# Intro to Coding with Python– Prototyping

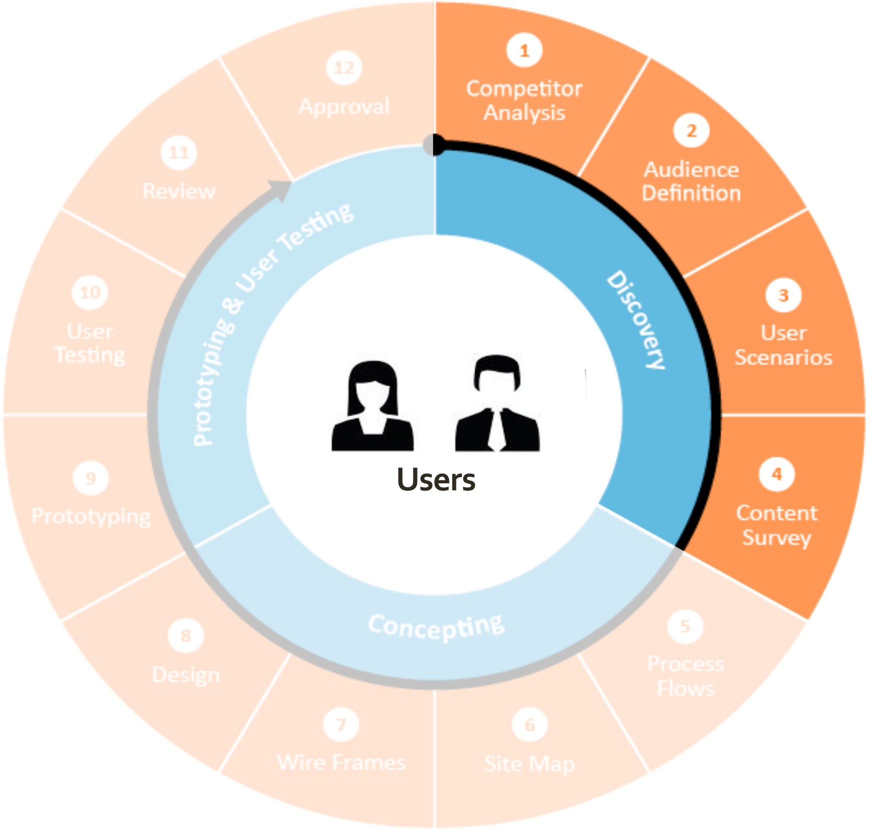
Dr. Ab Mosca (they/them)

Slides based off slides courtesy of Jordan Crouser (https://jcrouser.github.io/)

# Plan for Today

- User-centered design
  - Prototyping

# User-centered design framework



## 1) Discovery

- Learning about your users
- Modeling your users
- Analyzing your users’ tasks
- Eliciting and defining clear product requirements

## 2) Concepting Phase

- Developing conceptual models
- Solving design problems through ideation
- Detailed design activities

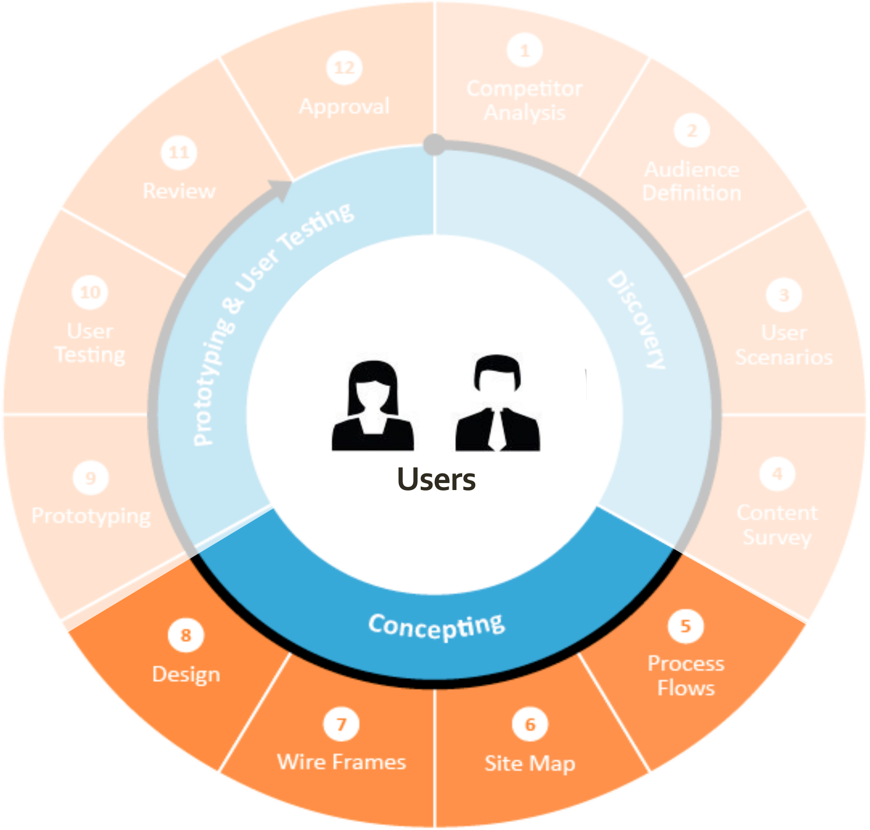
## 3) Prototyping + User Testing

- Delivery of a high-quality product that meets users’ needs and is easy to learn and use

http://www.uxmatters.com/mt/archives/2010/07/design-is-a-process-not-a-methodology.php

Now that we’ve got some end users in mind, what would a **prototype** look like?

# User-centered design framework



1) Discovery

- Learning about your users
- Modeling your users
- Analyzing your users' tasks
- Eliciting and defining clear product requirements

2) **Concepting Phase**

- Developing conceptual models
- Solving design problems through ideation
- Detailed design activities

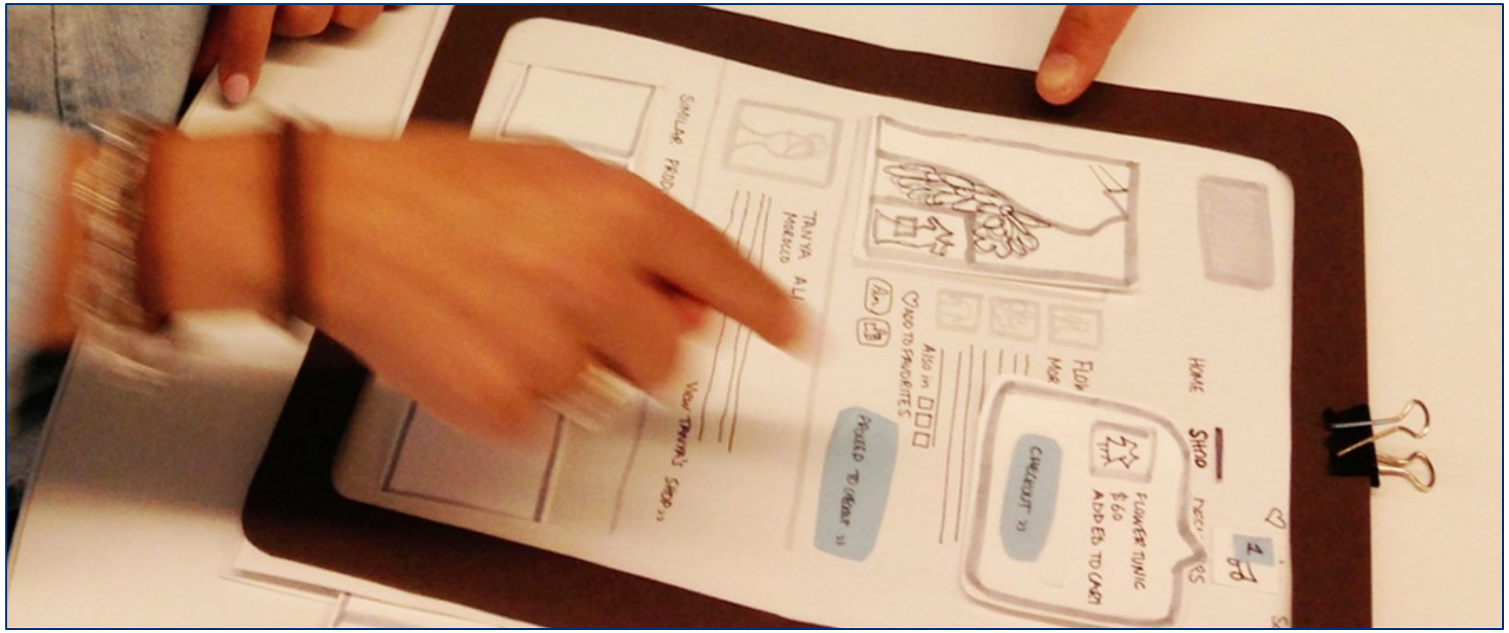
3) **Prototyping + User Testing**

- Delivery of a high-quality product that meets users' needs and is easy to learn and use

http://www.uxmatters.com/mt/archives/2010/07/design-is-a-process-not-a-methodology.php

# CS Life Skill #1: "Paper prototyping"

- **Big idea:**
  - Not sure yet whether or not an **idea** will work?
  - Making a **paper version** of an interface is a lot faster and easier than coding a working prototype – start there!



# “Paper prototyping” goals

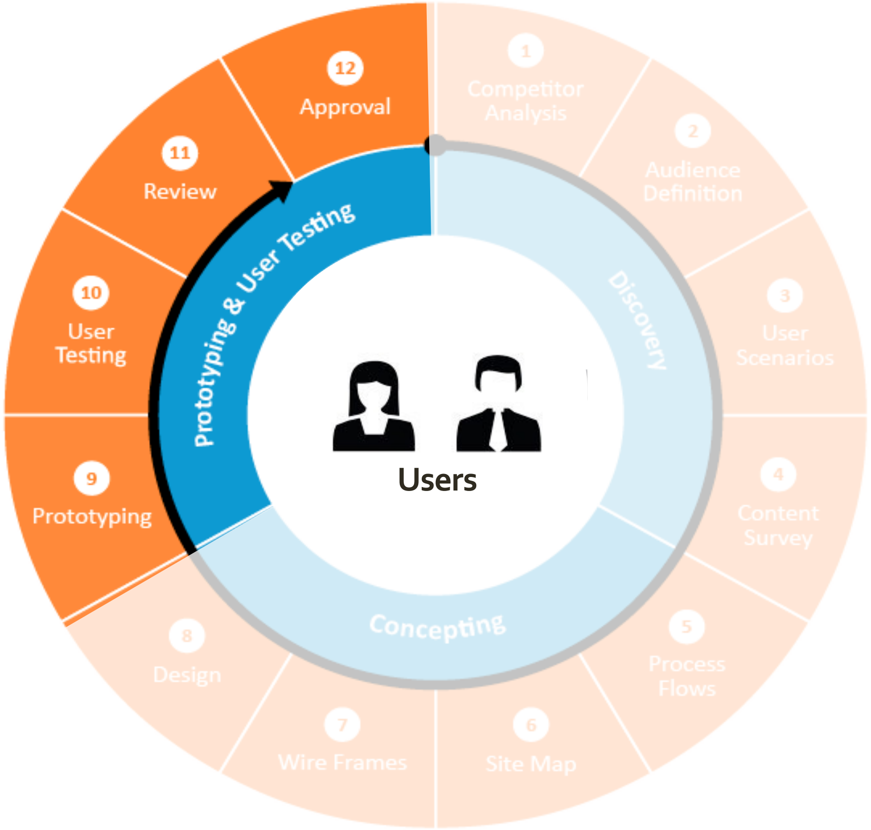
- Generate **lots of ideas**
- Engage **other people** in the design process
- Identify **potential problems** before you waste time coding
- Get **feedback** quickly, from lots of different people
- Some tips:
  - Focus on the **big picture**, don’t worry about the details
  - **Think about what you want it to do,** rather than what you know how to implement (we’ll worry about that later)
  - Not so into arts and crafts? It doesn’t have to be **actual paper**… Whiteboard / PowerPoint / Keynote will also do the trick!

# “Paper prototyping”

Examples:

- https://www.youtube.com/watch?v=nA9QP9lkl2o
- https://www.youtube.com/watch?v=y20E3qBmHpg
- https://www.youtube.com/watch?v=yafaGNFu8Eg

# User-centered design framework



## 1) Discovery

- Learning about your users
- Modeling your users
- Analyzing your users' tasks
- Eliciting and defining clear product requirements

## 2) Concepting Phase

- Developing conceptual models
- Solving design problems through ideation
- Detailed design activities

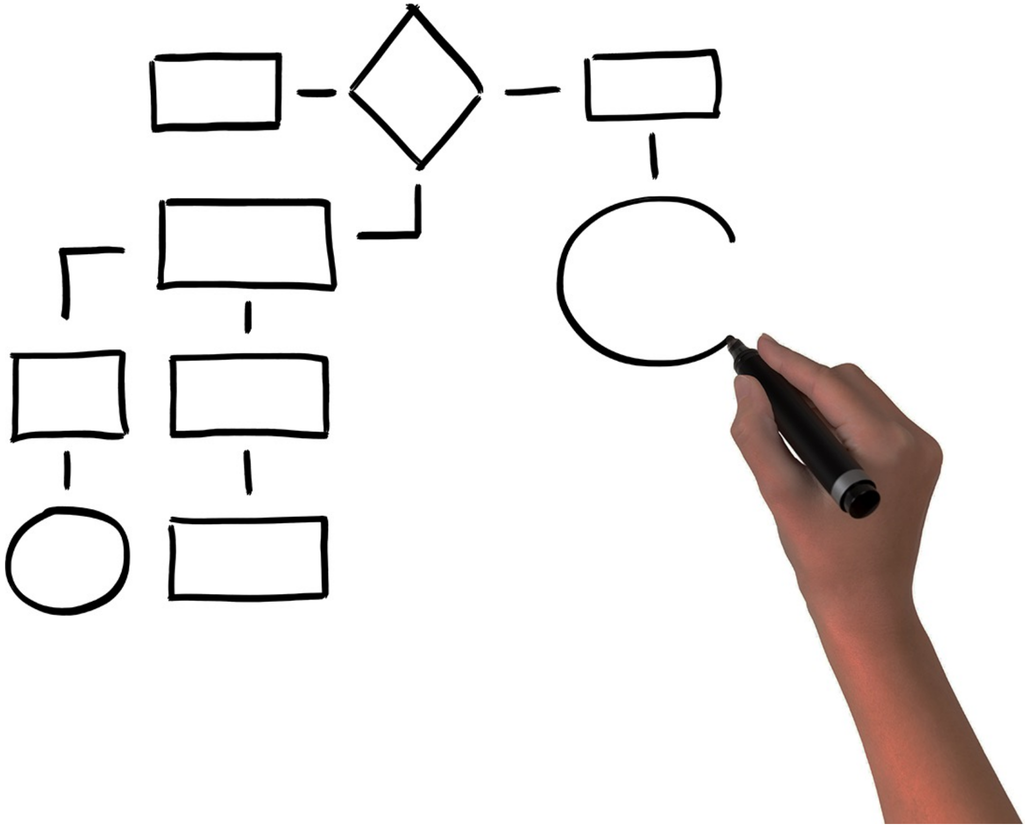
## 3) Prototyping + User Testing

- Delivery of a high-quality product that meets users' needs and is easy to learn and use

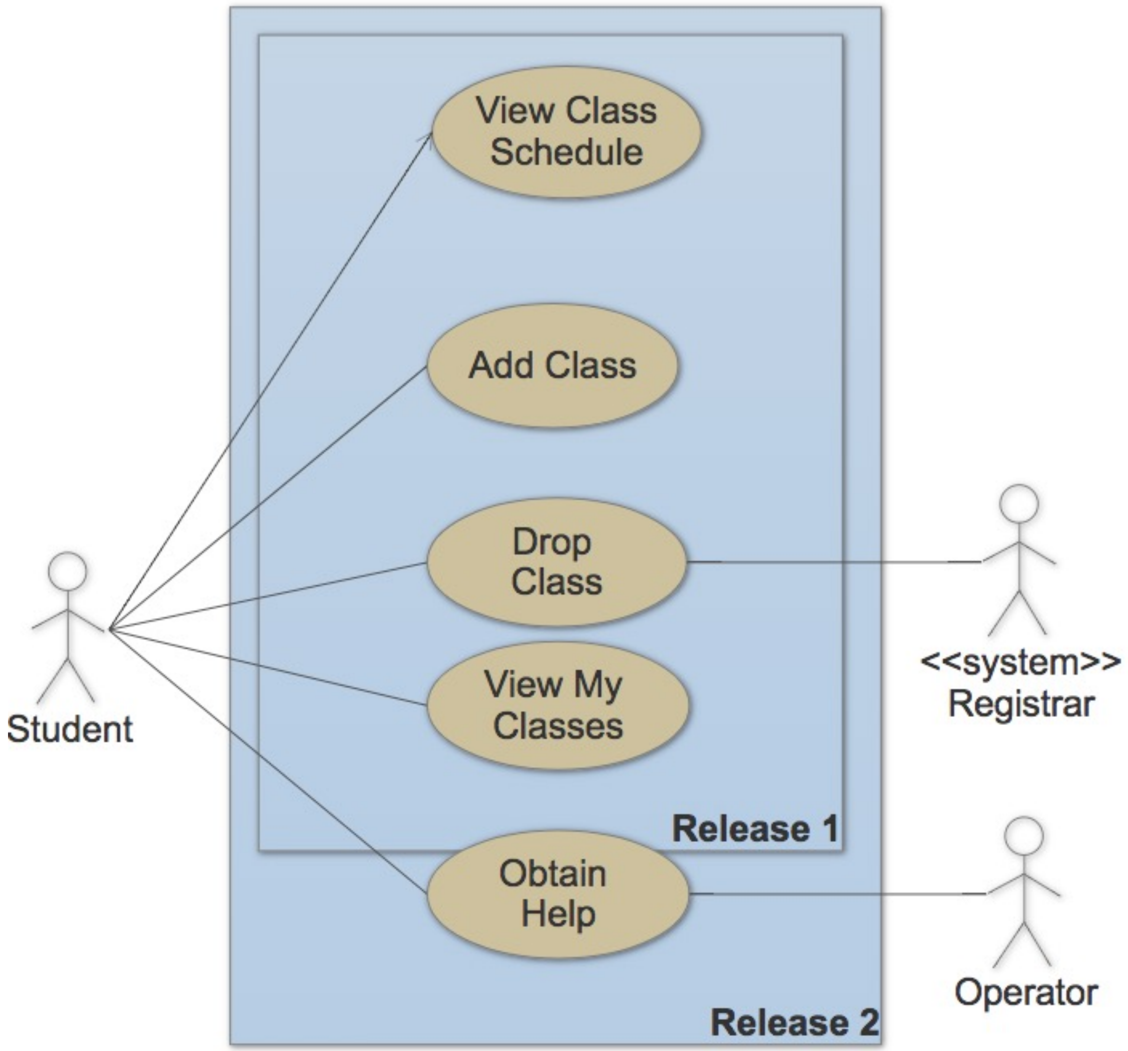
http://www.uxmatters.com/mt/archives/2010/07/design-is-a-process-not-a-methodology.php

# "Architecture diagrams"

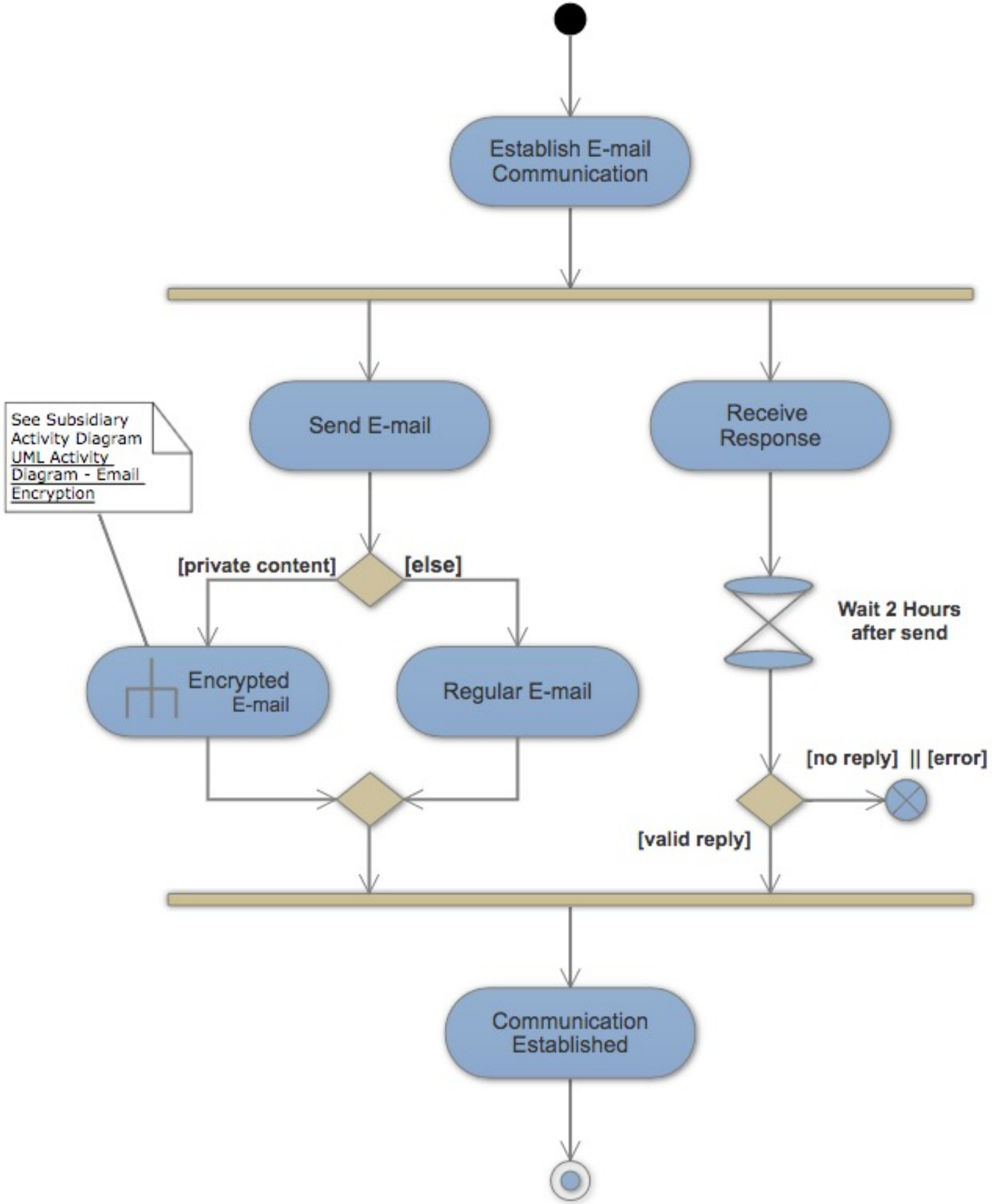
- **Big idea:**
  - Now that you've got an idea of what your interface might look like, break that down into **manageable pieces** so you can get started
  - This can happen at **several levels of detail**



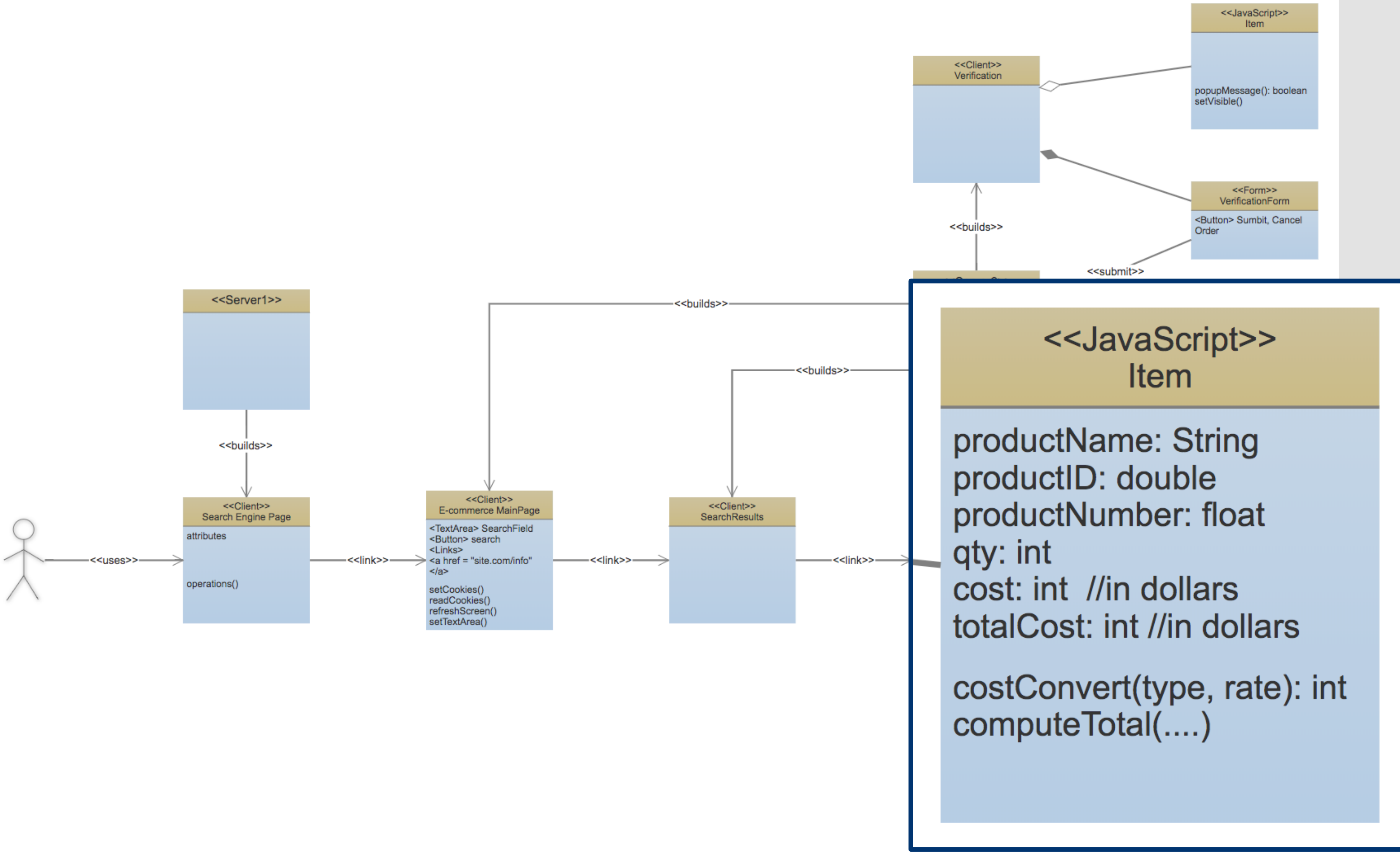
# Example: use case diagram (high level)



# Example: activity diagram (mid level)



# Example: class diagram (low level)



<<JavaScript>>
Item

popupMessage(): boolean
setVisible()

<<Client>>
Verification

<<Form>>
VerificationForm

<Button> Sumbit, Cancel Order

<<builds>>

<<submit>>

<<Server1>>

<<builds>>

<<builds>>

<<builds>>

<<Client>>
Search Engine Page

attributes

operations()

<<uses>>

<<link>>

<<Client>>
E-commerce MainPage

<TextArea> SearchField
<Button> search
<Links>
<a href = "site.com/info"
</a>

setCookies()
readCookies()
refreshScreen()
setTextArea()

<<link>>

<<Client>>
SearchResults

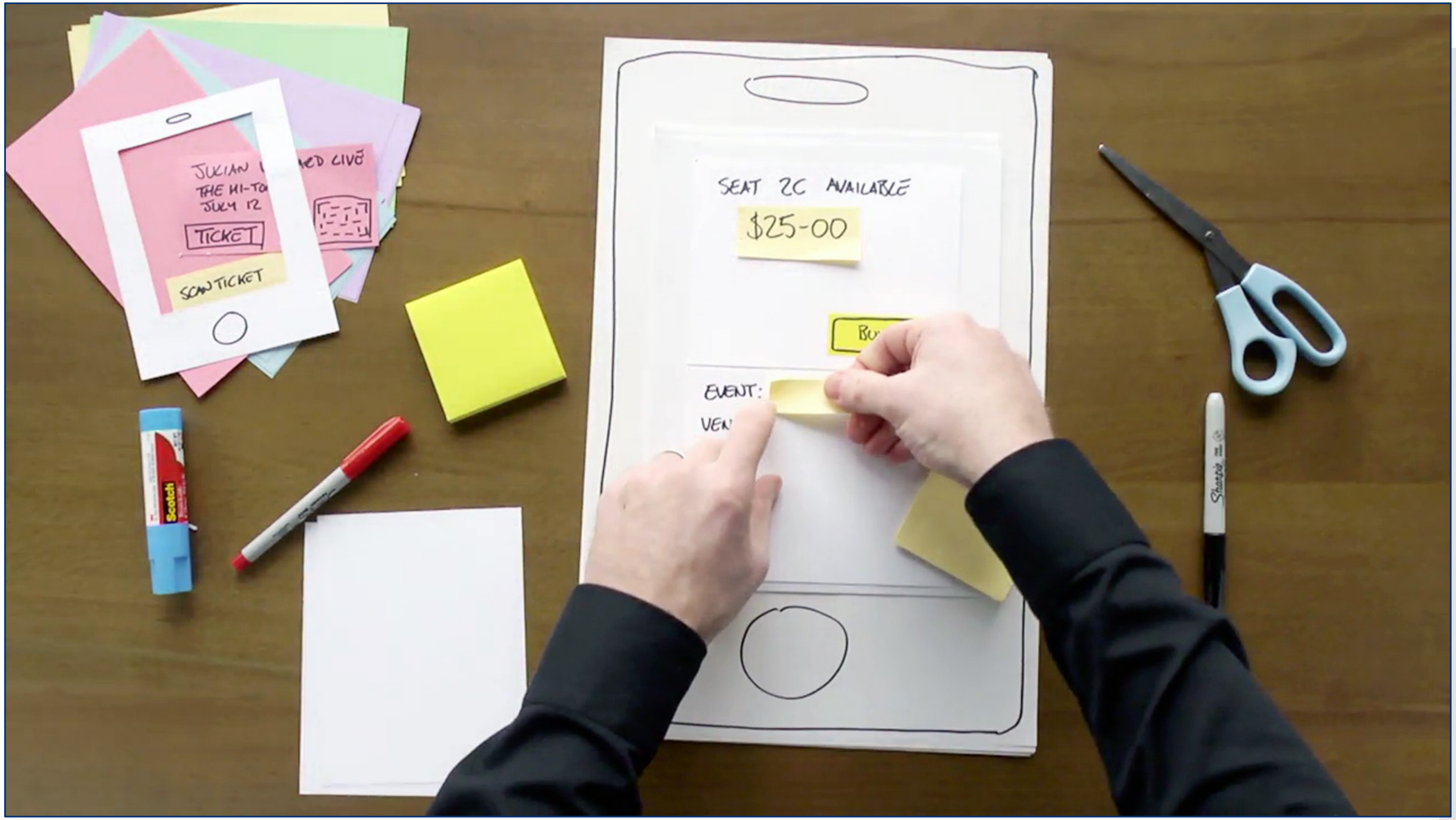
<<link>>

<<JavaScript>>
Item

productName: String
productID: double
productNumber: float
qty: int
cost: int //in dollars
totalCost: int //in dollars

costConvert(type, rate): int
computeTotal(....)

# Your turn!



Work with a partner to create a paper prototype for a transit app. Take a picture of your prototype to turn in on Gradescope.

# Takeaways

- Thinking about your end user early → you're more likely to **build something that actually solves the problem**
- **"Low-fidelity" prototyping** saves time and energy by helping identify problems before you commit to code
- **Architecture diagrams** help you plan out your implementation so you don't run out of time
- Also, the process is **kinda fun**…