On the job work!

What kind of work are you getting?

- New feature
- Bugfix
- New Product

New Feature

Very common

- Work with existing code
- Established patterns of code
    - And User Interface (UI)

Bugfix

- May need to confirm bug
    - Operating from user report
- Need to reliably replicate :(
    - Write a test
- Need to understand code
- Fix the bug

New Product

- Rarest of Work
- May have constraints
    - Libraries, patterns
- Lots of details
    - Setup
    - UI
    - QA
    - Deployment

UI/Flow Design

- Do you have Designer? PM?
    - Who is deciding the user flow?
- May get wireframes, mockups, or prototypes
    - Wireframes: Outline of page parts
    - Mockups: Image of page
    - Prototype: HTML-based mockup
        - w/limited interactivity

First Lesson: Do not agree!

You will be asked for your opinion

- You may want to say "Looks good!"
    - DO NOT DO THIS!
- They are asking if you can implement it as shown
    - You didn't consider that yet
    - If you agree you will have rude surprises

What to consider

- Do they give you info to replicate
    - Fonts, colors, distances
    - Image assets
    - Text
- Do they handle responsive page size?
    - vs fixed width
    - Mobile dimensions?
- Do they indicate interactivity?
    - Hover/Touch effects?
    - Error messaging?
    - Loading indicators?

Data Sources?

- Existing services?
  - Do they have the data you need?
  - Will you have the required inputs/auth?
- New Services?
  - Who is writing?
  - What is schedule?
    - When will you know the contracts?
    - How does that impact your deliverables?
- Error conditions?
- Load timing/delays?

Answer: Can do X by Y if Z

- Clarify the expectations of yourself
- Set clear expectations of others
- Set clear limitations on your abilities
- Remember how hard estimation is!
- Push back when needed
    - That's your role!
        - They won't do it for you
    - Say what you need
    - Target sustainable pace

Technical Design

- Smaller features won't have explicit step
    - But you need to update tests and docs
- Mesh your concept with the existing code
    - Patterns and models
- You need to understand what exists
- "Frankenstein" code is terrible to work with

Updating Data Models

- Can you describe what your code will do
    - Without visuals
    - In terms of data models
- Is this compatible with existing data models?

Tests

- New Feature will need to be tested
- Do you know how existing tests work?
- TDD gets tests first
    - But is slower if you aren't used to TDD
- Without TDD
    - Know what the tests will test
    - And how
- Working code that is hard to test is not great

Explicit Tech Design

- Write docs summarizing what will happen
    - Data Flow
    - Data Models
- Docs get reviewed
    - Commented
    - Approved?
- Address issues
    - Possibly cycle again

Writing Code

- Broken into explicit tasks
- Following designs
- Includes tests
    - Can be longer than you expect!

Coding Tasks

- Usually 1-3 days
- Tasks has definition of "done"
- May need to be demonstratable
- Includes tests

Testing Code

Do not expect QA to catch your mistakes

- They should be verifying that there aren't any
    - Because you already tested yourself
- QA makes you look good
    - Unless you abuse them

Demonstrating

- Some teams "demonstrate" during/after sprint
- Should be fast, visible
    - Be ready
    - May not be "impressive"
        - That's fine

Code Review

Code Review before PR is merged

- Other people: not on your schedule
- Leave time!
- Provide context and anticipate questions

Human Review

Programming is Communication

- Code Review only a little about "does it work"
- More about "Does it communicate to humans"

Do not be insulted by questions or suggestions

- When reviewing, focus on "real" problems
- Always have clarity about what change is required

Tests

Most CI efforts will run tests during Code Review

- Merge not allowed without passing tests
- May include integration, e2e tests
- Slow, possibly brittle

Merging the Pull Request

You will have to merge after approval

- Different than this class

Merge as soon as you can!

- Other merges will retrigger tests
- Causes delay

Deployment

Deployment styles vary

- CD will deploy immediately
- Daily/Weekly/Bi-Weekly/Monthly/Quarterly
- May have "hotfix" deployments
- May have "bugfix" vs "feature" deployments

Clean Deployments

Bad Deploys

- Impact many people
- Scramble to fix
- Looks bad with users

Team will want CLEAN deployments

- Testing gets you there
- Leave time for confidence

Feature Flags

Some way to turn a feature "on" for a user

- Allows incremental releases of a feature
- Allows for turn on/off without a deployment
- Details vary wildly
- Lots of `if()` in code
    - Later remove