The COS 333 Project

Robert M. Dondero, Ph.D.
Princeton University

Agenda

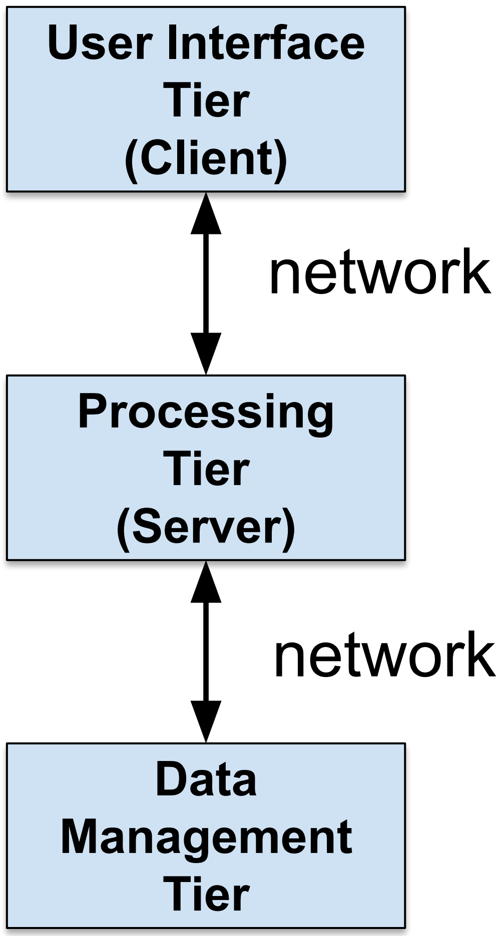
- **Overview**
- Process
- Deliverables

Overview

- A simulation of reality
- In teams of 3-5 people...

- Build a substantial networked ***three tier*** application

Overview



Allows users to access the app

Implements logic

Stores data persistently

Overview

- Working with instructors
  - First-level adviser: your TA
    - Will monitor & help
    - Will not manage
  - Second-level adviser: the lead instructor
    - Will monitor & help, directly or through your TA
    - Will not manage

Agenda

- Overview
- **Process**
- Deliverables

Process

- This is **not** a process:
  - Chat about the app for an hour or so
  - Hack some code together
  - Test it a bit
  - Do some debugging
  - Fix the obvious bugs
  - Repeat until the semester ends

Process

- Formal software engineering process models
  - Waterfall, agile, extreme,…
- Recommended informal 7-step process…

Process: Get Started

- **Step 1**: Get started
  - Find a topic
    - Check out *Previous Projects* web page
    - Check out *Project Ideas* web page
    - Look both inward and outward
    - Think both big and small

Process: Get Started

- **Step 1**: Get started
  - Form a team
    - Use ProjectFinder app (required)
    - Use Ed (optional)

Process: Get Started

- **Step 1**: Get started
  - Choose a leader
    - Goal: ***conceptual integrity*** (Brooks)

Process: Define Requirements

- **Step 2**: Define requirements
  - **Who** are the users?
    - Identify them by name
  - **What** should the app do?
    - Gather requirements
      - Interview users
      - Watch users work
    - Structure requirements
      - Compose scenarios
      - Compose wireframes, storyboards
  - **Involve the users!!!**

Process: Design

- **Step 3**: Design
  - **How** will the app work?

Process: Design

- **Step 3**: Design
  - Design “both ends toward the middle”
    - Early in the project: design your **UI**
    - Early in the project: design your **DB**
    - Rest of the project: connect the two

Process: Design

- **Step 3**: Design
  - Design module interfaces
    - Module = interface + implementation
    - Interface
      - The **public** part of a module
      - A module's **contract** with clients
      - Hides design decisions

Process: Design

- **Step 3**: Design
  - Choose technologies

Course goal	Use default technologies	Use non-default technologies
Learn 3-tier programming technologies	-	+
Learn software engineering	+	-

Process: Design

- **Step 3**: Design
  - Choose **user interface tier** technologies

Desktop app	Python**, PyQt5**, Java, Swing, …
Web app	HTML**, CSS**, Bootstrap**, JavaScript**, AJAX**, jQuery**, React**, …
Native mobile app	Java, Kotlin, Android*, Objective-C, Swift, iOS*, JavaScript**, ReactNative, …

** Default technology (covered in lectures/asgts)
* Covered in supplementary lecture material or appendices

Aside: React

- **React** is:
  - (pro) Hot!
  - (pro) Good for large projects
  - (con) Overkill for small projects
  - (con) Hard to learn

Process: Design

- **Step 3**: Design
  - Choose **processing tier** technologies

Language	Python**, Java, JavaScript**, ...
Framework	For Python: Flask**, Django*, … For Java: Spark*, Spring, … For JavaScript: Express*, …
Hosting service	Render**, Heroku*, …

** Default technology (covered in lectures/asgts)
* Covered in supplementary lecture material or appendices

Process: Design

- **Step 3**: Design
  - Choose **data management tier** technologies

Data store	Relational DBMS: PostgreSQL**, MySQL, … NoSQL DBMS: Redis, MongoDB, … Another app API **Don’t use SQLite!**
Hosting service	For PostgreSQL: ElephantSQL**, Render*, Heroku* For MongoDB: Atlas

** Default technology (covered in lectures/asgts)
* Covered in supplementary lecture material or appendices

Process: Design

- **Step 3**: Design
  - Suggestions for choosing technologies:
    - Talk with course instructors
    - Do many simple tech experiments early

Process: Implement

- **Step 4**: Implement
  - Compose module implementations
  - **Rule 1**: You need not compose all of the code, but the overall product must be your work
  - **Rule 2**: Every team member must compose a substantial amount of code

Process: Test

- **Step 5**: Test
  - Does the app work as **you** intend?
  - Integrated with Implementation step
  - Additional distinct step at the end

Process: Evaluate

- **Step 6**: Evaluate
  - Does the app work as its **users** intend?
  - Does the app fulfill the users’ needs?

Process: Document

- **Step 7**: Document
  - Integrated with previous steps
  - Additional distinct step at the end
    - *Grader's Guide* document
    - *Product Eval* document
    - *Project Eval* document

Process: General Advice

- Iterate
  - Iterate between **Implement** and **Test** frequently
  - Revisit **Define Requirements** and **Design** less frequently

Question (lecture05)

- Consider these 2 approaches:
  - **Approach 1**: Build the app so it requires a "big merge" where nothing works until everything works
  - **Approach 2**: Build the app in **stages** such that at all times (near end of semester) you have a working app
- Which of those two approaches is better, and why?
- Browse to https://cos333attend.cs.princeton.edu to answer

Process: General Advice

- Stage the work
  - Develop a MVP
  - Develop stretch goal 1
  - Develop stretch goal 2
  - …

Process: General Advice

- Do ***least-risk design***
  - Minimize risk
  - The module to develop next should be the one with maximal risk
  - The module to develop next should be the one which, if problematic, will have the largest negative impact on the app as a whole

Process: General Advice

- Use a version control system for all code
  - **Git** is mandatory
  - **GitHub** is mandatory

Process: General Advice

- Allocate time for “overhead” activities
  - Changing your mind
  - Disaster
  - Sickness
  - Health!
  - Deliverables…

Agenda

- Overview
- Process
- **Deliverables**

Deliverables

- Deliverables
  - See *Project* web page for details
  - See *Schedule* web page for due dates
  - All deliverables are graded

Deliverables

When	**Deliverable**
Pre-project	ProjectFinder entry
Pre-project	Project pre-approval meetings (optional)
Pre-project	***Project Approval Meeting***
Early project	***Team Directory***
Early project	***Project Overview*** document

Deliverables

When	**Deliverable**
Mid-project	Weekly status meetings
Mid-project	***Timeline*** document
Mid-project	***Wireframes***
Mid-project	Demo of ***Prototype***
Mid-project	Demo of ***Alpha version***
Mid-project	Demo of ***Beta version***

Deliverables

When	Deliverable
Reading Period	***Presentation***
Dean’s Date	***Grader’s Guide*** doc
Dean’s Date	***Product Eval*** doc
Dean’s Date	***Project Eval*** doc
Dean’s Date	The application

Keys to Success

- Keys to success in COS 333:
  - Find a good project
  - Find good teammates