendly ways to promote their services to travelers in specific regions.

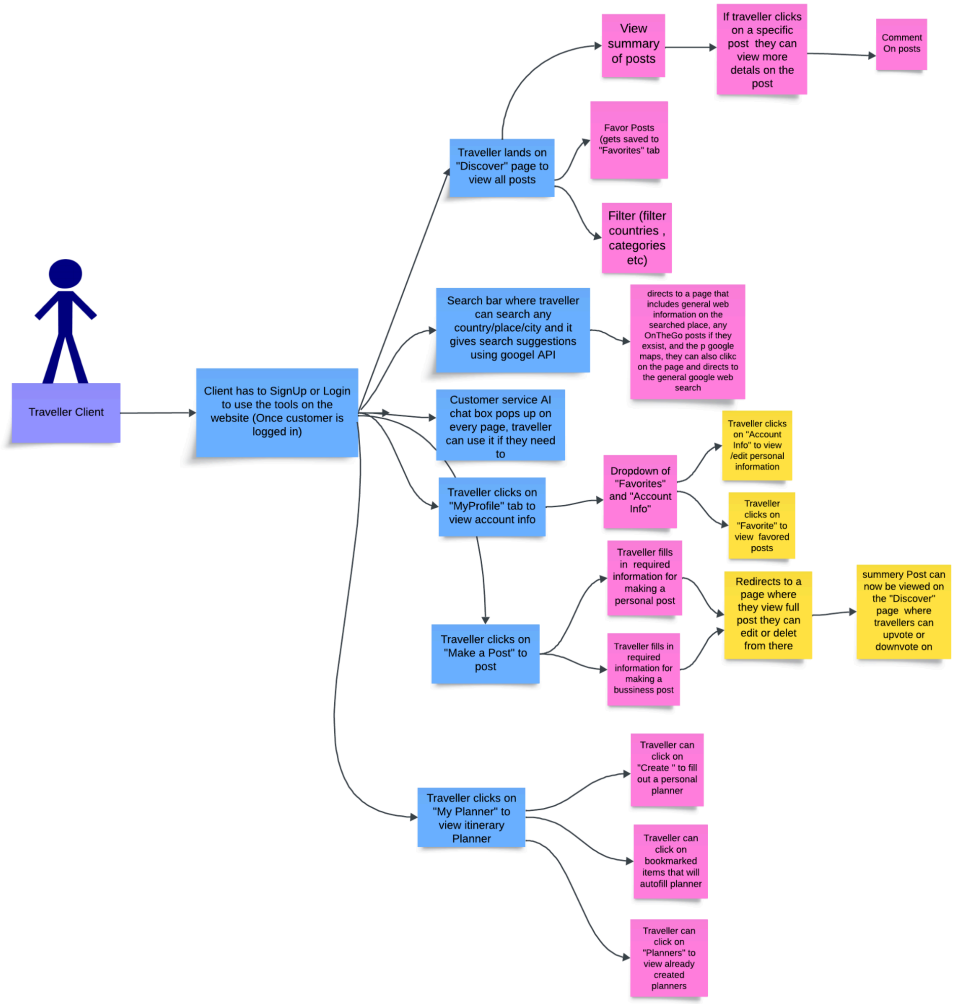
   OnTheGo! provides a unified solution by offering features thar reflect these needs. Travelers can create, view, edit, and delete posts about their experiences in different cities and countries, as well as interact with others through comments, upvotes, and bookmarks. A powerful search bar, integrated with Google's Search Suggestions API, enables users to quickly discover places of interest and access detailed information, including reviews, contact details, hours of operation, and

Google Maps. This results page also includes filtered posts related to their search and they are also able to access the general web search results associated with their search. Filters on the Discover page also enhances personalization, allowing users to sort content by ratings, categories, locations, and other criteria.

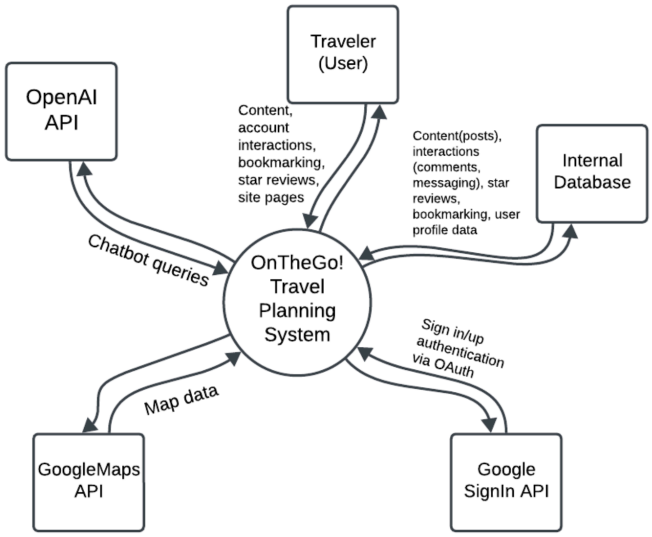
The platform also includes a customizable planner feature that lets travelers create and modify itineraries. An upcoming enhancement will enable users to auto-fill their planners using bookmarked posts, which simplifies the planning process. Additionally, businesses can promote their services by leveraging posts relevant to specific destinations. To further enhance user experience, OnTheGo is integrating an AI-powered customer service chatbot, providing instant assistance on every page.

By combining community-driven content, authentic personal experiences, seamless trip planning, and targeted promotional opportunities, OnTheGo creates a comprehensive platform for exploration, connection, and growth. It serves as a hub where travelers can find trustworthy insights, plan memorable trips, and connect with other adventurers, and businesses gain valuable exposure to the right targets.

Context Diagram



Use Case Diagram



User stories

OnTheGo!

Author(s): Isabella Ochsner, Raneem Salameh, David Taylor, Isaac Amanor, Kyra Atkinson

Date: **Oct 6, 2024**

Version: 0.0.1

USE CASE NAME:	**Sign In/Sign Up**	
USE CASE ID:	00001	**USE CASE TYPE** Business Requirements: ☐ System Analysis: ☐
PRIORITY:	Low	
SOURCE:	OnTheGo! Design Specifications Team Research Industry Standards	
PRIMARY BUSINESS ACTOR	Travelers	
PRIMARY SYSTEM ACTOR	OnTheGo! Application	
OTHER PARTICIPATING ACTORS:	• For Business: • Business trying to promote their restaurant, activities, transportation etc • Traveler influencers • Tour Guide Companies etc • For System: • Authentication Services (Google)	
OTHER INTERESTED STAKEHOLDERS:	• Business Owners • Airlines • Traveler Agents • International Businesses trying to promote their restaurant, activities, transportation etc	

DESCRIPTION:	When a traveler opens the OnTheGo! application, they are prompted to sign in or sign up before accessing any features. The traveler may either use an existing account to sign in or create a new account where the system requires user to enter information such as email, username, password, name etc. The sign-in ensures user authentication and allows access to personalized features.	
PRE-CONDITION:	The application is opened on the web browser, and the traveler has either opened the website for the first time or logged out from a previous session.	
TRIGGER:	The traveler launches the OnTheGo**!** website .	
TYPICAL COURSE OF EVENTS:	**Actor Action**	**System Response**
	Step 1:	**Step 2**:
	The traveler launches the OnTheGo! application.	The application displays a pop-up window with options to Sign In or Sign Up.
	The traveler selects either Sign In or Sign Up.	If Sign In is selected, the system prompts the traveler for their email and password and authenticates them.If Sign Up is selected, the system collects the traveler's details (name, email, password) and creates a new account.
	The traveler enters the required information	The system verifies the information
	The traveler is successfully signed	Now traveler can use all tools of the website
ALTERNATE COURSES:	Traveler enters incorrect sign-in credentials | System shows an error message and prompts the traveler to re-enter the correct credentials or use the "Forgot Password" option.	
CONCLUSION:	The traveler is successfully authenticated and can now access the app's main features.	
POST-CONDITION:	The traveler's session is active, and they can interact with the app's tools and services.	
BUSINESS RULES	• The system must enforce secure password policies (e.g., minimum length, special characters). • The system must ensure that email addresses are unique during account creation. • Traveler can only view main tools on the website is logged in	
IMPLEMENTATION CONSTRAINTS AND SPECIFICATIONS	• The system must be integrated with a secure authentication service. • MongoDB for saving sessions • Frontend/UI development with responsive design • Backend development • Session management/user authentication	
ASSUMPTIONS:	• Travelers will have access to valid credentials (email and password). • The system's sign-up process will require minimal input (email, password, and personal information).	

OPEN ISSUES:	None at this time

USE CASE NAME:	**View Discover Page/View Posts**	**USE CASE TYPE** Business Requirements: □ **System Analysis:** □
USE CASE ID:	00002	
PRIORITY:	High	
SOURCE:	OnTheGo! Design Specifications Team Research Industry Standards	
PRIMARY BUSINESS ACTOR	Travelers	
PRIMARY SYSTEM ACTOR	OnTheGo! Application	
OTHER PARTICIPATING ACTORS:	• For Business: • Business trying to promote their restaurant, activities, transportation etc • Traveler influencers • Tour Guide Companies etc	
OTHER INTERESTED STAKEHOLDERS:	• Airlines • Traveler Agents • International Businesses trying to promote their restaurant, activities, transportation etc	
DESCRIPTION:	A traveler may be interested in browsing posts on the discover page in efforts to influence their travel priorities	
PRE-CONDITION:	Traveler must be logged into OnTheGo! website	
TRIGGER:	Once Traveler signs in, they land on the Discover page to view all posts If already logged in then Traveler can click on the Discover tab in the Nav bar	
TYPICAL COURSE OF EVENTS:	**Actor Action**	**System Response**
	Step 1:	**Step 2**:
	Traveler Signs In	System redirects traveler to Discover page
ALTERNATE COURSES:	Traveler clicks Discover tab in Navbar	
	System redirects traveler to Discover page	
CONCLUSION:	Traveler successfully is able to browse posts on Discover page	
POST-CONDITION:	Traveler succefully views the posts on the Discover page and is able to interact with the post (bookmark and comment)	
BUSINESS RULES	• User must be signed in to view or interact with the posts on the Discover page	
IMPLEMENTATION CONSTRAINTS AND SPECIFICATIONS	• Session management/user authentication • MongoDB • Frontend/UI development with responsive design • Backend development • Session management/user authentication	
ASSUMPTIONS:	Traveler is signed into the application	
OPEN ISSUES:	None at this time	

USE CASE NAME:	**Comment on Posts**	**USE CASE TYPE** Business Requirements: □ System Analysis: □
USE CASE ID:	00003	
PRIORITY:	Medium	
SOURCE:	OnTheGo! Design Specifications Team Research Industry Standards	
PRIMARY BUSINESS ACTOR	Travelers	
PRIMARY SYSTEM ACTOR	OnTheGo! Application	
OTHER PARTICIPATING ACTORS:	• For Business: • Business trying to promote their restaurant, activities, transportation etc • Traveler influencers • Tour Guide Companies etc	
OTHER INTERESTED STAKEHOLDERS:	• Airlines • Traveler Agents • International Businesses trying to promote their restaurant, activities, transportation etc	
DESCRIPTION:	Travelers are able to comment on posts located on the Discover page to share their experiences, ask questions, or interact with other users. Businesses and influencers can also engage in the comment section to promote their services or share additional information.	
PRE-CONDITION:	The traveler must be logged into the OnTheGo! application. There must be posts on the Discover page.	
TRIGGER:	The traveler views a post on the Discover page and chooses to add a comment.	
TYPICAL COURSE OF EVENTS:	**Actor Action**	**System Response**
	Step 1:	**Step 2**:
	The traveler logs into the **OnTheGo!** application.	The system validates the traveler's session to confirm they are logged in.
	The traveler navigates to the Discover page either through the "Discover" tab. Or by simply being on thelanding page.	System displays Discover page where all posts can be viewed
	The traveler navigates to a post they are interested in and clicks on the comment section and types their comment.	The system processes and saves the comment associated with the post.
ALTERNATE COURSES:	The traveler attempts to comment without being logged in and System redirects the traveler to the login/sign-up page.	
CONCLUSION:	The traveler's comment is successfully added to the post, and other users can view it and engage with it.	
POST-CONDITION:	The comment is stored in the system database and linked to the specific post. The comment is visible to the traveler and other users.	

BUSINESS RULES	• Users must be authenticated to comment on posts. • Comments must follow community guidelines to avoid inappropriate content.
IMPLEMENTATION CONSTRAINTS AND SPECIFICATIONS	• System uses MongoDB to store sessions • Comments must be scalable to handle large volumes of users comments • Comments should be stored in the database thats linked to the specific post ID • Frontend/UI development with responsive design • Backend development • Session management/user authentication
ASSUMPTIONS:	• Users are familiar with commenting systems • User must be logged in
OPEN ISSUES:	None at this time

USE CASE NAME:	**Bookmark Posts**	**USE CASE TYPE** Business Requirements: □ System Analysis: □
USE CASE ID:	00004	
PRIORITY:	Medium	
SOURCE:	OnTheGo! Design Specifications Team Research Industry Standards	
PRIMARY BUSINESS ACTOR	Travelers	
PRIMARY SYSTEM ACTOR	OnTheGo! Application	
OTHER PARTICIPATING ACTORS:	• For Business: • Business trying to promote their restaurant, activities, transportation etc • Traveler influencers • Tour Guide Companies etc	
OTHER INTERESTED STAKEHOLDERS:	• Airlines • Traveler Agents • International Businesses trying to promote their restaurant, activities, transportation etc	
DESCRIPTION:	When a traveler views a post on the Discover page, they have the option to bookmark the post by clicking on a Bookmark button. Once bookmarked, the post is saved to the Favorites section under the MyProfile tab in the navigation bar. The traveler can access this tab at any time to view or manage their saved posts. This feature allows users to save posts for future reference.	
PRE-CONDITION:	The traveler must be logged into the OnTheGo! application. The post must be accessible on the Discover page. The Favorites tab must exist under the MyProfile section.	
TRIGGER:	The traveler clicks the Bookmark button on a post they want to save	
TYPICAL COURSE OF EVENTS:	**Actor Action**	**System Response**
	Step 1:	**Step 2**:
	The traveler logs into the OnTheGo! application.	The system validates the traveler's session to confirm they are logged in.

	The traveler navigates to the Discover page.	
	The traveler navigates to a post they want to bookmark.	
	The traveler clicks the Bookmark button.	The system saves the post to the traveler's Favorites section under the MyProfile tab.
ALTERNATE COURSES:	The traveler is not logged in and attempts to bookmark a post.The system redirects the traveler to the login/sign-up page.	
CONCLUSION:	The post is successfully bookmarked, and the traveler can access it from the Favorites section under the MyProfile tab.	
POST-CONDITION:	The bookmarked post is stored in the system database and linked to the traveler's account. The post is visible in the Favorites tab under MyProfile for future access.	
BUSINESS RULES	• Users must be logged in to bookmark posts. • Each post can be bookmarked only once per user. • Bookmarks must be easily retrievable under the Favorites section in the MyProfile tab.	
IMPLEMENTATION CONSTRAINTS AND SPECIFICATIONS	• The system must use MongoDB for storing session and user activity data. • Backend and Frontend is required to make function happen • The bookmarked posts should be stored as a reference in the user's profile • Frontend/UI development with responsive design • Backend development • Session management/user authentication	
ASSUMPTIONS:	• Travelers wants to bookmark posts for future reference • Travelers are familiar with the concept of bookmarking • Tarveler is logged in	
OPEN ISSUES:	None at this time	

USE CASE NAME:	**Filter Posts**	**USE CASE TYPE** Business Requirements: □ System Analysis: □
USE CASE ID:	00005	
PRIORITY:	Medium	
SOURCE:	OnTheGo! Design Specifications Team Research Industry Standards	
PRIMARY BUSINESS ACTOR	Travelers	
PRIMARY SYSTEM ACTOR	OnTheGo! Application	

OTHER PARTICIPATING ACTORS:	• For Business: • Business trying to promote their restaurant, activities, transportation etc • Traveler influencers • Tour Guide Companies etc	
OTHER INTERESTED STAKEHOLDERS:	• Airlines • Traveler Agents • International Businesses trying to promote their restaurant, activities, transportation etc	
DESCRIPTION:	The filter feature on the Discover page allows travelers to filtr the posts they see by selecting specific criteria such as ratings, categories (restaurants, hidden gems), and countries etc.This feature helps users discover content that is most relevant to their interests or travel plans.	
PRE-CONDITION:	The traveler must be logged into the **OnTheGo!** application. Posts must be available on the Discover page. Filters for ratings, categories, and countries must be predefined in the system.	
TRIGGER:	The traveler clicks on the filter option on the Discover page and selects the criteria to filter posts.	
TYPICAL COURSE OF EVENTS:	**Actor Action**	**System Response**
	Step 1:	**Step 2**:
	The traveler logs into the OnTheGo! application.	
	The traveler navigates to the Discover page.	
	The traveler clicks the Filter button to open the filtering options.	
	The traveler selects one or more filter criteria (e.g., high-rated posts, posts from Italy).	The system retrieves and displays posts that match the selected filter criteria.
	The traveler clicks the Apply Filters button	The system updates the displayed posts on the Discover page based on filters selected.
ALTERNATE COURSES:	The traveler selects filter criteria, but no posts match the chosen criteria.The system displays a message indicating no posts were found .	
	The traveler attempts to filter posts without logging in.The system prompts the traveler to log in to use the filter feature.	
CONCLUSION:	The traveler successfully filters posts based on their selected criteria.	

POST-CONDITION:	The filtered posts are displayed based on the traveler's selected criteria. The system updates the traveler's Discover page to reflect the applied filters.
BUSINESS RULES	• Travelers must be authenticated to use the filter feature. • Only posts that meet the selected filter criteria should be displayed. • Filter options should be easy to use • Filter option should provide real-time updates to the Discover page.
IMPLEMENTATION CONSTRAINTS AND SPECIFICATIONS	• The system must use MongoDB for storing and retrieving user session data. • The filtering mechanism must be able to handle multiple filter criteria • Filters must be dynamically updated based on the availability of posts • Frontend/UI development with responsive design • Backend development • Session management/user authentication
ASSUMPTIONS:	• Travelers are familiar with using filters to refine search result • There will be a sufficient number of posts to support filtering options across multiple categories and countries
OPEN ISSUES:	None at this time

USE CASE NAME:	**Click/Hover over MyProfile**	**USE CASE TYPE** Business Requirements: □ System Analysis: □
USE CASE ID:	00006	
PRIORITY:	High	
SOURCE:	OnTheGo! Design Specifications Team Research Industry Standards	
PRIMARY BUSINESS ACTOR	Travelers	
PRIMARY SYSTEM ACTOR	OnTheGo! Application	
OTHER PARTICIPATING ACTORS:	• For Business: • Business trying to promote their restaurant, activities, transportation etc • Traveler influencers • Tour Guide Companies etc	
OTHER INTERESTED STAKEHOLDERS:	• Airlines • Traveler Agents • International Businesses trying to promote their restaurant, activities, transportation etc	
DESCRIPTION:	A traveler may be interested in the options hovering over or clicking my profile offers, they may want to view their favorites or their account information.	
PRE-CONDITION:	Traveler must be logged into OnTheGo! application.	
TRIGGER:	Once the traveler signs in, they land on the Discover page, they can navigate to the MyProfile icon and hover over/click it.	
TYPICAL COURSE OF EVENTS:	**Actor Action**	**System Response**
	Step 1:	**Step 2**:
	Traveler hovers over MyProfile icon in the navbar	System displays "Account Info" and "Favorites" options through dropdown
ALTERNATE COURSES:	Traveler clicks MyProfile icon	
	System displays "Account Info" and "Favorites" options through dropdown	

CONCLUSION:	Traveler successfully views MyProfile options
POST-CONDITION:	System successfully displays MyProfile options
BUSINESS RULES	• User must be signed in to view MyProfile options
IMPLEMENTATION CONSTRAINTS AND SPECIFICATIONS	• Frontend/UI development with responsive design • Backend development • Session management/user authentication
ASSUMPTIONS:	User is signed into the application
OPEN ISSUES:	None at this time

USE CASE NAME:	**View Account Info**	**USE CASE TYPE** Business Requirements: □ System Analysis: □
USE CASE ID:	00007	
PRIORITY:	Medium	
SOURCE:	OnTheGo! Design Specifications Team Research Industry Standards	
PRIMARY BUSINESS ACTOR	Travelers	
PRIMARY SYSTEM ACTOR	OnTheGo! Application	
OTHER PARTICIPATING ACTORS:	• For Business: • Business trying to promote their restaurant, activities, transportation etc • Traveler influencers • Tour Guide Companies etc	
OTHER INTERESTED STAKEHOLDERS:	• Airlines • Traveler Agents • International Businesses trying to promote their restaurant, activities, transportation etc	
DESCRIPTION:	A traveler may be interested in viewing their personal information on their account.	
PRE-CONDITION:	Traveler must be logged into OnTheGo! application and has either clicked or hovered over the myProfile icon to view options.	
TRIGGER:	Traveler clicks “Account Info”	
TYPICAL COURSE OF EVENTS:	**Actor Action**	**System Response**
	Step 1:	**Step 2**:
	Traveler clicks “Account Info”	System redirects traveler to Account Information page
ALTERNATE COURSES:	Traveler clicks out of page and is redirected.	
CONCLUSION:	Traveler successfully is able to view their account information	
POST-CONDITION:	System displays traveler’s Account information page	
BUSINESS RULES	• User must be signed in to view their account information	
IMPLEMENTATION CONSTRAINTS AND SPECIFICATIONS	• Frontend/UI development with responsive design • Backend development • Session management/user authentication • MongoDB for data retrieval	
ASSUMPTIONS:	User is signed into the application	
OPEN ISSUES:	None at this time	

USE CASE NAME:	View Favorites	USE CASE TYPE Business Requirements: □ System Analysis: □
USE CASE ID:	00008	
PRIORITY:	Medium	
SOURCE:	OnTheGo! Design Specifications Team Research Industry Standards	
PRIMARY BUSINESS ACTOR	Travelers	
PRIMARY SYSTEM ACTOR	OnTheGo! Application	
OTHER PARTICIPATING ACTORS:	• For Business: • Business trying to promote their restaurant, activities, transportation etc • Traveler influencers • Tour Guide Companies etc	
OTHER INTERESTED STAKEHOLDERS:	• Airlines • Traveler Agents • International Businesses trying to promote their restaurant, activities, transportation etc	
DESCRIPTION:	A traveler may be interested in viewing the posts they have bookmarked.	
PRE-CONDITION:	Traveler must be logged into OnTheGo! application and as either clicked or hovered over the myProfile icon to view options.	
TRIGGER:	Traveler clicks “Favorites” (Or “Bookmarks”, same functionality)	
TYPICAL COURSE OF EVENTS:	**Actor Action**	**System Response**
	Step 1:	**Step 2**:
	Traveler clicks “Favorites”	System redirects traveler to Favorites page
ALTERNATE COURSES:	Traveler clicks out of page and is redirected.	
CONCLUSION:	Traveler successfully is able to view their favorited posts	
POST-CONDITION:	System displays traveler’s favorited posts.	
BUSINESS RULES	• User must be signed in • There is no limit on favorited posts • Favorited posts will be displayed chronologically • If user has no favorited posts, the page will note that instead of being blank	
IMPLEMENTATION CONSTRAINTS AND SPECIFICATIONS	• Frontend/UI development with responsive design • Backend development • Session management/user authentication • MongoDB for data retrieval	
ASSUMPTIONS:	User is signed into the application User has favorited/bookmarked posts (However this is not mandated)	
OPEN ISSUES:	None at this time	

USE CASE NAME:	Click Make Post	USE CASE TYPE Business Requirements: □ System Analysis: □
USE CASE ID:	00009	
PRIORITY:	High	
SOURCE:	OnTheGo! Design Specifications Team Research Industry Standards	

PRIMARY BUSINESS ACTOR	Travelers	
PRIMARY SYSTEM ACTOR	OnTheGo! Application	
OTHER PARTICIPATING ACTORS:	• For Business: • Business trying to promote their restaurant, activities, transportation etc • Traveler influencers • Tour Guide Companies etc	
OTHER INTERESTED STAKEHOLDERS:	• Airlines • Traveler Agents • International Businesses trying to promote their restaurant, activities, transportation etc	
DESCRIPTION:	A traveler may be interested in creating a post on the discover page. They can click "Make a Post" tab in the nav bar on the Discover page. They can then fill out its required information if they would like.	
PRE-CONDITION:	Traveler must be logged into OnTheGo! application.	
TRIGGER:	Traveler clicks "Make a Post" icon in the navbar.	
TYPICAL COURSE OF EVENTS:	**Actor Action**	**System Response**
	Step 1:	**Step 2**:
	Traveler clicks "Make a Post" icon in the navbar.	System redirects user to template page, displaying a post template
ALTERNATE COURSES:	Traveler clicks out of page and is redirected.	
CONCLUSION:	Traveler successfully views post template	
POST-CONDITION:	System successfully displays post template	
BUSINESS RULES	• User must be signed in to view post template	
IMPLEMENTATION CONSTRAINTS AND SPECIFICATIONS	• Frontend/UI development with responsive design • Backend development • Session management/user authentication	
ASSUMPTIONS:	• User is signed into the application	
OPEN ISSUES:	None at this time	

USE CASE NAME:	**Make Post**	**USE CASE TYPE** Business Requirements: □ System Analysis: □
USE CASE ID:	00010	
PRIORITY:	High	
SOURCE:	OnTheGo! Design Specifications Team Research Industry Standards	
PRIMARY BUSINESS ACTOR	Travelers	
PRIMARY SYSTEM ACTOR	OnTheGo! Application	
OTHER PARTICIPATING ACTORS:	• For Business: • Business trying to promote their restaurant, activities, transportation etc • Traveler influencers • Tour Guide Companies etc	

OTHER INTERESTED STAKEHOLDERS:	• Airlines • Traveler Agents • International Businesses trying to promote their restaurant, activities, transportation etc	
DESCRIPTION:	A traveler may be interested in creating a post on the discover page. They can click "Make a Post" tab located on the Discover page and fill out its contents if they would like to make a post. After filling it out they can click the "Post" button on the bottom of the forum and thats when it would be posted on the Discover page and can be viewed there.	
PRE-CONDITION:	Traveler must be logged into OnTheGo! application.	
TRIGGER:	Traveler has clicked "Make a Post" icon in the navbar.	
TYPICAL COURSE OF EVENTS:	**Actor Action**	**System Response**
	Step 1:	**Step 2**:
	Traveler completes required template fields	System saves content in real time
	Traveler clicks "Submit" button in lower right corner	System validates post and publishes it to the Discover page
ALTERNATE COURSES:	Traveler clicks out of "Make a Post" template page and is redirected.	
CONCLUSION:	Traveler successfully creates a post that posts to the discover page.	
POST-CONDITION:	System successfully stores & posts user data.	
BUSINESS RULES	• User must be signed in to create a post	
IMPLEMENTATION CONSTRAINTS AND SPECIFICATIONS	• Frontend/UI development with responsive design • Backend development • Session management/user authentication • MongoDB for data retrieval	
ASSUMPTIONS:	• User is signed into the application	
OPEN ISSUES:	None at this time	

USE CASE NAME:	**Click MyPlanner**	**USE CASE TYPE** Business Requirements: □ System Analysis: □
USE CASE ID:	00011	
PRIORITY:	Medium	
SOURCE:	OnTheGo! Design Specifications Team Research Industry Standards	
PRIMARY BUSINESS ACTOR	Travelers	
PRIMARY SYSTEM ACTOR	OnTheGo! Application	
OTHER PARTICIPATING ACTORS:	• For Business: • Business trying to promote their restaurant, activities, transportation etc • Traveler influencers • Tour Guide Companies etc	
OTHER INTERESTED STAKEHOLDERS:	• Airlines • Traveler Agents • International Businesses trying to promote their restaurant, activities, transportation etc	
DESCRIPTION:	A traveler may be interested in the options clicking MyPlanner offers, they may want to view their planners or create a new one.	

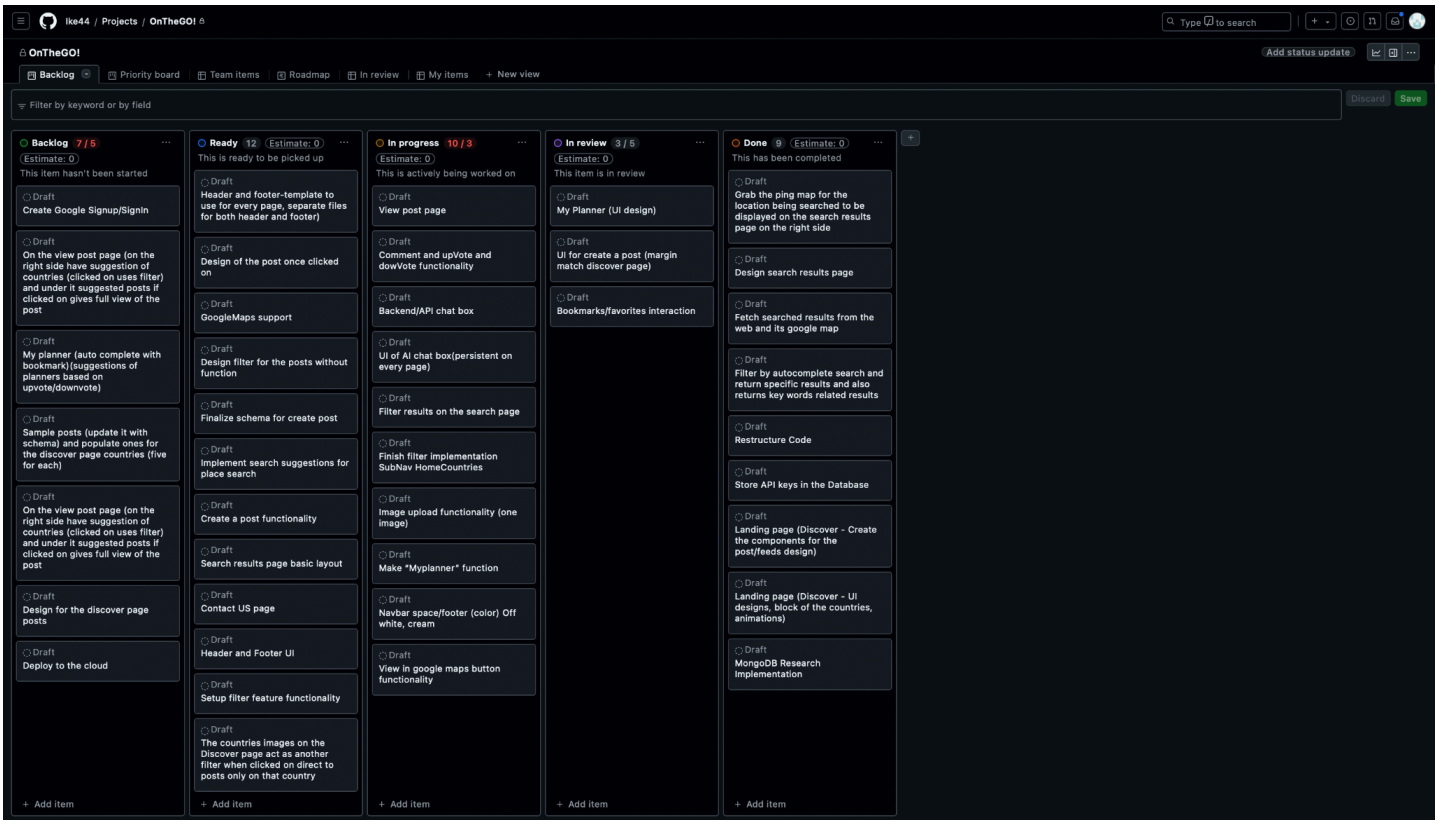
PRE-CONDITION:	Traveler must be logged into OnTheGo! application.	
TRIGGER:	Traveler clicks “My Planner” button	
TYPICAL COURSE OF EVENTS:	**Actor Action**	**System Response**
	Step 1:	**Step 2**:
	Traveler clicks My Planner button in the navbar	System redirects to the MyPlanner page
ALTERNATE COURSES:	Traveler clicks out of page and is redirected.	
CONCLUSION:	Traveler successfully views MyPlanner page	
POST-CONDITION:	System successfully displays MyPlanner page	
BUSINESS RULES	• Traveler must be signed in to view MyPlanner options	
IMPLEMENTATION CONSTRAINTS AND SPECIFICATIONS	• Frontend/UI development with responsive design • Backend development • Session management/user authentication	
ASSUMPTIONS:	• Traveler is signed into the application	
OPEN ISSUES:	None at this time	

USE CASE NAME:	**View Planners**	**USE CASE TYPE** Business Requirements: □ System Analysis: □
USE CASE ID:	00012	
PRIORITY:	Low	
SOURCE:	OnTheGo! Design Specifications Team Research Industry Standards	
PRIMARY BUSINESS ACTOR	Travelers	
PRIMARY SYSTEM ACTOR	OnTheGo! Application	
OTHER PARTICIPATING ACTORS:	• For Business: • Business trying to promote their restaurant, activities, transportation etc • Traveler influencers • Tour Guide Companies etc	
OTHER INTERESTED STAKEHOLDERS:	• Airlines • Traveler Agents • International Businesses trying to promote their restaurant, activities, transportation etc	
DESCRIPTION:	A traveler may be interested in viewing their planners.	
PRE-CONDITION:	Traveler must be logged into OnTheGo! application and has clicked “My Planners” in the navbar.	
TRIGGER:	Traveler clicks “Planners”	
TYPICAL COURSE OF EVENTS:	**Actor Action**	**System Response**
	Step 1:	**Step 2**:
	Traveler clicks “Planners”	System redirects traveler to Planner page
ALTERNATE COURSES:	Traveler clicks out of page and is redirected.	
CONCLUSION:	Traveler successfully views their created planners	
POST-CONDITION:	System successfully displays user’s created planners	
BUSINESS RULES	• User must be signed in • There is no limit on planners • Planners will be displayed chronologically • If user has no planners, the page will note that instead of being blank	

IMPLEMENTATION CONSTRAINTS AND SPECIFICATIONS	• Frontend/UI development with responsive design • Backend development • Session management/user authentication • MongoDB for data retrieval
ASSUMPTIONS:	• User is signed into the application • User has already created planners (However this is not mandated)
OPEN ISSUES:	None at this time

USE CASE NAME:	**Make Planner**		**USE CASE TYPE** Business Requirements: □ System Analysis: □
USE CASE ID:	00013		
PRIORITY:	Low		
SOURCE:	OnTheGo! Design Specifications Team Research Industry Standards		
PRIMARY BUSINESS ACTOR	Travelers		
PRIMARY SYSTEM ACTOR	OnTheGo! Application		
OTHER PARTICIPATING ACTORS:	• For Business: • Business trying to promote their restaurant, activities, transportation etc • Traveler influencers • Tour Guide Companies etc		
OTHER INTERESTED STAKEHOLDERS:	• Airlines • Traveler Agents • International Businesses trying to promote their restaurant, activities, transportation etc		
DESCRIPTION:	A traveler may be interested in planning a trip using the OnTheGo! Planner template.		
PRE-CONDITION:	Traveler must be logged into OnTheGo! application and has clicked “MyPlanners” in the navbar.		
TRIGGER:	Traveler clicks “New Planner”		
TYPICAL COURSE OF EVENTS:	**Actor Action**	**System Response**	
	Step 1:	**Step 2**:	
	Traveler clicks “New Planner”	System redirects user to planner template page	
	Traveler fills in necessary information to create new planner and clicks “Submit”	System stores new planner and redirects user to My Planners page.	
ALTERNATE COURSES:	Traveler clicks out of page and is redirected.		
CONCLUSION:	Traveler successfully creates a new planner		
POST-CONDITION:	System successfully stores a new planner for user.		
BUSINESS RULES	• User must be signed in to view their account information		
IMPLEMENTATION CONSTRAINTS AND SPECIFICATIONS	• Frontend/UI development with responsive design • Backend development • Session management/user authentication • MongoDB for data retrieval		
ASSUMPTIONS:	• User is signed into the application		
OPEN ISSUES:	None at this time		

Backlog



Main use-cases presenting the technical details based on requirement engineering:

1. Search and Discovery

- Actors: Travelers
- Description: Travelers use a search bar that integrates Google Search Suggestions API to find specific locations, experiences, or businesses. The search results include posts by other users, business promotions, and Google-sourced data like reviews and operational hours.
- Technical Details:Integration with Google APIs to fetch search suggestions and place information., filters to refine search results based on ratings, categories, and other tags, aerver-side logic to merge user-generated content with Google data based on search parameters.

2. Create Post

- Actors: Travelers
- Description: Travelers can create, view, edit, and delete their posts about travel experiences. They can also upvote, downvote, and bookmark posts.
- Technical Details: CRUD operations on the posts stored in MongoDB, user authentication to ensure only the post creator can edit or delete their posts,front-end components in React for creating and managing posts

3. Interaction with posts

- Actors: Travelers
- Description: Users interact with posts through comments, upvotes, and downvotes, and by bookmarking posts for later reference.
- Technical Details:Implementation of interactive elements like buttons and forms,backend APIs to handle the logic of user interactions such as voting and bookmarking, real-time update features to reflect changes without reloading.

4. Itinerary Planning

- Actors: Travelers
- Description: Travelers can create customized travel itineraries using a planner feature, which can auto-fill with data from bookmarked posts.
- Technical Details: Backend services to store and retrieve user-created plans and bookmarks.

5. Business post

- Actors: Business Owners
- Description: Business owners promote their services or products directly on the platform, targeting travelers interested in specific locations or types of travel.
- Technical Details: tools for businesses to create, manage, and display promotional content, filtering and tagging system to associate promotions with relevant content dynamically.

6. AI Customer Support Chatbot

- Actors: Travelers and Business Owners
- Description: An AI chatbot available on every page to provide instant customer support and assistance.
- Technical Details: Integration of AI technologies to interpret and respond to user queries, continuous learning and updating of the AI model to improve response quality and relevance.

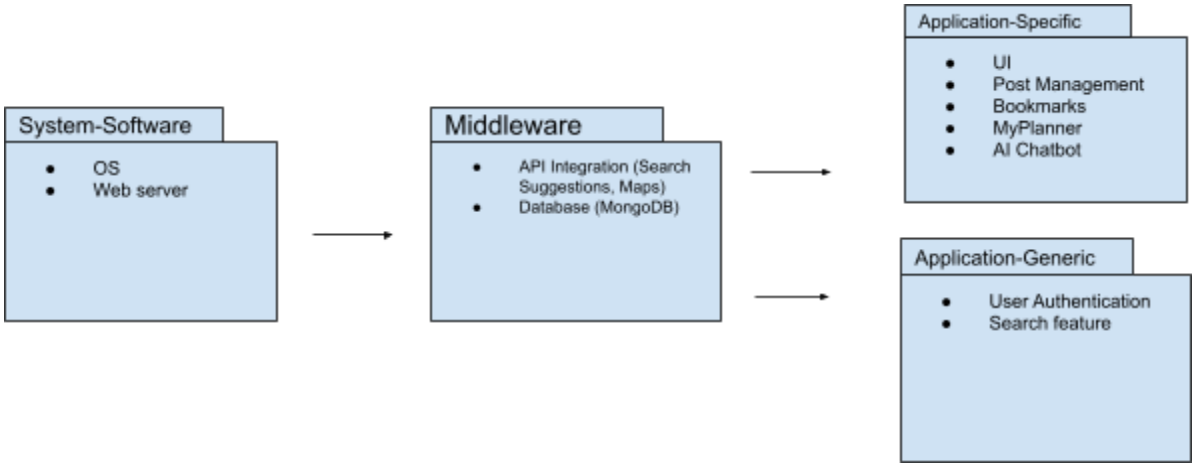
2. Architectural Overview

From the start, the rationale behind the OnTheGo architecture has remained fairly consistent. Our team decided on a client-server architecture, with a frontend built with React and a Node.js backend. The internal data is stored in MongoDB and we integrate external APIs. This architecture effectively meets our requirements by being scalable, maintainable, and dynamic.

The application's frontend is built with React to provide a user-friendly and responsive interface. It handles user interactions, including but not limited to: creating posts, filtering, and search functionalities. Initially our approach involved a static dropdown for location based features, however after ample research we pivoted to integrating the Google Search Suggestions API to meet usability requirements and offer real time validated suggestions. The application also implements the Google Maps API to display locations on maps and provide users with information on businesses and places. The application's backend is developed in Node.js to reliably handle requests from the frontend, integrates the APIs, and manages data flow. For the database our team selected MongoDB early on based on our familiarity with it. That said, MongoDB is more than a reasonable selection as it offers dynamic storage for various data types, it is scalable, and suitable for our application.

While the architecture was fairly set from the start, we did weigh our options and consider other approaches to building OnTheGo on a client-server base. Microservices were considered because of their flexibility in managing independent functionalities but that was scrapped because the project evolved into an overcomplication of maintaining separate services. A monolithic approach was also considered but was also scrapped because our team decided a more modular approach would be most beneficial to the project. Ultimately, a client-server architecture is the best fit for OnTheGo as it provides a secure, scalable, and reliable structure to our system.

2.1 Subsystem Architecture



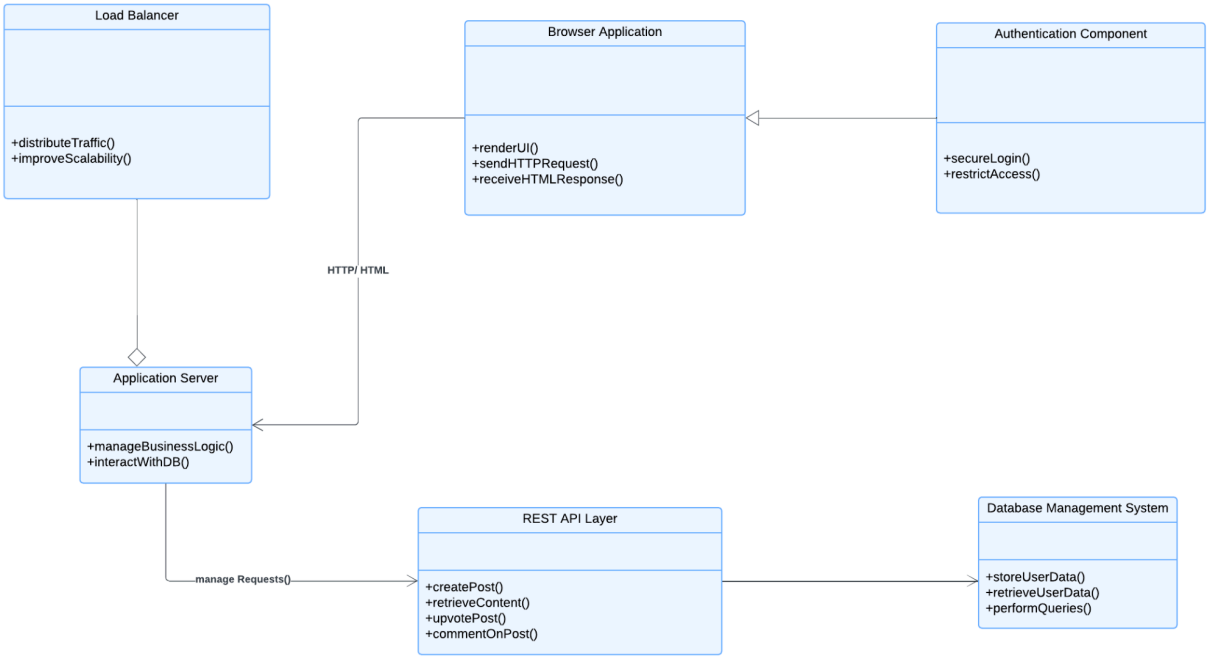
System-Software: This package defines the foundation of the system.
Middleware: This package holds the main operations including server management, API integration, and the database.
Application-Generic: This package includes management for user sign in/up, and the search feature.
Application-Specific: This package includes user-interface, as well as application specific features such as post management, bookmarks, MyPlanner, and the AI Chatbot.

2.2 Deployment Architecture

This section should include a UML Deployment Diagram (and supporting discussion) that includes a mapping of subsystems to deployment. The UML deployment diagram should show how the major components of your system will be assigned to different processors or different computers, and explain how those computers are connected (e.g., wireless, Ethernet LAN, Internet). Also, describe the communication protocol used (e.g., plain Java sockets, Java RMI, Java JDBC, HTTP, etc.) and why.

If you have a single-threaded software system that runs on a local device and does not connect to any other systems running on different computers across a network, then you do not need to complete this section – just write a single sentence stating "This software will run on a single processor."

For references on UML deployment diagrams, see the Sparx tutorial or the optional UML text listed on the syllabus, or visit www.uml.org.



We use HTML as the main communication protocol due to the fact that it can communicate with all web browsers, which allows for webpages to be displayed properly for a wide range of devices.

2.3 Persistent Data Storage

This section should identify what pieces of information must be stored and your approach to storing data (e.g., flat files, relational database). If you are using a flat file, you will identify the file format (what are the elements stored in the file, how are they separated, how do you represent the file). If you are using a database for persistent storage, give the database schema (i.e., describe the tables and their columns).

If your system does not need to store any data after a complete execution of the system (e.g., after the user exits), then you do not need to complete this section – just write a single sentence stating “This software does not require persistent data storage.”

For the OnTheGo! project, data is stored persistently using a MongoDB Atlas database, named onthegoDB. This database contains multiple collections to support different functionalities in the app. Here’s an outline of the database schema:

Post Collection:

- title (String, required): The title of the post.
- body (String, required): The main content of the post.
- image (String, required): The image URL associated with the post.
- upvotes (Number, default: 0): Count of upvotes for the post.
- downvotes (Number, default: 0): Count of downvotes for the post.
- createdAt (Date, default: Date.now): Timestamp of post creation.
- country (String, required): The country associated with the post.
- city (String, required): The city associated with the post.
- category (String, required): Category label for the post.
- review (String, required): User review or description of the location.
- website (String): Link to the associated website.
- locationID (String, required): Identifier for the location.
- duration (Number): Suggested duration for visiting the location.

Planner Collection:

- title (String, required): Title of the plan.
- description (String, required): Description of the plan.
- dueDate (Date, required): Target completion date for the plan.
- status (String, enum: ‘pending’, ‘in-progress’, ‘completed’, default: ‘pending’): Current state of the plan.
- createdAt (Date, default: Date.now): Creation timestamp.

Comment Collection:

- postId (ObjectId, reference to Post, required): ID of the associated post.
- author (String, required): Name of the comment’s author.
- content (String, required): Comment text.
- createdAt (Date, default: Date.now): Comment creation timestamp.

Bookmark Collection:

- userId (ObjectId, reference to User, required): ID of the user who bookmarked the post.
- postId (ObjectId, reference to Post, required): ID of the bookmarked post.
- createdAt (Date, default: Date.now): Bookmark creation timestamp.

2.4 Global Control Flow

This section should describe assumptions about how your system's execution is controlled. You should identify any of the following models of control flow, and explain how they are applied (it is perfectly acceptable that you apply more than one of these):

- *Procedural or event-driven?*: Is your system *procedure-driven* and executes in a "linear" fashion, where every user every time has to go through the same steps, or is it an *event-driven* system that waits in a loop for events, and every user can generate the actions in a different order?
- *Time dependency?*: Do you have any timer-controlled actions in your system? Is your system of event response type, with no concern for real time, or is it a real-time system? If it is real-time, is it periodic, and what are the time constraints for each period?
- *Concurrency?*: Does your system use multiple threads? If so, identify the components that have separate threads of control and describe how the threads are synchronized.

Procedural or Event-Driven:

The system is event-driven, allowing users to interact in a non-linear fashion. Users can initiate different actions, such as creating posts, adding comments, or managing their planner without a fixed sequence, with each user interaction triggering specific system responses.

Time Dependency:

The system operates as an event-response type without real-time requirements. No timer-controlled actions are implemented, as it mainly relies on user-initiated events without strict timing constraints.

Concurrency:

The system currently does not implement multiple threads, as MongoDB manages concurrent database requests, providing built-in mechanisms for handling concurrent data access.

3 Detailed System Design

This section presents a detailed design for the architecture described in the Architectural Overview, and should be consistent with the architectural styles and package diagram identified in Sections 1 and 2 of your documents. For each major part (i.e., component, module, or package) identified in

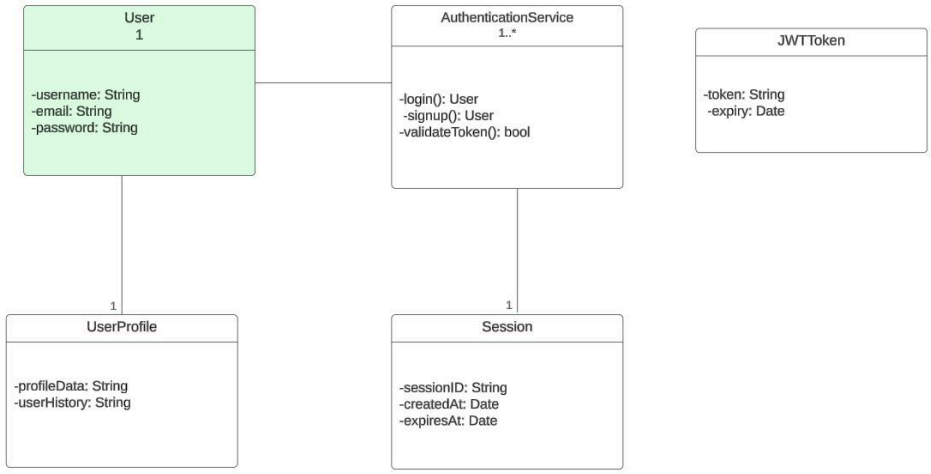
your architecture, create a subsection. In each subsection that corresponds to a component or package, you will model the static view with UML class diagrams and the dynamic view with UML sequence diagrams:

3.1 Static view

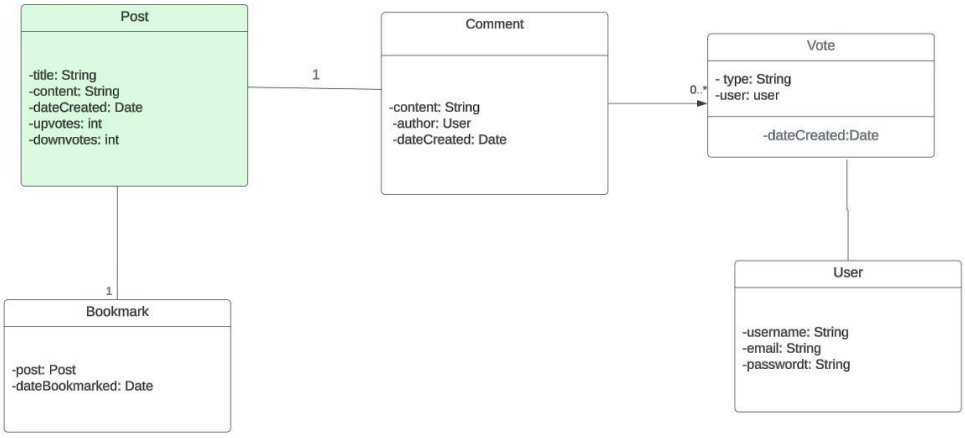
You must include UML class diagrams showing further decomposition of the major modules and the relationships among these classes. In each class in the UML class diagram, you must show:

- Important attributes, their type, and their visibility
- Important operation/method names, their parameters, return types, and their visibility (public, private, protected, package)
- Associations between classes and multiplicity constraints

Component/Module User Management and Authentication UML Diagram



Component Module: Post Management UML Diagram



In addition, you will also need to write support text that justifies your decomposition of your modules into the classes shown in your UML class diagram. This justification should discuss the responsibilities of each class. The justification should describe other alternative designs, if any, and why your design is better.
Finally, you should describe any design patterns that are included in this design and why you've applied them.

The classes in the UML diagram are decomposed to separate concerns and ensure scalability. The User Management module includes the JWTToken, User, and AuthenticationService classes. The user class is used to store the user data and the AuthenticationService handles the login and signup. JWTToken manages the JWT generation for stateless authentication. The Post Management modules include comment, vote, post, and bookmark classes. The classes handle post content, user interactions like votes and comments, and bookmarking.

Designs could have combined classes such as post and vote, but keeping the two separate enhances flexibility and scalability. Design Patterns like Singleton for centralized authentication, Factory for token creation, Strategy for flexibility authentication methods, Observer for updating comments/votes, and Composite for managing posts with comments make the system more maintainable, modular, and extensible.

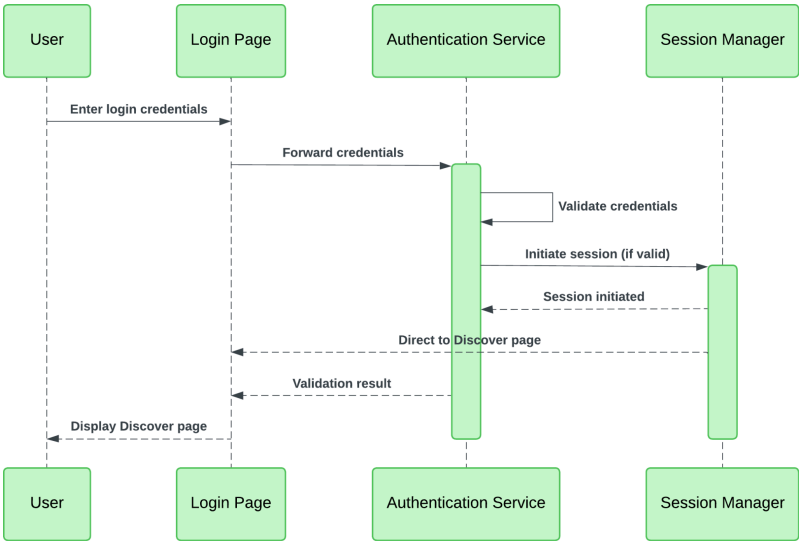
For reference material, see [Robert Martin's article on UML class diagrams](), the Sparx tutorial on UML [package]() and [class]() diagrams, or the optional UML textbook recommended on the syllabus. For reference material on design patterns, check out the POSA and GoF or this very handy [online catalog of design patterns]() or this website on [common patterns]().

3.2 Dynamic view

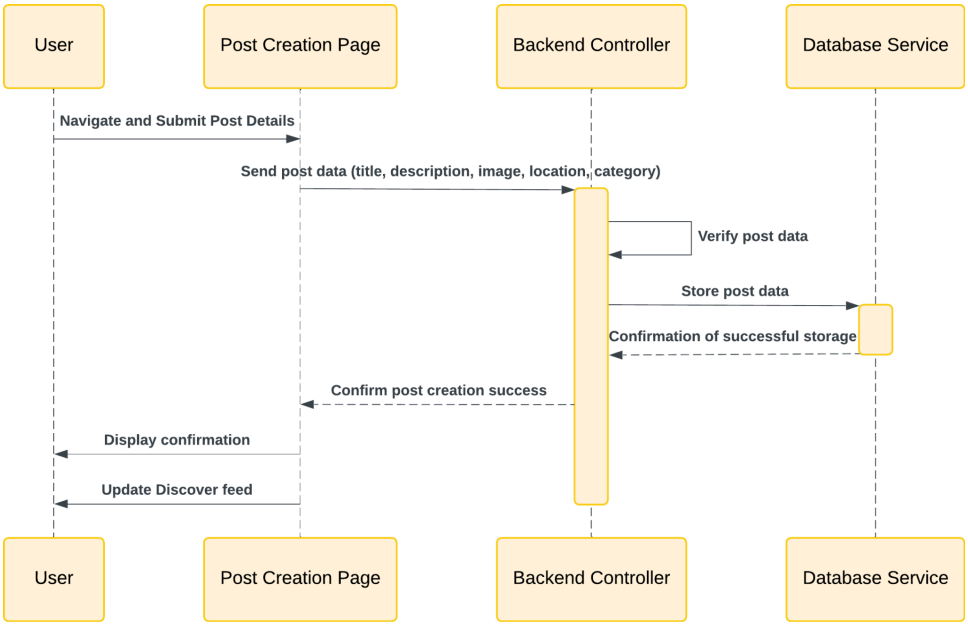
You must show the design of your system's behavior using UML sequence diagrams. These sequence diagrams should show the time-ordered sequence of interactions among classes to support an important system function. Your sequence diagrams should be consistent with the class diagrams given in Section 3.1. In other words, you should not have participating objects in an interaction that do not appear in a class diagram; if you find that this is the case, you should go back and revise your class diagram to include the new element. You may supplement your sequence diagrams with state-transition diagrams or activity diagrams (useful for describing algorithms), but these are not required.

Make sure to add your Testplans, previous and current sprints and brief sprint reviews as well as any additional items you feel are helpful at any point. Remember, it's not necessary to add documents that are not providing useful information. Only necessary items. You can add/subtract items or update them using version control to your design.

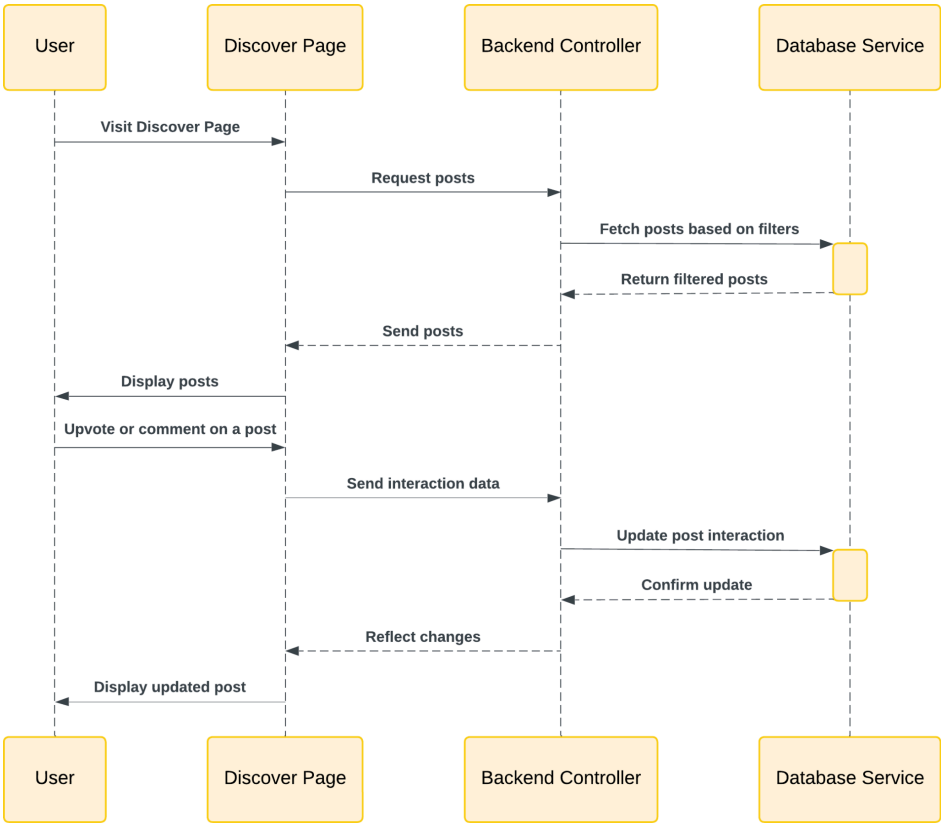
User Login and Authentication:



Making a Post:



Viewing and Interacting with Posts on the Discover Page:



Submission:

Submit your design documents inside your team Github repo. Moreover, **submit a PDF document in Canvas along with the link to repo; we will not grade any other document type**. Name your

electronic submission as follows: **Team<number>_D2.pdf.** Submit your team assignment via Canvas. Only one team member needs to submit the document.

Tips:

Provide supplemental text to explain your diagrams through captions for them! Make sure that you have described, somewhere in the document, the responsibilities of the elements (e.g., components/modules/classes) in your architecture/class/sequence diagrams. You should describe your design decisions that led you to this design, including a discussion of any alternative designs you considered but discarded. Providing these kinds of descriptions helps in understanding your design (as such, they can have a positive impact on your grade!). *Where to Stop/When to Stop Drawing Interaction Diagrams*? Generating a sequence diagram for the most important user stories is typically a good way to start. If you find that you have a bunch of sequence diagrams that essentially repeat the same interaction, then you are not creating useful models, just redundant ones. In this case, generalize the interaction and provide supplemental text that explains how and where the generalized interaction can be applied to capture multiple user stories/use cases. **Remember, you should model only what is needed and useful as discussed in the class.**

Apply Design Principles! You have learned about the importance of modules with high cohesion, a design with low coupling, and the benefits of relying on abstraction versus a concrete realization (e.g., application logic communicates with a hardware abstraction layer instead of directly with devices). Make sure that you apply these principles and clearly explain how your design achieves them.

Proofread your documents! Everyone on your team should proofread the document before it is submitted. It is important that we be able to understand your design to evaluate it, so you must take care in communicating your design. If you are not skilled in technical writing, plan in advance so that you can ask someone to proofread it for you. The university has a writing resource center that may be helpful to you.

What UML tools should I use to draw the diagrams? Any design tool may be used to draw your UML diagrams such as Draw.io and Lucidchart as we discussed in the class.

Be aware that while drawing programs may provide template tools for creating UML diagrams, those templates may not meet the diagramming guidelines we discussed in class. For instance, some versions of Microsoft Visio provided the wrong arrow for an “extends” relationship in the past for use cases. You should check to make sure your diagram meets the standards discussed in class and make manual edits in the drawing program if necessary to adhere to those standards.