Lecture: January 18

Class Syllabus Review and Coding++

Agenda

- Spring Schedule Review
- Rubric for Sprint grades
- Testing Overview
- Unit Tests and System Tests
- Usability Tests
- Code Reviews

Schedule and Syllabus for Spring Semester

- Lectures and Alumni Panels - **Send it your topic requests!**
- 3 Demos: Beta, Prelim Prototype, and Final
- 2 Presentations: Revamped Elevator Pitch and Final
- Assignments:
  - Presentations and Mock Demo: 10%
  - Webpage: 10%
  - Sprints: 20%
  - Demos: 10%
  - Participation: 5%
  - Final Demo and Presentation: 35%
  - Final Package: 10%

Syllabus
Schedule

Sprints Rubric

Last Semester

5 points for strong level of commitment and active updates to board

4 points for moderate board activity

2 points for minimal board activity

0 points for no board activity

Spring Semester

5 points for 1.) team member has strong level of commitment, 2.) all tickets addressed as either “Done”, “Won’t Do”, or moved to next sprint, and 3.) sprint includes QA Testing and Code Review cards

4 points for 1.) team member has moderate level of commitment, 2.) majority of the tickets addressed as either “Done”, “Won’t Do”, or moved to next sprint, and 3.) sprint includes QA Testing and Code Review cards

2 points for 1.) team member has minimal level of commitment, 2.) a few tickets addressed as either “Done”, “Won’t Do”, or moved to next sprint, and 3.) sprint does not include QA Testing and Code Review cards

0 points for no board activity

Sprints Rubric

Last Semester Sprints

October Sprint

November Sprint

Spring Semester Sprints

December+January Sprint

February Sprint

March Sprint

April Sprint

QA Testing

Types of Testing

- Unit Tests: tests individual pieces of a solution
  - Focused and low-level
  - Ensures that each piece of code is working as intended
- System Tests: tests the entire system working together
  - Good for high-level confirmation that the system is working
  - Can help you see more complex use cases
  - Also known as integration tests
  - Harder to write and run
- Manual testing by Engineer
- Usability Testing or Heuristic Evaluation

Structuring Unit Tests

- Isomorphism
  - Your test structure should match your code structure
- Naming tests
  - Helpful to include the name of the method and the thing being tested in the test name (eg. test_loadUserData_cantOpenFile)
- Each test should be short and focused
  - A bunch of smaller tests is preferred over one large test
  - Easier to have code reviews of functionality and its tests
- Code Coverage
  - Goal: test every piece of functionality
  - Isomorphism helps to visualize this
  - Can use testing tools to gauge code coverage

Usability Testing

Why?

- You won’t know how good your app is until users use it
- Expert users/builders know too much
- Hard to predict what users will do

Types

- Guided interview
- Journal or Diary Study

Note, User testing answers the question **“Do users need the app?”**, whereas Usability testing answers the question **“Can users use the app?”**

Usability Testing vs Heuristic Evaluation

Usability Testing	Heuristic Evaluation
Potential users test out the user interface with real tasks	Someone looks at the user interface and identifies the problems
Ask users to complete specific tasks	Explain nothing but the purpose of the system
Create structure for how each participant will be guided through tasks	Create rules for how participants can structure their feedback
Performed by potential users	Performed by experts

Structuring and Conducting Usability Tests

Preparing for the Interview

1. Define the goal of the interview (What do you want to learn? What purpose will this achieve?)
2. Define the user groups you'd like to target
3. Prepare any assets (prototypes, mockups, terms, etc.) and make sure they're uniformly defined/tested for users
4. Section your interview into distinct topic areas, so you can orient the user: "Let's start with some questions about your background...Let's move onto [X topic]..."
5. Prepare the interview structure to be easily copy/paste-able so you can take notes during the interview per user
6. Prepare a script if the interview covers a sensitive topic / if there are questions you may anticipate
7. Consider randomizing lists of items if that is an interview question

Choosing Participants

Representative of your persona types

- Use the persona types you have listed in your Writings

Approximate, when needed

- Eg. Looking for professors, so you might interview a TA instead

Incentivize participants

- Swag, Free coffee/pizza, Offering to be a participant in their research

Conducting the Test

- Record the interview
- **[Part 1]** User's background (so you can section responses, sample questions on next slide)
- **[Part 2]** Ask the user to complete a few tasks within your interface
- **[Closing]** Thank the user for their time!
- Feel free to ask about competition if relevant (for benchmarking, etc.)
- Note that whatever is stated in the conversation should remain private as we haven't launched anything and some details the user might want to keep private
  - You should say this depending on the sensitivity of the interview topic

Part 1: User's Background

Understand the user demographic and characteristics to determine how close they fit within your target audience

General Information / Screener Questions

- How old are you?
- Are you graduating soon and looking to move?

Pre-Test Questions

- Have you ever leased an apartment?
- How comfortable are you with browsing real estate marketplace apps?
- When searching for an apartment, do you have a list of must-have's and if so, what are they?
- Have you ever used Zillow, Apartments.com, or StreetEasy?

Let's say we're testing a Real Estate Marketplace app targeted at New Grads

Part 2: Questions about your interview topic

Have the participant complete 3-5 tasks to determine usability of the features

“I have the application loaded on the device here and will ask you to complete a few tasks. Please share your thoughts aloud as you work through the tasks and note that any opinion or feedback you have is valuable to our team.”

- Sign-up / Login Flows
- Create profile and set preferences
- 2-3 use cases from use case document
    - For new grad real estate marketplace app: Search for an apartment, View apartment details, Contact apartment broker, Save apartment listing, View Saved listings

Analyzing Results

Types of Data Collected

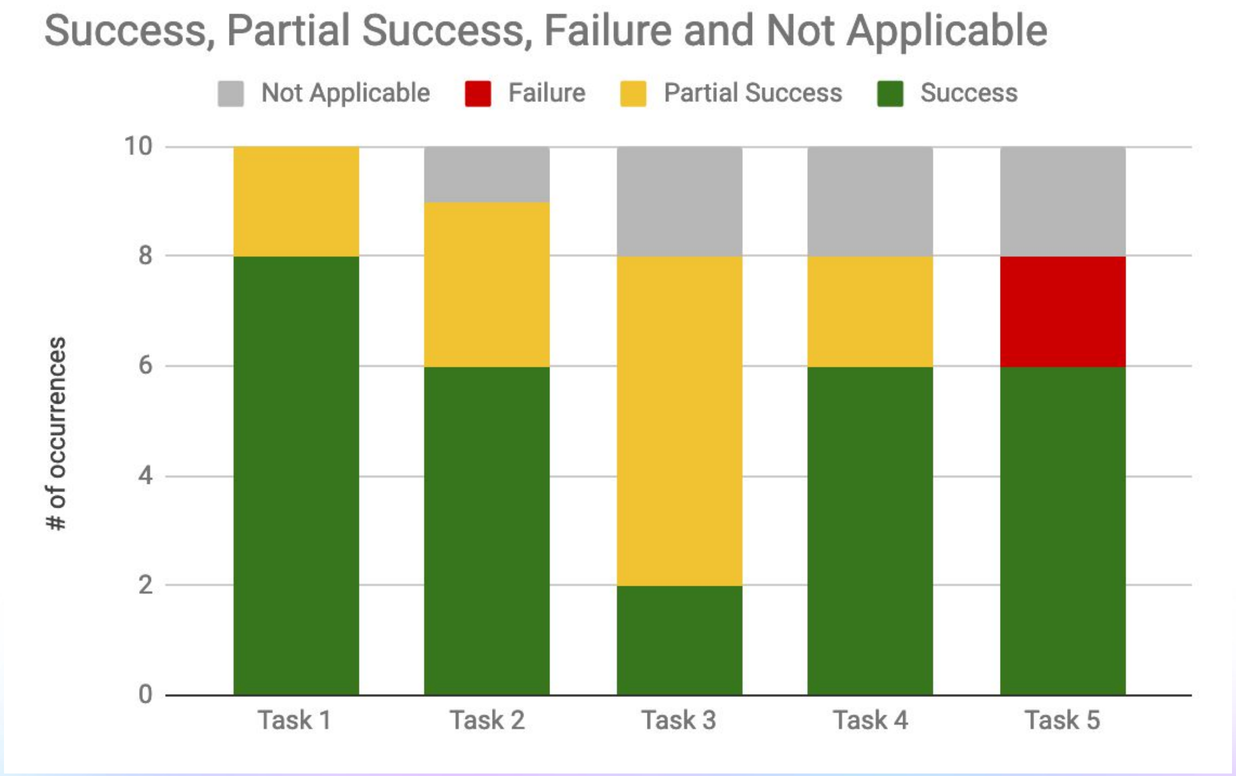
Action-based data

- Summary of tasks that the user successfully or unsuccessfully completed
- Quantitative data
- *Eg. The participant successfully navigated to the Profile and set their price range.*

Thinking out loud data

- Observations of what the user is doing or thinking
- Qualitative data
- *Eg. The participant said they were lost when trying to Save an apartment.*

Quantitative Data



“Thinking out loud” Data

- Prompt the user: “Tell me what you’re thinking”
- Pros:
    - **Understand what** the user is thinking
    - **Understand what** the user is trying to do
    - **Understand what** questions the user has
    - **Understand what** the user is reading or paying attention to on screen
- Cons:
    - May be uncomfortable for the user
    - May take more time to run the interview and process the feedback

Analyzing Qualitative Results

- Collect all feedback into a list
- Rate the items in the list on frequency, severity, and level of effort
  - Frequency: How often did this feedback occur?
  - Severity: Does this issue prevent the user from completing the task or serve as a distraction?
  - Level of effort: How much work will it require to fix this?
- Prioritize the list
- Decide on needed changes
- Note, not all feedback needs to be fixed/addressed

Example Results for a Scheduling App

Priority	Feedback	Frequency	Severity	LOE	Type
1	When I try to search for an apartment within my price range, the filter always resets to the default range and I cannot search.	3	High	M	Bug
1	When I view my Saved apartments, I want to filter by my top housing preferences.	4	Medium	S	Feature Request
2	I would like an email notification when an apartment on my Saved list goes OFF market.	2	Low	L	Feature Request
3	I would like a daily list of recommended apartments similar to my searches.	2	Low	M	Feature Request

Testing for your Project

- Track QA work in Trello cards
- Set goals for Unit tests, System tests, and Usability tests, when applicable
- For Usability tests
    - Create testing questions and tasks
    - Choose participants
    - Run tests
    - Document results
    - Decide on which results to act on

Code Reviews

Code Reviews

A code review (also referred to as peer code review) is a process where one or two developers analyze a teammate’s code, identifying bugs, logic errors, and overlooked edge cases

- PR Reviews
- Live code pairing

Talk to your mentor about setting team guidelines on Code Reviews

Code Reviews

Considerations

- **Readability**: Are there any redundant comments in the code?
- **Security**: Does the code expose the system to a cyber attack?
- **Test coverage**: Is there a need to test more cases?
- **Architecture**: Does the code use encapsulation and modularization to achieve separation of concerns?
- **Reusability**: Does the code use reusable components, functions, and services?

Don’t review more than 200-400 lines of code at a time.

QA Engineer

Who? Roles and Responsibilities?

- Set goals and objectives for unit test and system test coverage
- Establish framework and principles for testing
- Integrate automated testing software into development process
- Work with Engineering team and Product manager to determine key use cases to address in testing

Industry?

- Any Engineering organization