Lecture 5: October 5

Development and Delivery

Logistics

1. Presentation next week - Rubric posted this week
2. Writing 1 due next week - Rubric has been posted
3. Github Repos - add
    a. Clone to SD Class Main
    b. Keep on your own but add instructors and mentor as owner

Agenda

- Recap learnings
- Product Requirements Documentation
- Technical Specifications and Shaping
- Choosing your language, frameworks, and APIs
- Working with APIs
- Roles: Architect, Engineering Lead, Software Engineer, QA Engineer

Recap

Lecture 1: Types of Project Management Methodologies

Lecture 2: Discovery and Research

Lecture 3: Project Planning

Lecture 4: Mock Interviews

Lecture 5: Practice Presentations

Lecture 6: Development and Delivery

Lecture 7: First Graded Presentation

Lecture 1: Project Management Methodologies

- Two most common methods: Waterfall and Agile
- Software mainly uses Agile
- Methodologies are not mutually exclusive (eg. you can be “agile” in a waterfall process)
- A Project Manager, Scrum Owner, or Product Manager owns this process
- Trello, Jira, Confluence, and other “Atlassian” projects are industry standard for tooling in this space
- We will be using a “flavor” of Agile and Scrum for our projects

Lecture 2: Discovery and Research

- Discovery begins with the following questions:
  - What are the goals?
  - What are the problems?
  - What does success look like?
- The “Goals” and “Ideas” of the GIST Framework can help a team answer those questions
- Quantitative Research includes User Surveys, A/B testing, and Product Data Analysis (Clicks on a page, Views on a site, etc.)
- Qualitative Research includes User Interviews and Competitive Research
- ***Exit Criteria: Problem Statement Defined***

Lecture 3: Project Planning

- Product Requirements Documents and Gantt charts are tools that help Product & Engineering teams plan and track the execution of their solution
- Scrum Owners, Project Managers, or Product Managers own the planning process
- Technical Team (Designers and Engineers) own the Solutions and LOE
- T-Shirt Sizing is used to help determine the LOE required
- Alpha, Beta, and GA launches are defined by a set of User Stories that you hope to deliver in each phase to reach the MVP (Minimal Viable Product)
- ***Exit Criteria: Product Requirements and Project plan made***
    - ***Executive Summary, Technical Summary, and Gantt Chart thus far***

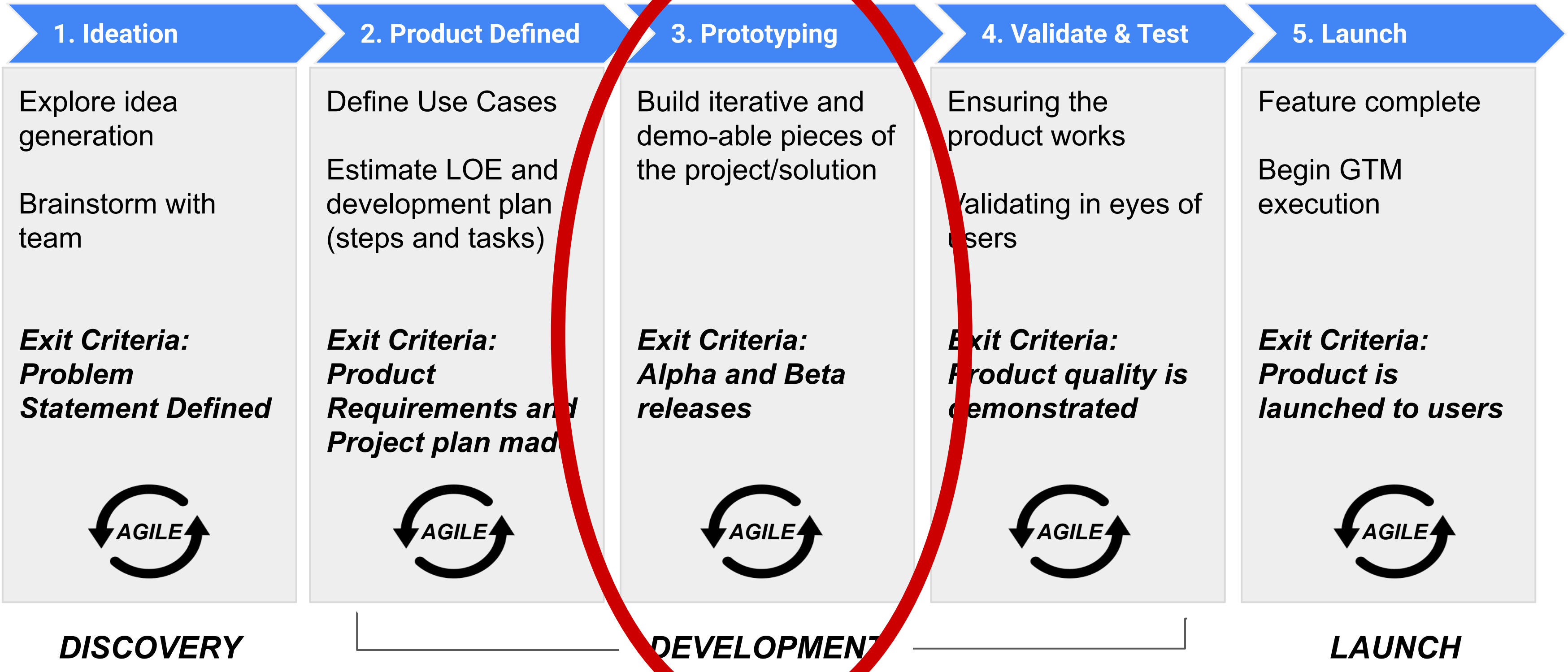
Lecture 4: Mock Interviews

- Technical Interviewing Practice
- Future Lecture (November 2nd) we will cover skills for Behavioral Interviews as well as finding, interviewing, and signing an offer for a job

Lecture 5: Presentation Practice

- First “pitch” of your project to your peers
- Main focus was on the Problem Statement and your Ideas for Solution
- First graded Presentation next week - same content.
- Future Presentations will become more about your Solution as you reach Alpha, Beta, and Final Demos

Product Definition Phase



1. Ideation

Explore idea generation

Brainstorm with team

Exit Criteria: Problem Statement Defined

2. Product Defined

Define Use Cases

Estimate LOE and development plan (steps and tasks)

Exit Criteria: Product Requirements and Project plan made

3. Prototyping

Build iterative and demo-able pieces of the project/solution

Exit Criteria: Alpha and Beta releases

4. Validate & Test

Ensuring the product works

Validating in eyes of users

Exit Criteria: Product quality is demonstrated

5. Launch

Feature complete

Begin GTM execution

Exit Criteria: Product is launched to users

DISCOVERY

DEVELOPMENT

LAUNCH

Product Requirements Document

(Pre-work to Development)

Section 1: Proposal

Section 2: Specs

Section 3: Considerations, Constraints, Dependencies

Section 4: Open Questions

Writing / Planning

Section 1: Proposal ...

Executive Summary ✅

Section 2: Specs...

Technical Summary ✅

Gantt Chart ✅

Product Specifications (upcoming)

Technical Specifications (Been creating via Trello cards, Slack messages, Notes)

“Putting it all together” ... Technical Design Document (upcoming)

Section 1: Proposal ✅

Your elevator pitch to your team, stakeholders, investors, etc.

- Goals

  What are you hoping to accomplish? What does Success look like?

- Users

  Who are you building for?

- Use Cases

  What will the user be able to do once you reach Success?

Section 2: Specs (upcoming writings 3 and 4)

The meaty "what" section to describe the work to be done.

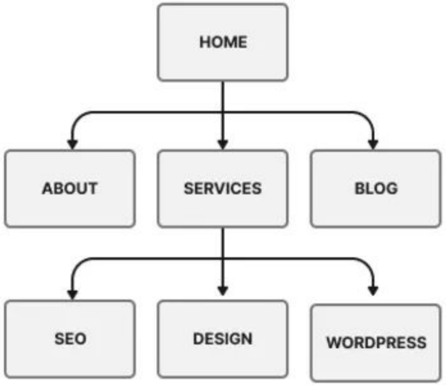
- Product Specifications
  - User Stories
  - Mockups
  - Flow Diagrams
- Technical Specifications
  - Languages
  - Frameworks, APIs
  - Theories, Algorithms

Specs: Product Shaping

September Sprint: What problems do we want to solve?

- User Experience Mockups, Flows, and Wireframes
  - What user flows exist in the project?
  - What actions can the user take?
  - What does the UX look?
- User Stories
  - As a ___ user, I would like to ___ so that I can ___.
  - Eg. As a Frisbee aficionado, I would like to see the best team matchups, so that I can bet on the winning team.

SITE MAP



HOME

ABOUT

SERVICES

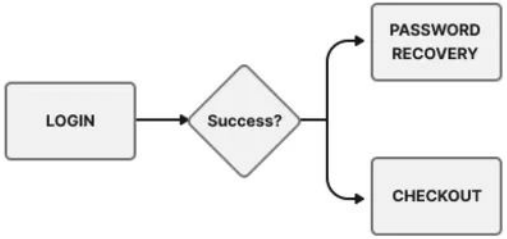
BLOG

SEO

DESIGN

WORDPRESS

FLOW DIAGRAM



LOGIN

Success?

PASSWORD RECOVERY

CHECKOUT

WIREFRAMES

User Stories

As a site user, I need to login so that I can checkout.

As a site user, I need to login so that I can recover my password.

etc …

Specs: Technical Shaping

October Sprint: What solutions will solve these problems?

- What language
  - Front end or backend
  - iOS or Android
  - Web App or Mobile App
- What algorithms
  - What algorithms am I building?
  - What algorithmic theory applies here?
- What APIs
  - What libraries, databases, or programs do I need to connect to in order to build my solution?
  - API Documentation - good example of technical documentation

Writing / Planning (Pre-work to Development)

Section 1: Proposal …

- Executive Summary ✅

Section 2: Specs…

- Technical Summary ✅
- Gantt Chart ✅
- Product Specifications (upcoming)
- Technical Specifications (Been creating via Trello cards, Slack messages, Notes)

“Putting it all together” … Technical Design Document (upcoming)

Coding Principles and Practices

1. Speed
2. Reusability
3. Deletability (Modularity)

In general, pick 2 and align with business’ needs

What’s an API?

Application **P**rogramming **I**nterface

- A communication layer between two computer programs
- Exposes key information and functions that the developer needs to use

REST (Built for APIs)

- POST
- GET
- PUT/PATCH
- DELETE

CRUD (Built for Database)

- CREATE
- READ
- UPDATE
- DELETE

Steps to using an API

1. Authentication
   a. Connect to the API using valid “login” credentials
   b. Generate valid token to use as credentials when calling on any method within the API
2. Know your inputs
   a. Required vs Non-required Parameters
3. Know your outputs
4. Synchronous vs Asynchronous
   a. **Synchronous**: The response to your request will be your output
   b. **Asynchronous**: The response to your request will be a **key**. You use that key to make another API call to retrieve your output

API Practice

Jokes API

PokeAPI

Trivia API

Recipes API

Roles

Architect

Engineering Lead

Software Engineer (Typically, Frontend or Backend)

QA Engineer

Typical 40-hr Work Week of a SWE or Architect**

Meetings (10 hours)

- Team “Rituals”: Standups, Sprint Planning, Retros, etc (3 hours)
- Pairing Sessions: 1 hour per team member per week
- Architect Chats: Cross-team meetings with other Architects/Leads (2 hours)
- General Meetings: Company-wide All Hands, Interviews, etc (2 hours)

PR Reviews (2-3 hours, sometimes more)

- Consideration: You will have to maintain this code
- Consideration: You will have to deal with legacy code

Thinking/Planning/Technical Designing (10-15 hours)

- What will this product look like now, in 6 months, in 2 years?

Hands on Keyboard (10-15 hours)

- Lots of prototyping code

**This was taken from interviews with SWEs and Architects at *my* company. But it’s not the same everywhere, ask your mentor about their typical work week!

Architect

Who? Roles and Responsibilities?

- Analyze the requirements for the Product, and extract those requirements concerning the Architectural significance
- Create a structure of solutions that can meet all the various requirements, balance goals, and constraints on the solution.
- Have to make sure that everyone is on the same page and understands the Architecture.
- Ensure that the Developers can realize the Architecture which could be done by a combination of mentoring and direct involvement.

Engineering Lead

Who? Roles and Responsibilities?

- Sometimes synonymous with Architect for a particular area of the product
- Leads one or more teams to successfully complete product launch or project
- Supports developers and works with Product Manager

Software Engineer

Who? Roles and Responsibilities?

- Build tools using software development practices
- Can *range* in type of product, language, area of the code base, frontend vs backend, and many more specifics to what the project needs
- Talk to your mentors about what their area of expertise is for their engineering team

QA Engineer

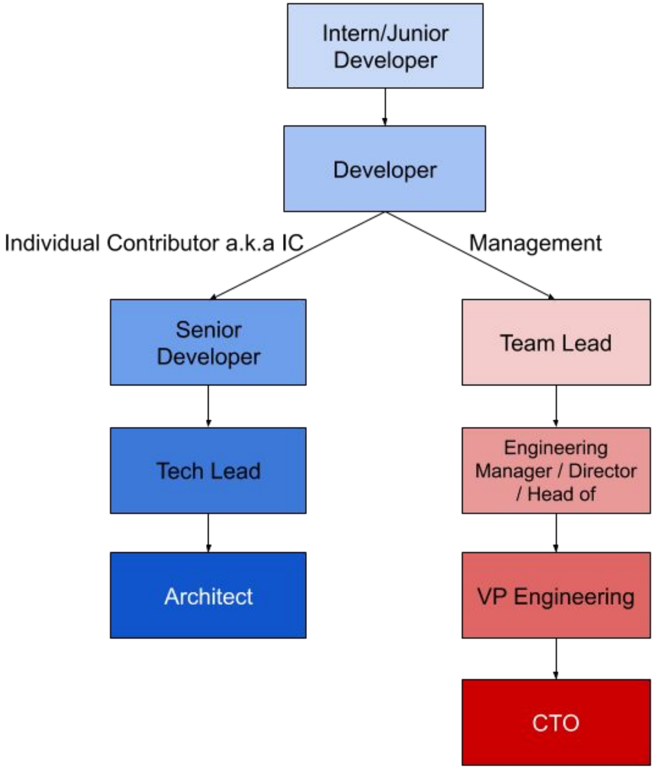
Who? Roles and Responsibilities?

- Responsible for making the QA plan for Product launch
- Works with Product Manager to define the user stories that must be tested
- Works with Engineers to define the technical unit tests that must be tested
- Builds and Maintains the QA frameworks and tooling
- Sometimes outsourced or done by the software engineers themselves

The IC vs Management Track

“IC” : Individual Contributor

Management: Anyone with a direct report


Intern/Junior Developer
Developer
Individual Contributor a.k.a IC
Management
Senior Developer
Tech Lead
Architect
Team Lead
Engineering Manager / Director / Head of
VP Engineering
CTO