**Destination Bucket List Application – Use Cases**

**Use Case: Register**

**Actor: User**

**Flow of Events:**

1. **User accesses the registration page.**
2. **User fills in the required information.**
3. **System validates the entered data.** 
   1. **If the entered data is invalid or incomplete or already exists, the system displays an error message and asks the user to correct the information.**
4. **System creates a new account for the user.**

**Entry Condition: User is not registered.**

**Exit Condition: User account is created, and user is redirected to the login page.**

**Quality Requirement: Registration process should take less than 10 seconds to execute, and the user interface should be friendly and helpful.**

**Use Case: Login**

**Actor: User, Admin**

**Flow of Events:**

1. **Actor enters their credentials on the login page.**
2. **System validates the entered credentials.** 
   1. **If the entered credentials are incorrect, the system displays an error message and asks the actor to try again.**
3. **System logs the actor in and redirects to the appropriate page (private bucket lists page for User, public location management page for Admin).**

**Entry Condition: Actor is registered.**

**Exit Condition: Actor is logged in.**

**Quality Requirement: In the login form, once the form is submitted, a loading circle appears during the data validation and login process.**

**Use Case: Logout**

**Actor: User, Admin**

**Flow of Events:**

1. **Actor clicks the logout button.**
2. **System logs the actor out.**
3. **System redirects the actor to the login page.**

**Entry Condition: Actor is logged in.**

**Exit Condition: Actor is logged out.**

**Quality Requirement: The logout button should be easily visible in the user interface.**

**Use Case: Manage Account**

**Actor: User Flow of Events:**

**a. ModifyAccount**

1. **User accesses the account settings page.**
2. **User updates account information.**
3. **System validates and saves the updated information.** 
   1. **If the updated information is invalid or incomplete, the system displays an error message and asks the user to correct the information.**

**b. Delete Account**

1. **User accesses the account settings page.**
2. **User requests account deletion.**
3. **System confirms the deletion request.**
4. **System deletes the user account and associated data.**

**Entry Condition: User is logged in.**

**Exit Condition: User account is modified or deleted.**

**Quality Requirement: Both options should be displayed together in a separate page.**

**Use Case: Manage Location**

**Actor: User, Admin**

**Flow of Events:**

**a. Create Location**

1. **Actor accesses the appropriate location management page (public list for Admin, private list(s) for User).**
2. **Actor enters the required information for a new location.**
3. **System validates and saves the new location.** 
   1. **If the entered information is invalid or incomplete, the system displays an error message and asks the actor to correct the information.**
4. **System adds the new location to the appropriate list (public list for Admin, private list for User).**

**b. Delete Location**

1. **Actor accesses the appropriate location management page (public list for Admin, private list(s) for User).**
2. **Actor selects a bucket list (in case it is a User), and from there, a location to delete.**
3. **System confirms the deletion request.**
4. **System removes the location from the appropriate list (public list for Admin, private list for User).**

**Entry Condition: Actor is logged in.**

**Exit Condition: A new location is created, or an existing location is deleted from the appropriate list. Quality Requirement: There should be a pop-up confirming the deletion of a location.**

**Use Case: Manage Private Bucket List**

**Actor: User**

**Flow of Events:**

**a. Create Private Bucket List**

1. **User requests the creation of a new private bucket list.**
2. **The system creates a new empty private bucket list for the user.**

**b. Add to Private Bucket List**

1. **User accesses a public list of destinations or their existing private bucket lists.**
2. **User selects a destination to add to a private bucket list.**
3. **System adds the selected destination to the user's chosen private bucket list.**

**c. Remove from Private Bucket List**

1. **Users access their private bucket list page.**
2. **User selects a destination to remove.**
3. **System confirms the removal request.**
4. **System removes the destination from the user's private bucket list.**

**d. Delete Private Bucket List**

1. **Users access their private bucket list page.**
2. **User requests to delete one of their private bucket lists.**
3. **System confirms the deletion request.**
4. **System deletes the user's selected private bucket list.**

**Entry Condition: User is logged in.**

**Exit Condition: User's private bucket list(s) is updated.**

**Quality Requirement: Manage Private bucket list and manage location functionality should be on the same page.**

**Use Case: Initialize Public Bucket List**

**Actor: System**

**Flow of Events:**

1. **System initializes the public bucket list when the admin logs in.**

**Entry Condition: The admin logs in for the first time.**

**Exit Condition: Public bucket list is initialized.**

**Quality Requirement: The initialization should not be visible by the admin.**