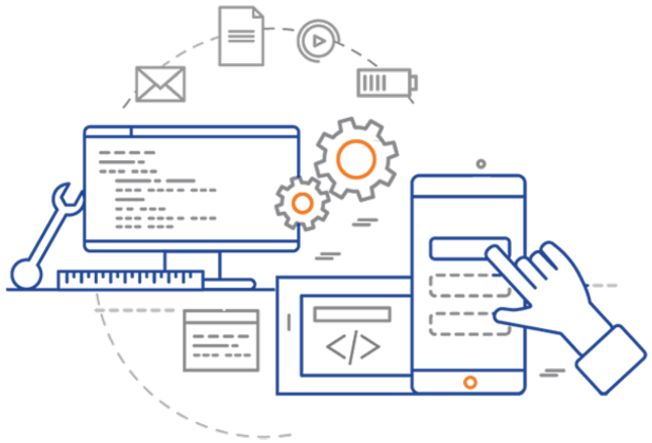


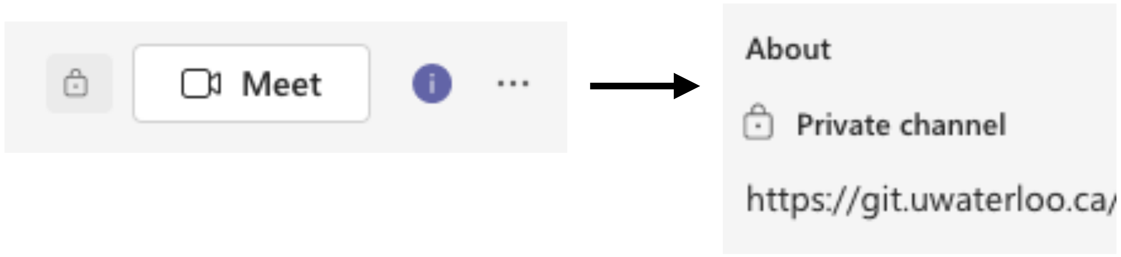
CS 398: Application Development

Week 03 Lecture: Analysis & Design 1

Goals; Architectural characteristics

Administrative

- Check your MS Teams channel to make sure that the URL to your repo is there.
  - If it's not listed, please add it (or email to Jeff if you cannot add it manually).
  - Also make sure that course staff are listed as Reporter or higher (Guest will not work!)



- Announcement today: return to in-person moved to at-least Feb 7th.
  - This changes nothing for us, we're fine to continue online! Design Review through MS Teams.

Where are we?

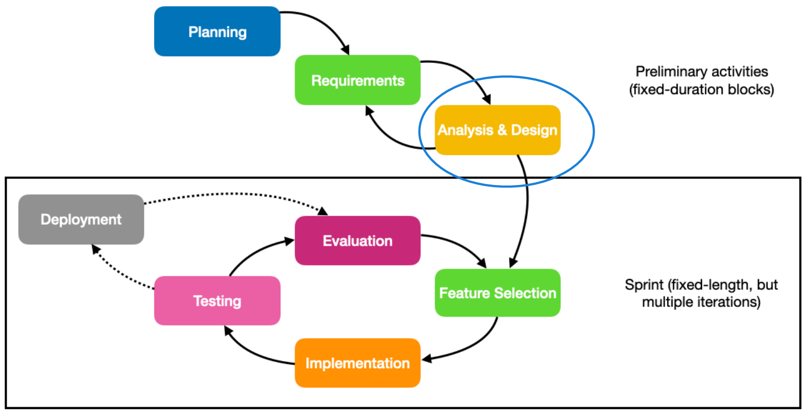
Planning

1. Create project plan ✓

Requirements

1. Pick users, (create personas) ✓
2. Interview people that fall into your role ✓
3. Identify requirements, (affinity diagram) ✓
4. Document requirements in GitLab. ✓

Analysis & Design ←



Software Development Lifecycle (SDLC)

Phase 2: Affinity Diagrams

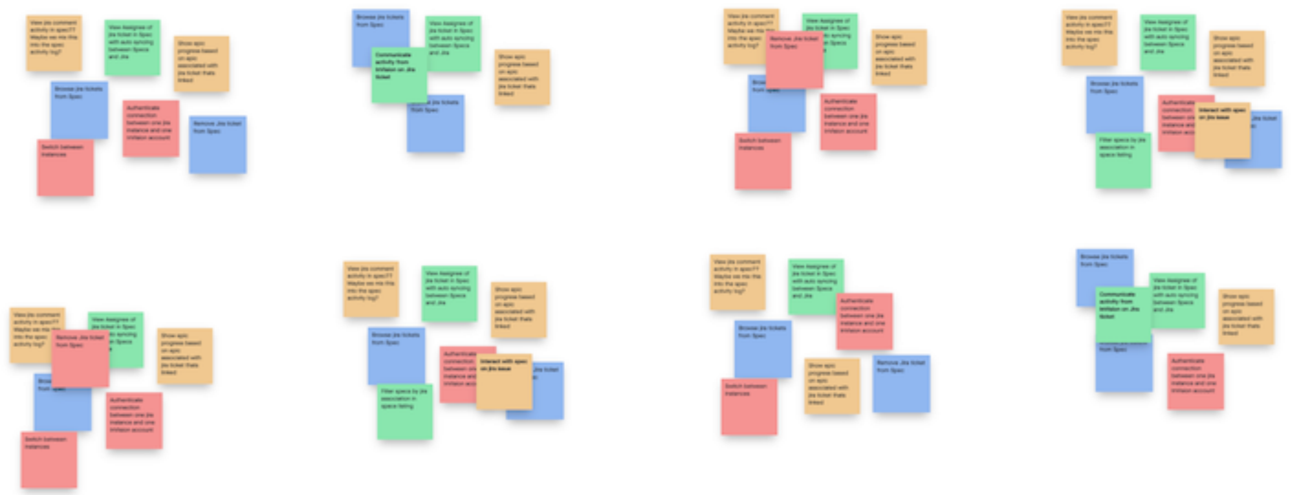
Recall from last week

The outcome of your interviews will likely be a lot of qualitative data -- opinions, suggestions and ideas that the user expressed. The challenge is extracting ideas, themes from this narrative that you can use in your design.

Use an **affinity diagram** to expose critical ideas or themes from the user's ideas. (You can even do this across multiple users and looks for which themes occur more frequently!).

To create an affinity diagram:

1. Record all notes or observations on individual cards or sticky notes
2. Look for patterns in notes or observations that are related and group them
3. Create a group for each pattern or theme
4. Give each theme or group a name
5. Create a statement of what you learned about each group (provide your analysis or key insight)



Affinity Diagram created in InVision.

https://www.invisionapp.com/freehand/templates/detail/affinity-diagram-template

Phase 3: Analysis & Design

1. Determine the **technical viability** of our requirements, and identify any **technical constraints** that they will impose.

2. Determine important **non-functional requirements** (or characteristics).

3. Finally, we want to determine the **structure** of our system.

Technical Impact

Review each requirement and consider its **technical impact**.

- Will we be able to to implement this feature? Is it even possible?
- Is this feature risky; does it require significant novel research, or is it complex to meet?
- What other technologies do we require for this to be feasible? (e.g. frameworks, libraries, toolkits).
- How would implementing this feature impact our system? How does it relate to other requirements?

You want to add *technical depth* to the existing requirements.

Non-Functional Requirements

The architecture phase is concerned about the **non-functional requirements** or “architectural characteristics” [Richards & Ford 2020] that emerge from the structure of our system.

An architectural characteristic meets three criteria:

1. specifies a non-domain design consideration (i.e. non-functional requirement),
2. influences some structural aspect of design,
3. is critical or important to the success of the application.

These are also called software ***qualities.*** *e.g. performance, reliability, scalability and so on.*

Identifying Characteristics

So how do you identify which characteristics are relevant? You should look at the functional requirement (and project goals if they apply) and determine if there are any characteristics that help address user needs.

These can support your feature list.

Domain Concern	What do we look for?	Relevant Characteristics
Time to market	Characteristics that ensure that we can execute quickly and reliably.	Agility; Testability; Deployability
User satisfaction	Does your user work directly with this product? These are characteristics that directly affect user's perception, and availability of your product.	**Performance**; **Usability**; **Configurability**; Localization
Privacy	Some types of data are sensitive (e.g. working in Healthcare). Anything that would protect the integrity and access to data.	Authorization; Authentication; **Security**
Competitive advantage	In a competitive market, we want to be able to introduce and modify features quickly, even after we've released to the market.	Testability; Deployability; Supportability

Logging NFRs

Once you’ve identified a characteristic to address, you need to document it as a non-functional requirement (NFR).

- Include details of how you will measure it (all NFRs need to be measurable)
- Document how you will verify that you have met the requirement.
- Include consideration of this NFR in your topology and design (below).
- Clearly communicate this as an undercutting requirement to your team.

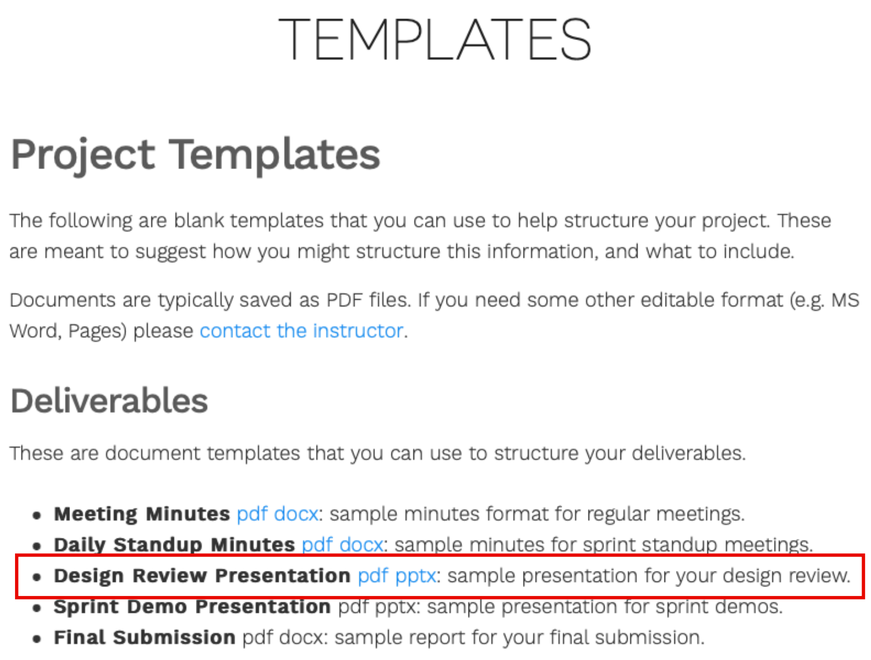
Example of non-functional requirements:

- “The application should launch and restore all state in less than 5 seconds on our target platform. We will measure this by manually timing execution during integration testing.” <— (Quantitative)
- “Users should find our interface usable. We will measure this by surveying 5 users, and asking them to rate our interface on a 1-5 Likert scale. The mean score should be 3 or higher.” <— (Qualitative)

Activities

Design Review

Design review at the end of next week. See the design review template!



TEMPLATES

Project Templates

The following are blank templates that you can use to help structure your project. These are meant to suggest how you might structure this information, and what to include.

Documents are typically saved as PDF files. If you need some other editable format (e.g. MS Word, Pages) please contact the instructor.

Deliverables

These are document templates that you can use to structure your deliverables.

- **Meeting Minutes** pdf docx: sample minutes format for regular meetings.
- **Daily Standup Minutes** pdf docx: sample minutes for sprint standup meetings.
- **Design Review Presentation** pdf pptx: sample presentation for your design review.
- **Sprint Demo Presentation** pdf pptx: sample presentation for sprint demos.
- **Final Submission** pdf docx: sample report for your final submission.

Right now you are "building evidence" to justify your choice of requirements.

TODO Today

Planning

1. Create project plan ✓

Requirements

1. Pick users, (optional) create personas ✓
2. Interview people that fall into your role ✓
3. Identify requirements, (affinity diagram) ✓
4. Document requirements in GitLab ✓

Analysis & Design

1. Determine technical impact ✎
2. Choose architectural style
3. Diagram system

Quiz in LEARN this week! Opens at 12 PM, due by Fri 11:59 PM.

You should be finishing up requirements today, and starting architecture ASAP.