

CPEN 321

Feedback on Requirements

Scope - 1/2

- Sufficient complexity: we ask for 5-6 non-trivial use cases/requirements, i.e.,
 - not too fine-grained
 - “Add item”, “Remove item”, “Update item” → not three separate use cases (make it “Manage Items”)
 - “Create study group”, “Join study group”, “Update study group” → not three separate use cases
- At least 3-4 should involve non-trivial use case implementation, i.e.,
 - should involve an algorithm/logic → not a simple API call or database query
 - good example: calculating matches based on user preferences and usage patterns
 - should be complete
 - good example: dealing with sign-up, sign out, password reset during the authentication

Read the feedback from the last week project presentations to get inspired!

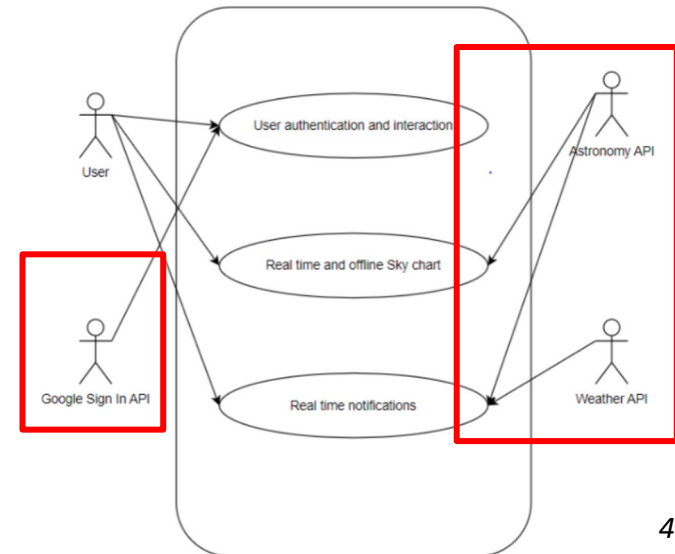
Scope - 2/2

The following must be included (might or might not have non-trivial implementation, depending on how you implement it)

- Authentication
 - Should use external authentication service, e.g., Google or Facebook authentication
 - Just Login (as in M1) is considered trivial
- Usage of one additional external API, e.g., Google calendar or Google maps
- Real- time updates, e.g., multi-user chats or real-time push notification (push vs. pull on the server side)

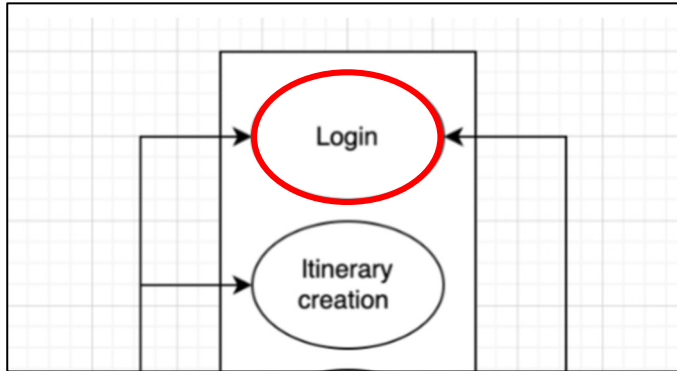
Use Cases and Actors

- Use cases are “active” (phrased as verbs not nouns)
 - E.g., "Study groups" => “Manage study groups”
- If the app does not need 2-3 actors, it is fine to have 1, as long as the app is well-scoped
- Actors are users of the app, **not app developers** who interact with code, fix bugs, etc.
- External APIs are not active users of the app (cannot have arrows pointing to use cases)



Use Case Name == Functional Requirement Name

- Use case names do not match the names of functional requirements



Functional Requirements:

Name of Requirement: User Registration and Authentication

Short Description of Requirement: Users should be able to create accounts and log in securely.

Primary Actor(s): Travelers, Admin

Success Scenarios

- Scenarios should not be a high-level description of the cases but a **sequence of execution steps**

Success Scenario(s):

- Users create groups and successfully collaborate on trip planning.
- Group members can make joint decisions on itinerary items.

Failure Scenario(s):

- Users encounter errors while trying to create or manage groups.

How?



Success Scenario(s):

- User can search for events by category (e.g., music, sports, art).
- User can search for events by location (e.g., city or region).
- User can search for events by date range (e.g., this weekend, next month).
- User can search for events using keywords (e.g., "concert," "yoga class").



See lecture notes "W3 L1 RE" (slides 46-47) and "W3 L2 More on RE" (slides 8-9) for examples of how to specify requirements correctly

Failure Scenarios - 1/2

- Each failure point corresponds to a failure in a particular success scenario steps
- Each failure point describes how the failure is handled (what the user see)

Main Success
Scenario

1. User presses “start chat with financial adviser” button
2. Financial adviser is notified of the request
3. User is connected with a financial adviser

Failure
Scenarios

- 3a. User cannot connect with financial adviser due to network problems
- 3a1. An error message is displayed telling user of the error, and potential solutions
 - 3a2. The app prompts the user to try again after a set time period (e.g. 30s).



Failure Scenarios - 2/2

- It is not a list of things that can possibly go wrong



Failure Scenario(s):

- User registration fails due to a technical issue.
- Users cannot log in due to incorrect credentials.

What is the resolution?

- Failure scenarios are not bugs or negative cases in the app:



2. The user chose to be both the customer and restaurant owner

- The user dislikes the recipes.

*Negative scenario
but still expected
(success not error)*

*This looks like a
bug/usability issue in
the app!*


Non-functional Requirements (NFR)


- Should be measurable and verifiable
- Should be project-specific (not generic sentences which can apply to any project)



Non-Functional Requirements:


1. Performance is crucial to ensure a smooth and responsive user experience, especially when generating itineraries and accessing recommendations. Performance can be validated by measuring the app's response time and ability to handle concurrent user requests under load.

- 
- NFR should be complete (cannot focus on usability of only one minor part of the app)



2. Browsing only requires scrolling & one tap to add a friend (usability)

- NFR should be realistic (you will need to make it work!)



d. The server should be available 24/7 to the users with no to minimal downtime

Today!

- 1) Feedback to groups on their Requirements
 - 2) Sign up to your group's voice and text channels on Discord (will be made private soon)
 - 3) "Project Selection and GitHub Repository Setup" assignment on Canvas.
- Discuss projects proposed by your group and your peer-group in your community.
Submit:
 - a) The name of the project your peer-group wants you to implement (out of the two that you suggested)
 - b) The name of the project you want your peer-group to implement (out of the two that your peer-group suggested)
 - Create a GitHub repository for your group project following the rules in the assignment. **Submit** the URL