



BACS 385

Fundamentals of Project Management

Why is this class important?

- Learn how to:

- Work effectively as part of a software development team
- Follow recognized development methodology
- Communicate effectively through written, oral and other forms of technical presentation
- Manage software development using industry standards

What is the structure of this class?

Short Lectures

Two Quizzes

Comprehensive
Final

short answers
and multiple
choice questions

Be part of a software
development team
(every day)

Meet with Clients

Gather Requirements

Manage Tasks

*follow
industry
standards*

- *There is no required book for this class*
- *All In-Class assignments will be submitted on canvas*
- *Expect to use Google Docs, Google Slides, and Draw.io.*
- **Late Policy:** Everyday CANVAS will **automatically deduct 8%** (late penalty) from any missing assignments that have passed due-date and has not been turned-in.

What is the emphasis in this class?

Requirements
Gathering

Requirements
Documentation

Task Creation

Task
Management

*Understand customer
expectations*

Generate artifacts

*Breakdown
requirements to
manageable tasks*

*Ensure task
completion is
monitored and
maintained*

*Articulate system
requirements by asking
what and why*

*Consider available
time, resources &
skills*

*Be ready to put out
fires*

Goals Today

On Canvas, complete:

- *Student Knowledge Assessment*



BACS385: 2nd Day of Class

- Agenda -

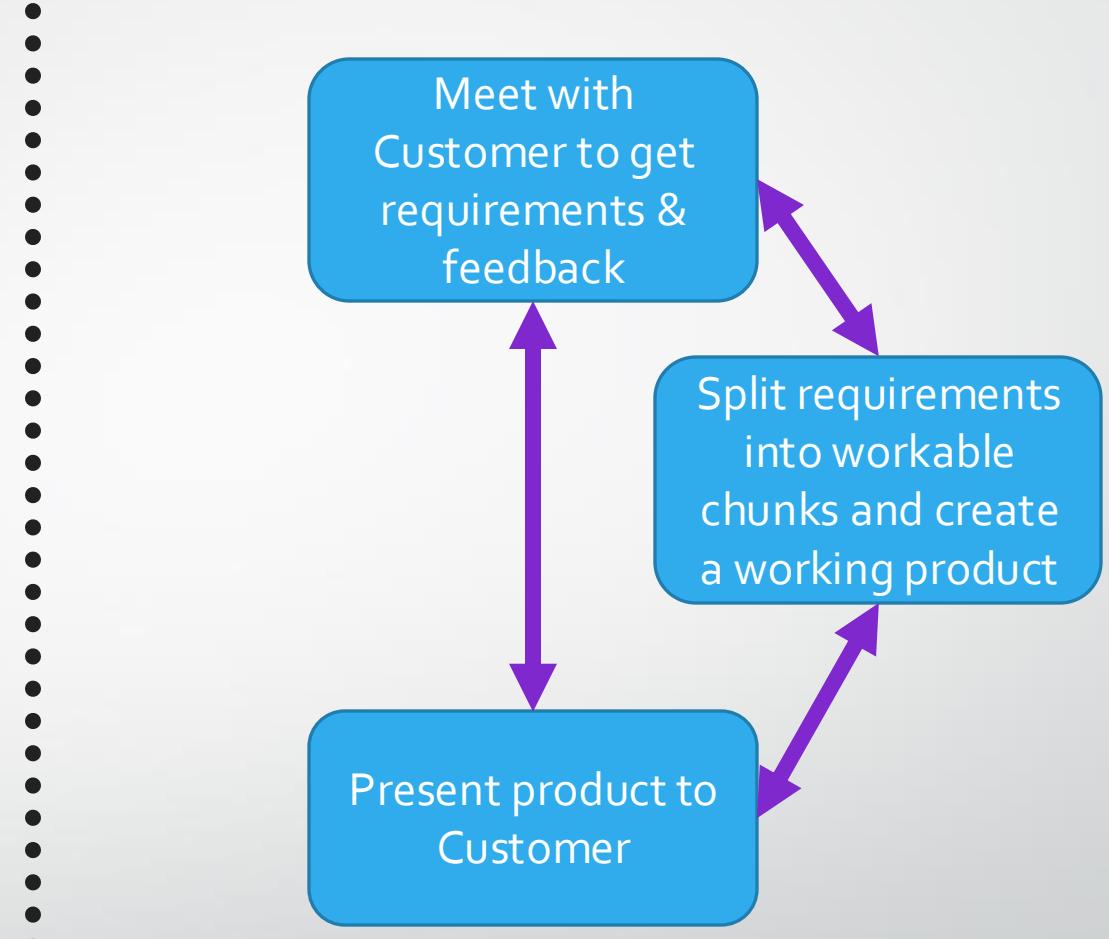
- Breakout into your Teams: In Class Activity #1
- Intro to Software Development Process & Life Cycle

Exercise

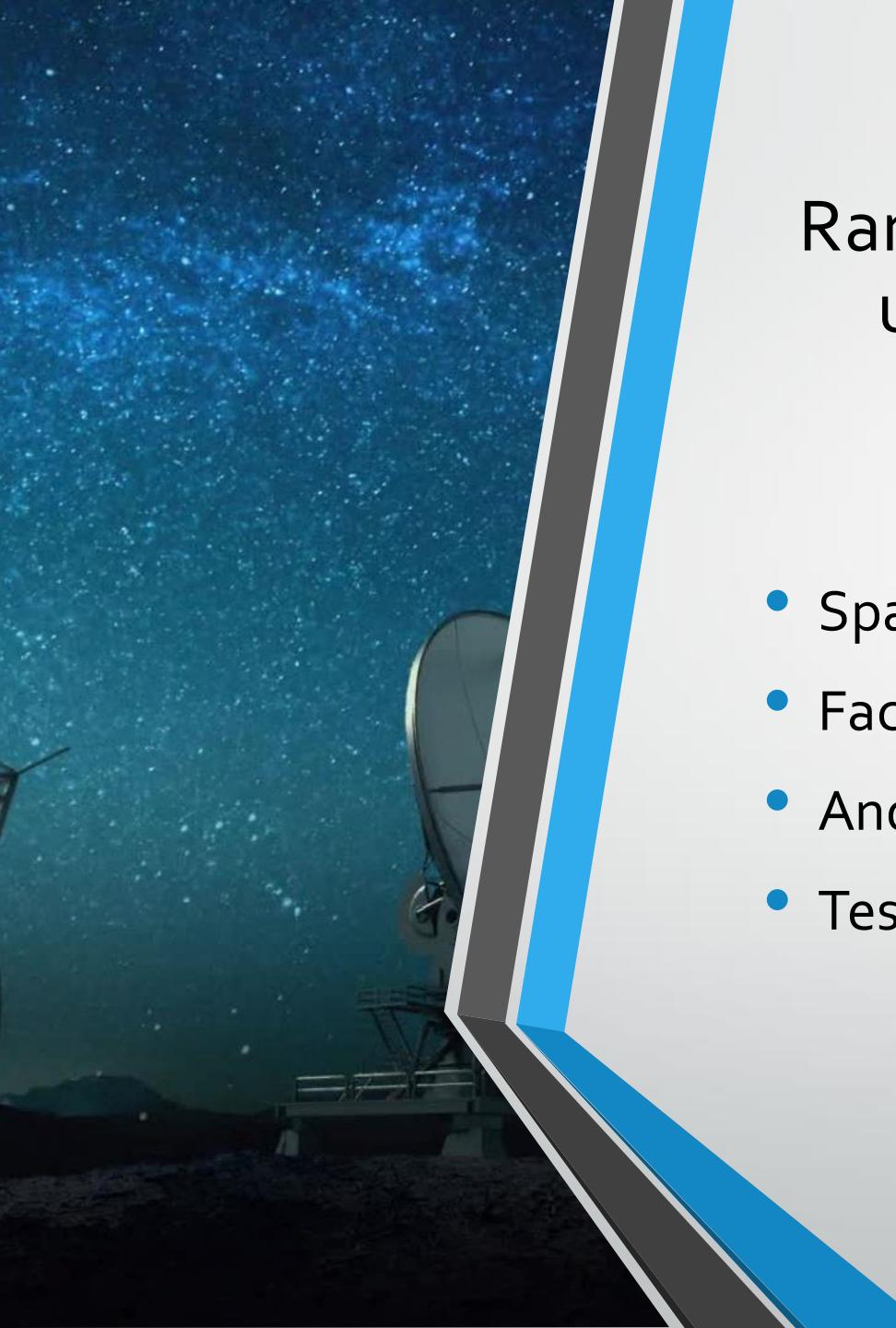
- You are the head of a software development company specializing in restaurant automation. You receive the following project description that your team needs to address:
 - *"ChefBots Inc. requires a software solution that enables restaurant managers and staff to monitor and track the progress of robots working in their restaurant. The software should provide real-time updates on the robots' tasks, including food preparation, serving, and cleaning. It should also include features for staff to intervene or provide instructions when necessary, and to gather data on robot efficiency and customer satisfaction."*
- As the project lead, what steps would you undertake to develop this software? This is a complex task, so in a **list of 10 steps**, detail the actions you would take.
 - Form a group of 5 colleagues, utilize Google Docs, and brainstorm a comprehensive plan for designing, developing, and implementing this restaurant robot tracking software.
- Requirements:
 - Do not use ChatGPT, I need you to critically think about an approach to tackling this request.
 - There are no right or wrong answers to this question



Linear Approach



Flexible Approach

A large satellite dish antenna is positioned on the left side of the slide, set against a dark blue background filled with numerous small white stars, suggesting a night sky or space environment.

Rank each system from most to lowest # of updates/changes after product release

- SpaceX's Starship spacecraft
- Facebook
- Android OS
- Tesla Model 3

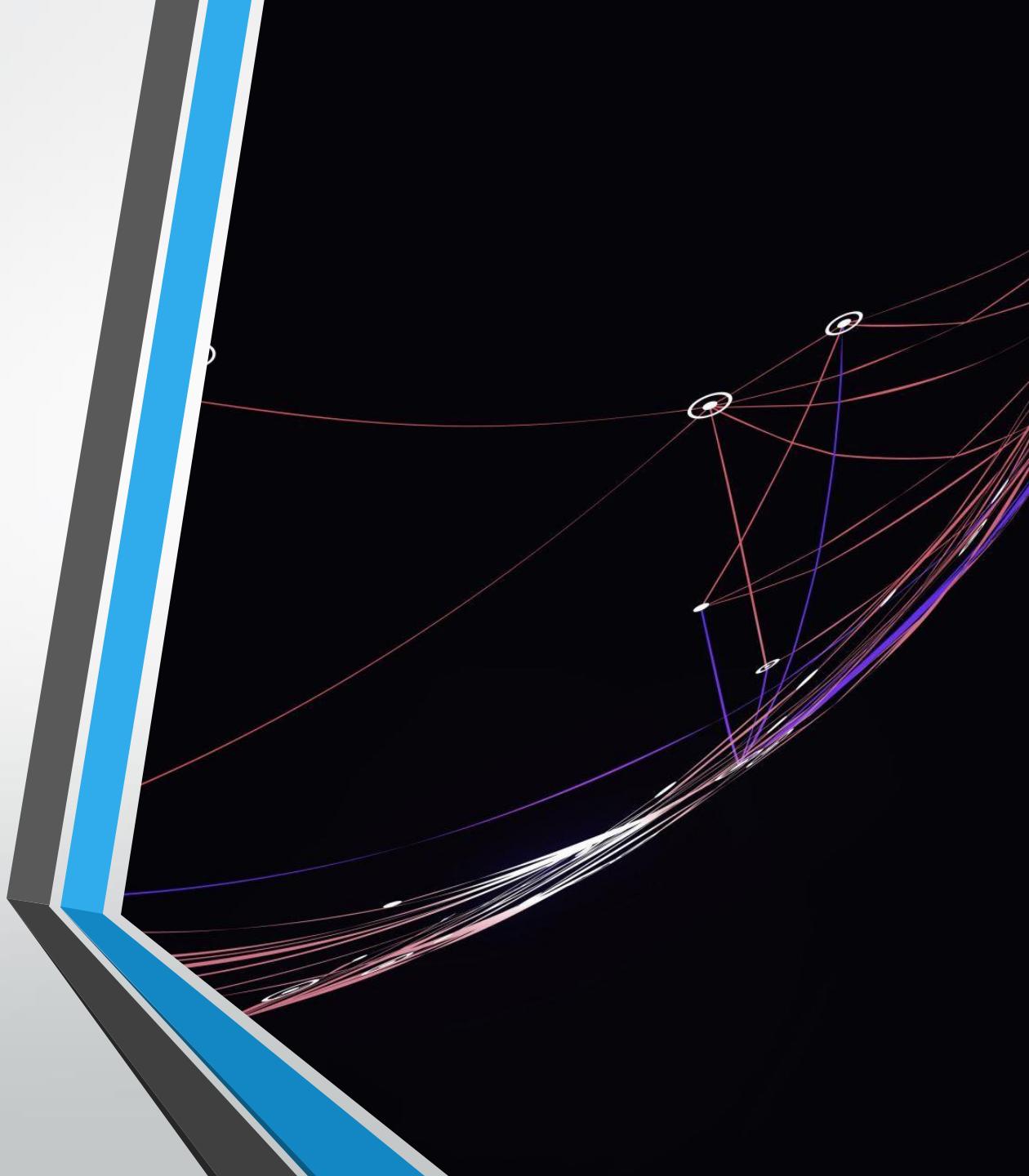


Rank each system from most to lowest # of updates/changes after product release

1. Facebook (over 2797 million active users)
2. Android OS (over 2.5 billion active users)
3. Tesla Model 3 (over 3 million **tesla** cars sold)
4. SpaceX's Starship spacecraft

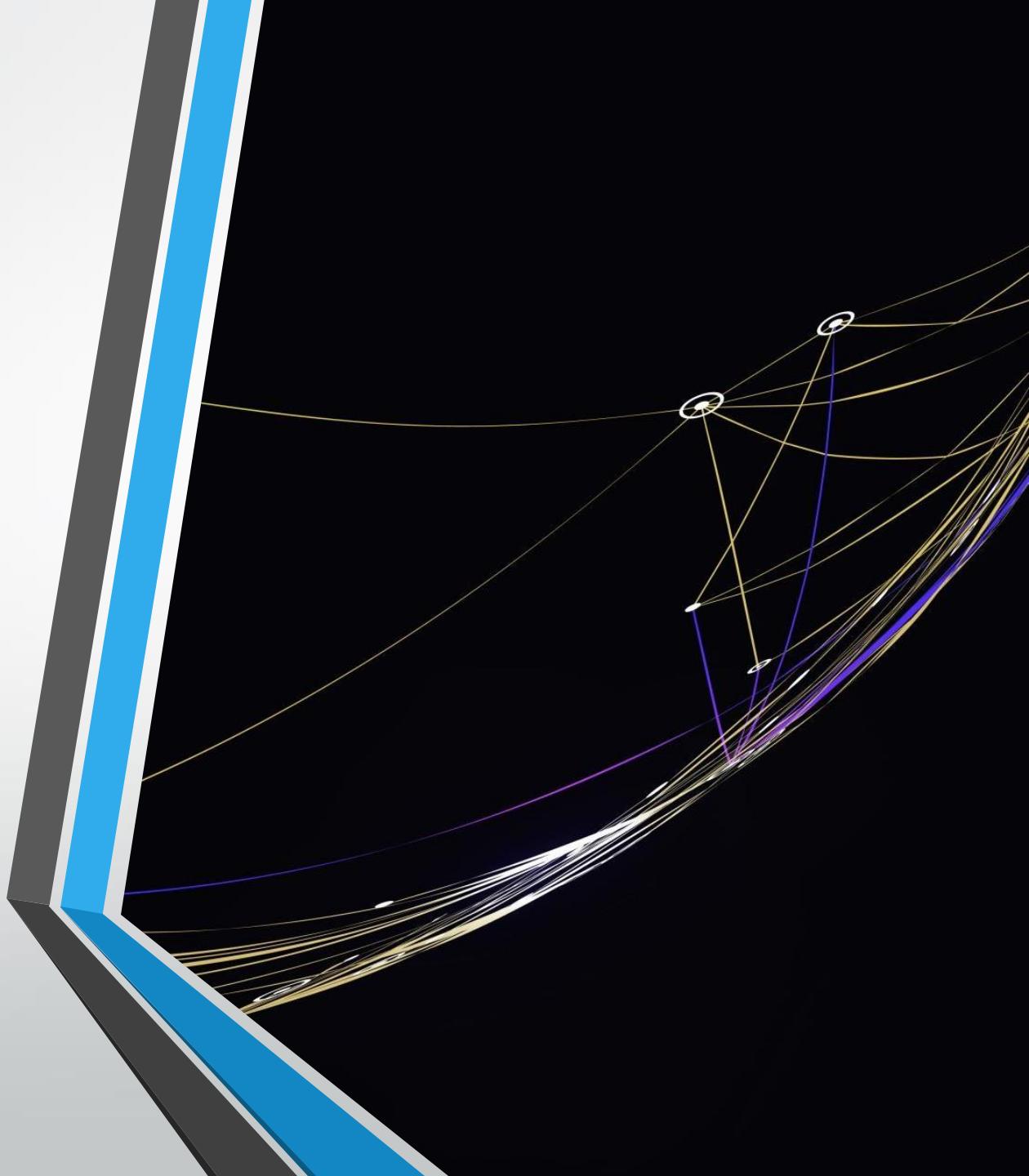
During the product development of each of the following systems, which development approach (Linear vs. Flexible) would you use? And why?

1. Facebook (over 2797 million active users)
2. Android OS (over 2.5 billion active users)
3. Tesla Model 3 (over 3 million **tesla** cars sold)
4. SpaceX's Starship spacecraft



During the product development of each of the following systems, which development approach (Linear vs. Flexible) would you use? And why?

1. Android OS ----- Flexible
2. Facebook ----- Flexible
3. Tesla Model 3 ----- Mix of Linear + Flexible
4. SpaceX's Starship spacecraft ----- Mostly Linear



BACS385: 3rd Day of Class - Agenda -

- Continue with Software Development Process & Life Cycle
- Requirements Gathering and Documentation
- Breakout into your Teams: In Class Activity #2

What is a Process ... ?

- When we provide a service or create a product, we always follow a sequence of steps to accomplish a set of tasks
- We can think of a series of activities as a **process**
- A process is a series of steps taken to achieve a particular end.
 - **Building a House:** Just like you wouldn't install drywall before the wiring in a house, software development requires certain steps in a specific order.
 - **Baking a Cake:** You wouldn't bake a cake before mixing the ingredients. Similarly, in software, certain foundations need to be laid down first.
- Think of a popular app like Instagram. Instagram wasn't built in a day; it followed a process from idea to implementation.
- In the context of software development, which of these steps is crucial to perform first?
 - **Option A:** Coding the software's main features.
 - **Option B:** Gathering and documenting user requirements.
 - **Option C:** Designing the user interface.
 - **Option D:** Testing the final product.

Processes and software



Software (unlike
buildings/bridges etc.)

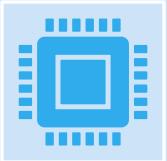
Can be changed at anytime

Is often required to change often after
construction



Benefits

Software can be improved almost without
limit



Leading to problems

Software often gets faults as it evolves

Software cost is hard to manage

Problems with user's experience and
expectations

The Software Process

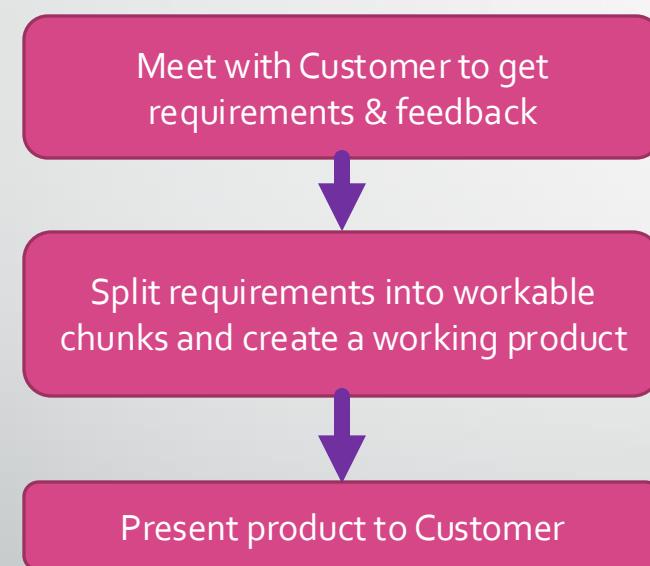
- Software development process is the process of dividing software development work into distinct phases to improve design, product management, and project management.
- It is also known as a **software development life cycle (SDLC)**.
- The Software Development Life Cycle (SDLC) is a systematic process for building software that ensures the quality and correctness of the software built.

Software Development Process

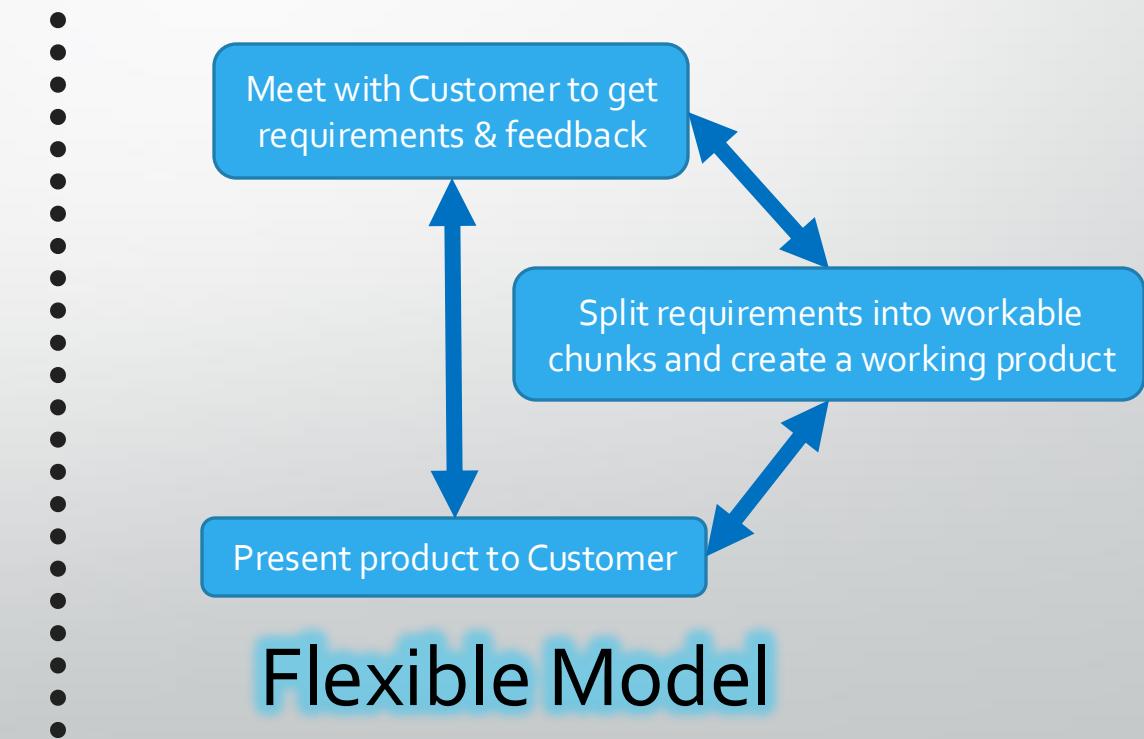
- Take an abstract idea (e.g., inside of customers head) and build a concrete/functional system that implements that idea.
- Fundamental in breaking down an idea into smaller steps that can be tackled individually.
- A structured set of activities/phases that include:
 - **Specification (Requirements Gathering and Documentation)**
 - **Design and implementation**
 - **Validation & Verification (Product Testing)**
 - **Deployment & Evolution (Product Maintenance)**

Software Process Model

An abstract representation of a process that presents a description of a process from some particular perspective.



Linear Model



Flexible Model



Software Specifications

(also known as *Requirements Engineering*)

- The goal of this step/phase of software development is to establish
 - A need or desire perceived by a stakeholder (e.g., customer)
 - A capability or property that a system should have
 - A documented representation of a need, property or capability



How the customer explained it



How the Project Leader understood it



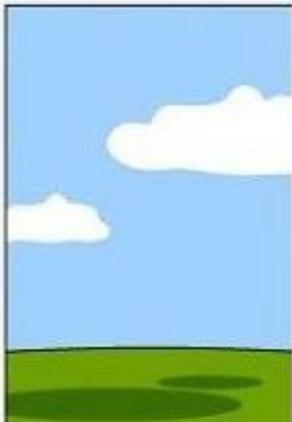
How the System Analyst designed it



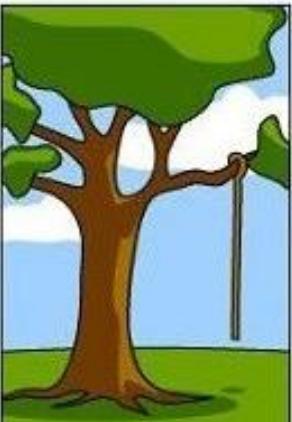
How the Programmer wrote it



How the Business Consultant described it



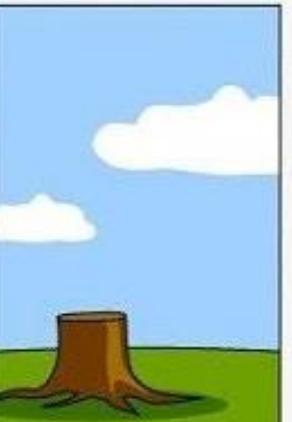
How the project was documented



What operations installed



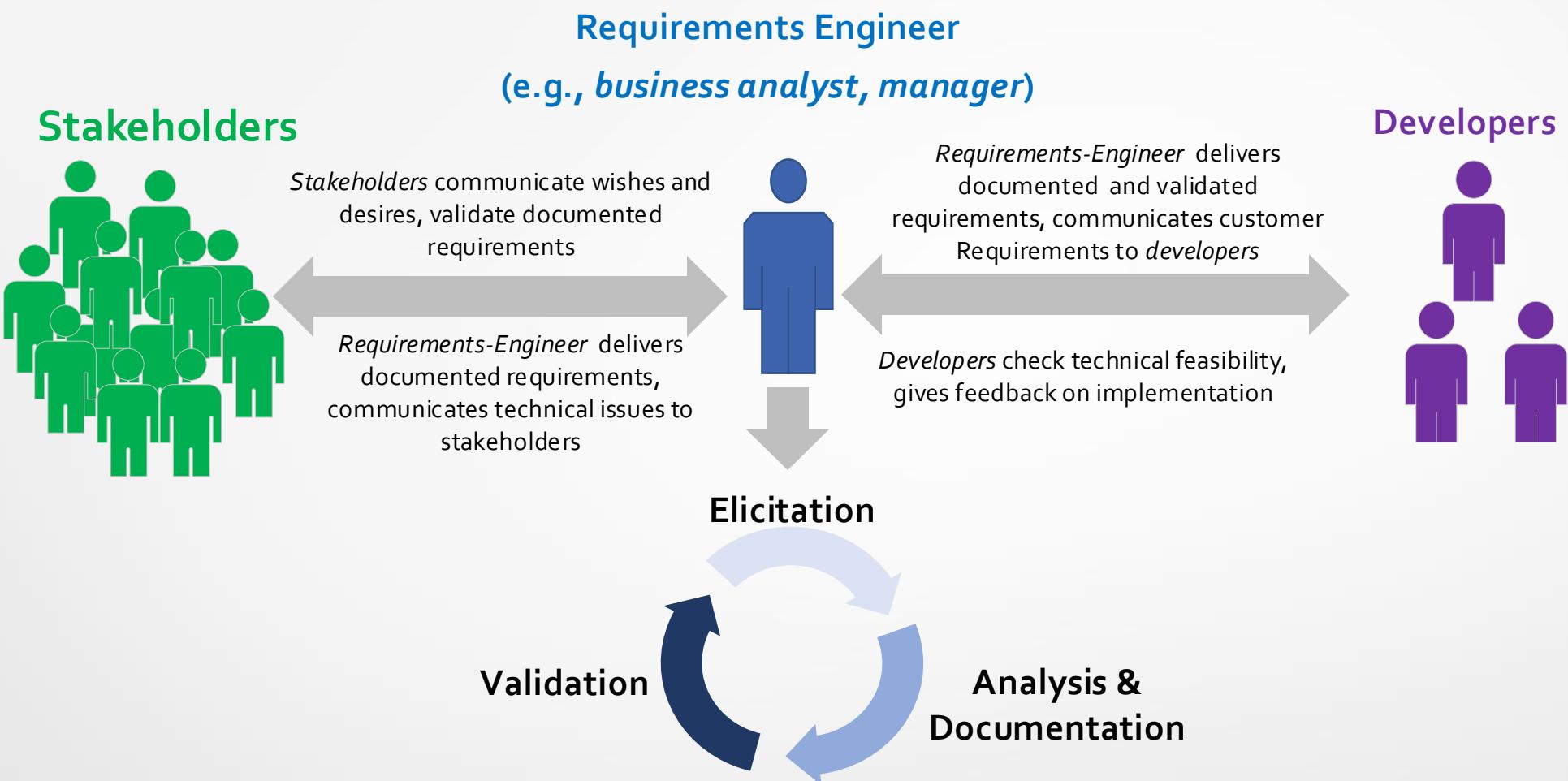
How the customer was billed



How it was supported



What the customer really needed



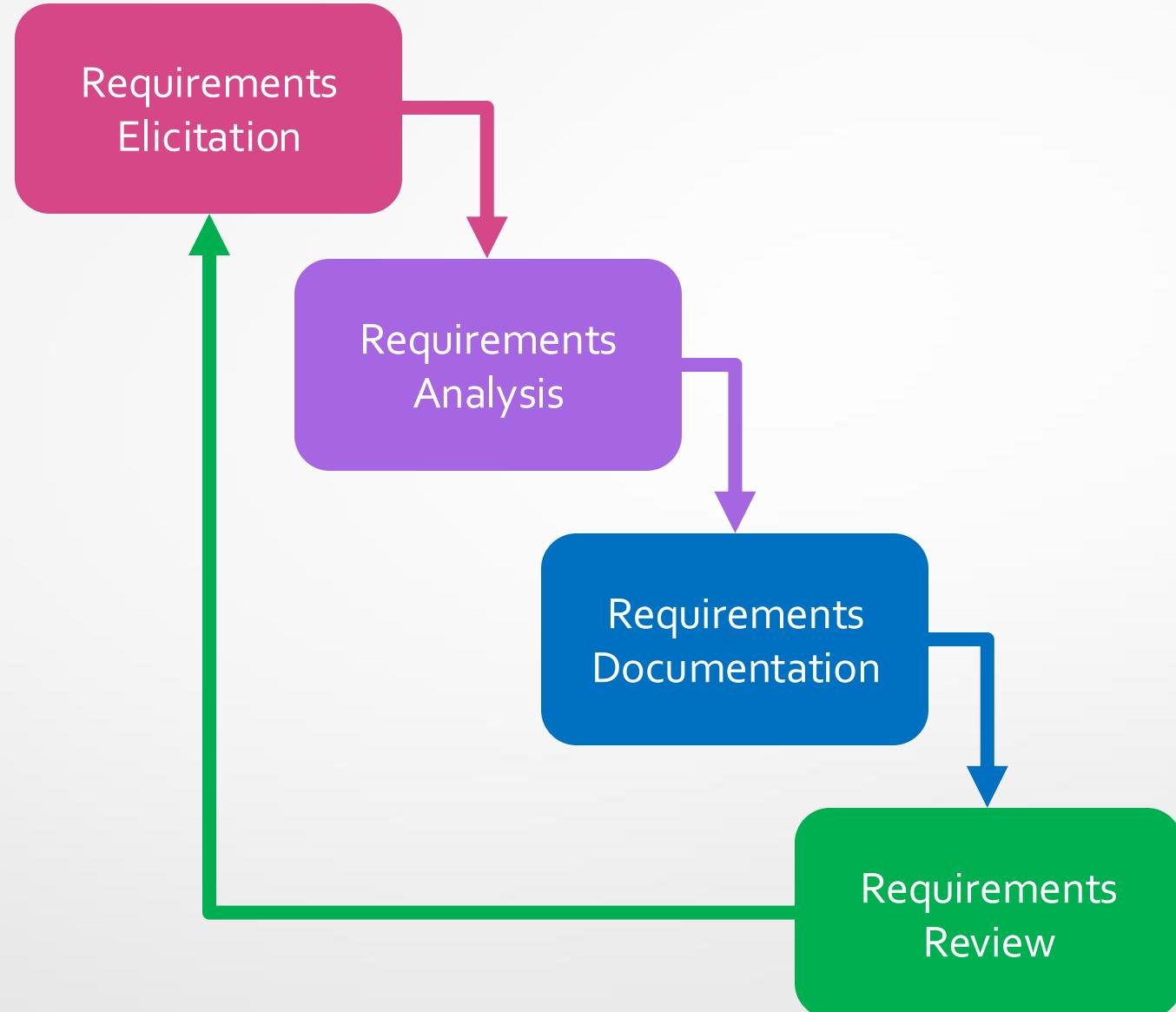
The role of a *Requirements-Engineer* can be described as an interface between the stakeholders and the development team

Software Specification... continued...

- **Requirements Gathering & Elicitation:** the process through which the customers, buyers, or users of a software system discover, reveal, articulate, and understand their requirements.
 - Business analyst/Manager/Requirements Engineer meets with stakeholders
- **Requirements Analysis:** the process of reasoning about the requirements that have been elicited; it involves activities such as examining requirements for conflicts or inconsistencies, combining related requirements, and identifying missing requirements.
 - Business analyst/Manager/Requirements Engineer analyzes and reviews requirements with development team

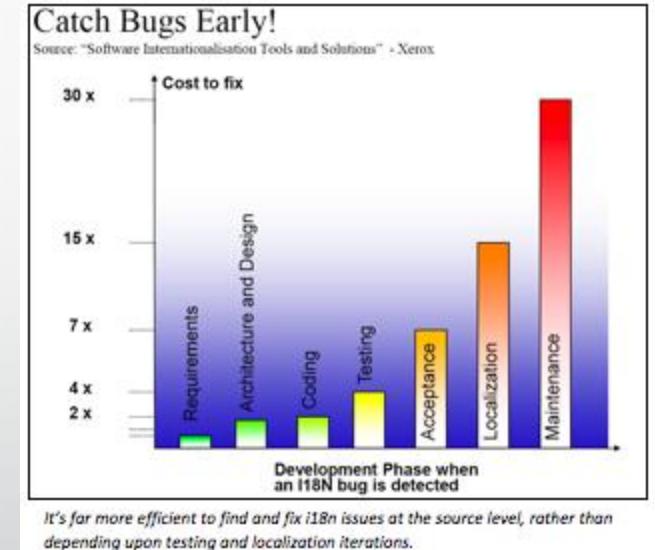
Software Specification... continued...

- **Requirements Documentation:** the process of recording the requirements in one or more forms, including natural language and formal, symbolic, or graphical representations; also, the product that is the document produced by that process.
 - Business analyst/Manager/Requirements Engineer documents the requirements after discussing with development team
- **Requirements Validation:** the process of confirming with the customer or user of the software that the specified requirements are valid, correct, and complete.
 - Business analyst/Manager/Requirements Engineer meets with stakeholders to review the documented requirements to get feedback



The cost of fixing bugs late...

- The Systems Sciences Institute at IBM has reported that “the cost to fix an error found after product release was four to five times as much as one uncovered during design, and up to 100 times more than one identified in the maintenance phase.”
- A 2003 study commissioned by the Department of Commerce’s National Institute of Standards and Technology found that software bugs cost the US economy \$59.5 billion annually.



In Class Activity #2:

- This activity requires the following steps:

Form your groups, then think of a software idea that does not exist. Try to think of something that is almost impossible to create in today's world.

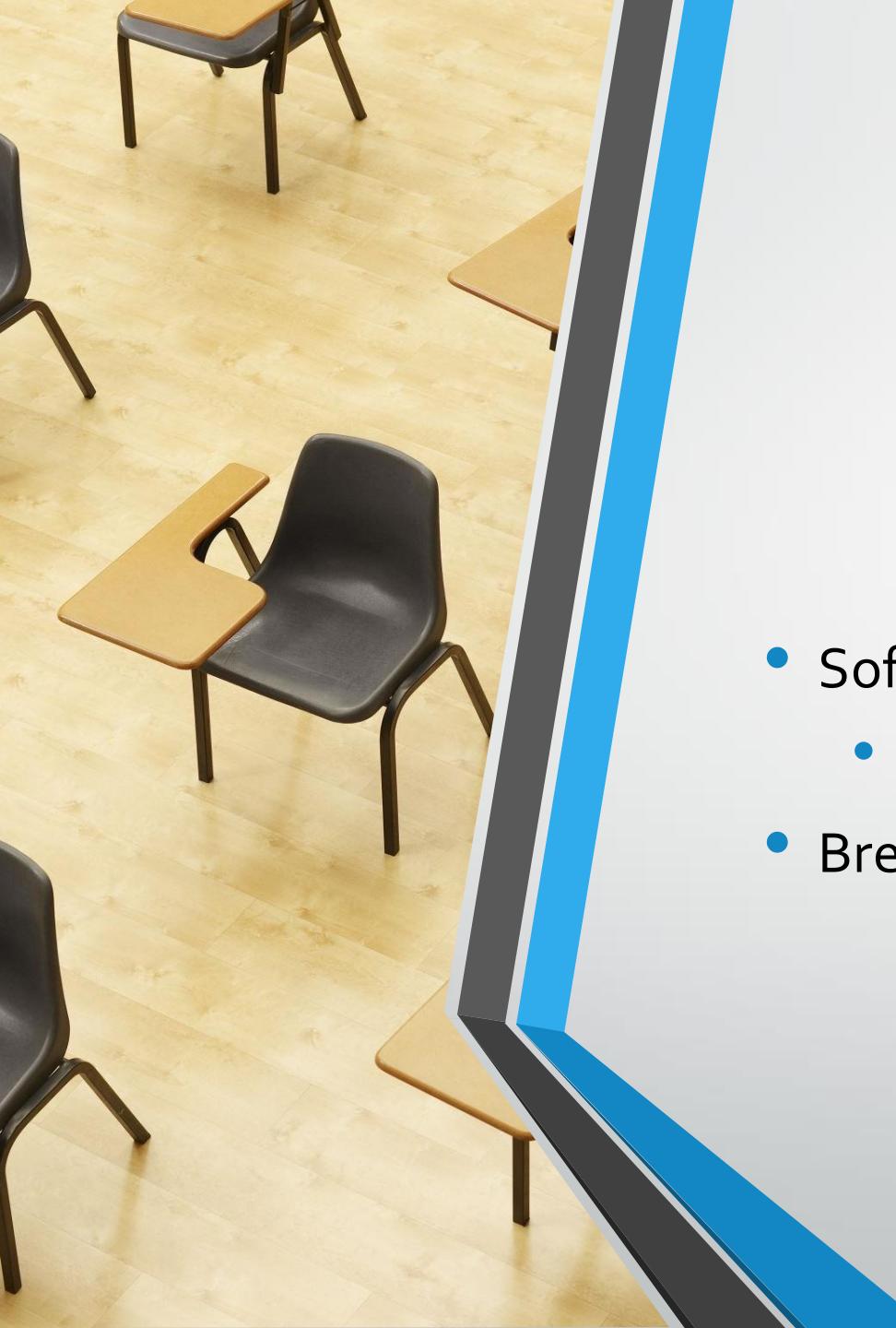
- Write a paragraph describing your product in high level detail
- Write down 10 software specifications



BACS385: 4th Day of Class

- Agenda -

- Recap: In Class Activity #2
- Breakout into your Teams: In Class Activity #3



BACS385: 5th Day of Class

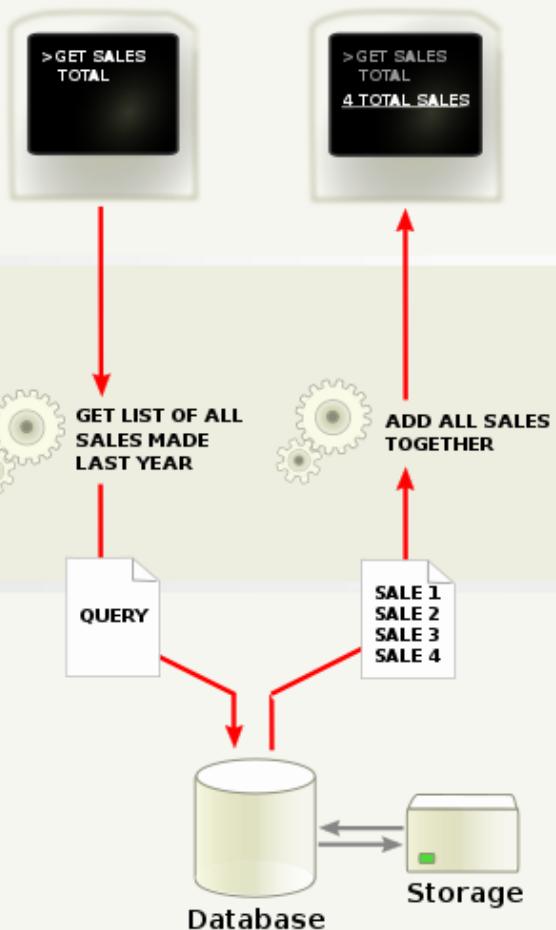
- Agenda -

- Software Specifications:
 - User Stories
- Breakout into your Teams: In Class Activity #4

Structure of Software

Presentation tier

The top-most level of the application is the user interface. The main function of the interface is to translate tasks and results to something the user can understand.

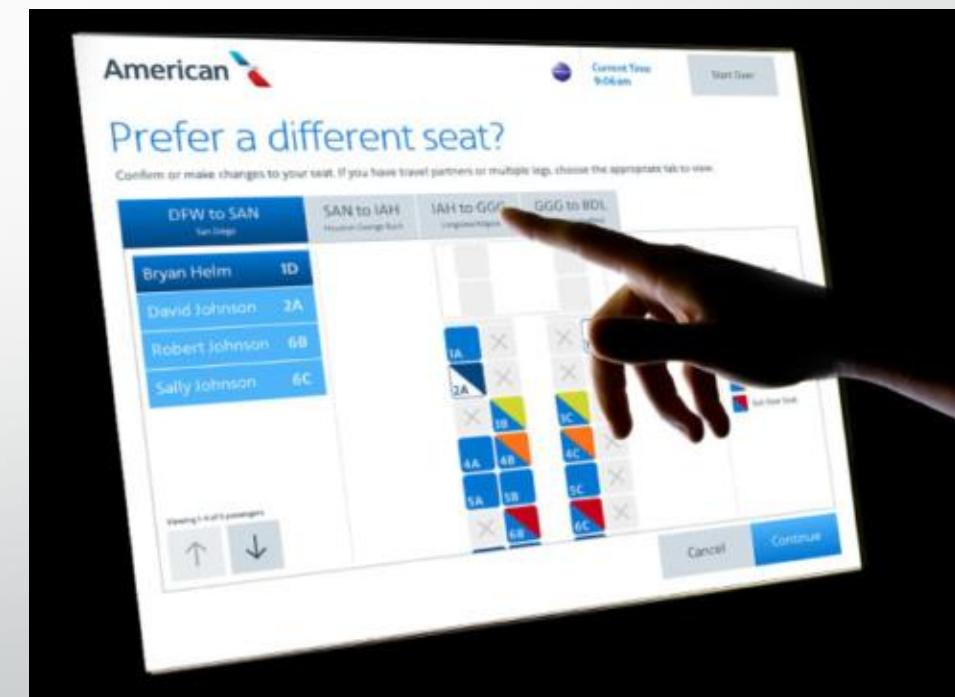


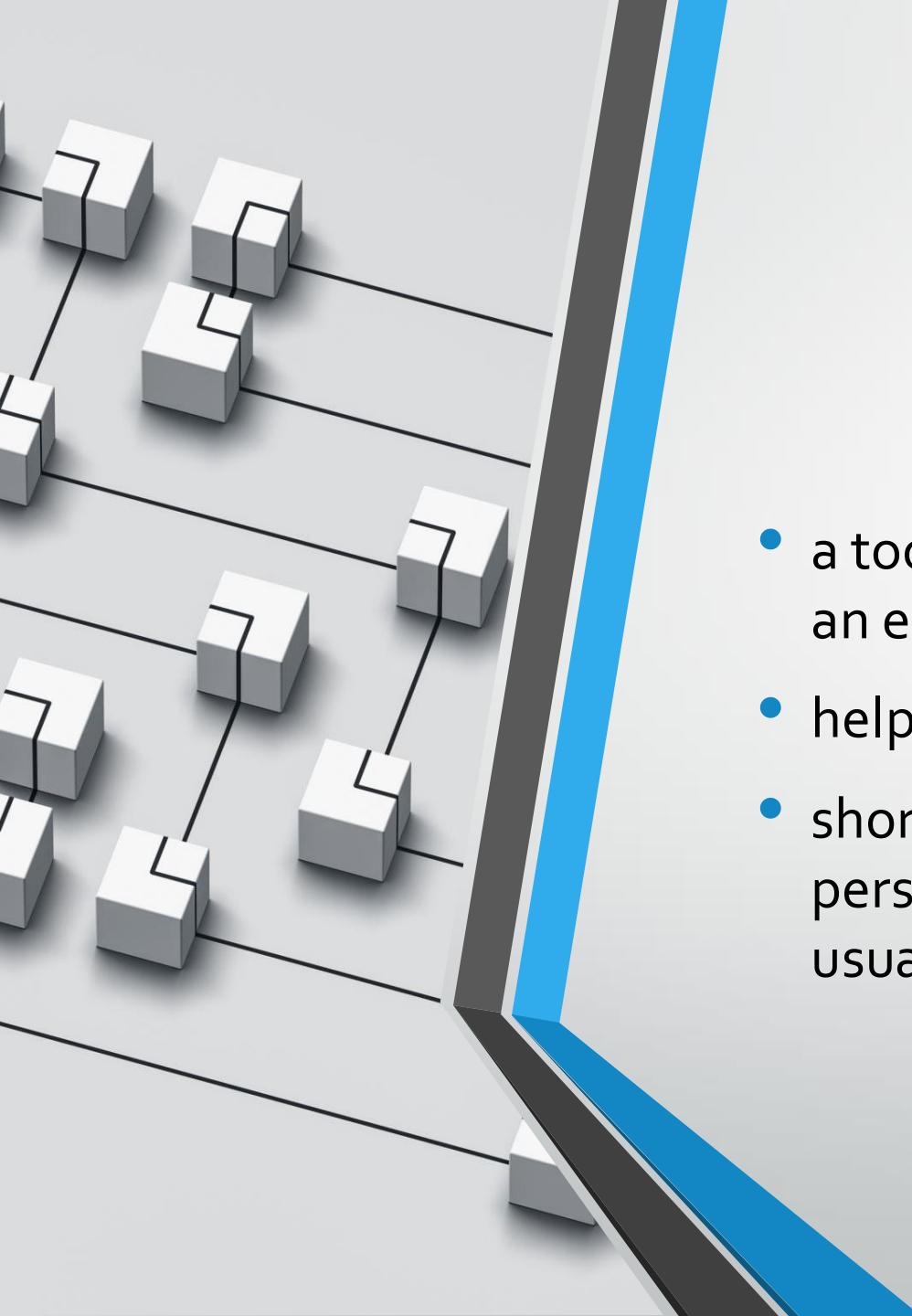
Logic tier

This layer coordinates the application, processes commands, makes logical decisions and evaluations, and performs calculations. It also moves and processes data between the two surrounding layers.

Data tier

Here information is stored and retrieved from a database or file system. The information is then passed back to the logic tier for processing, and then eventually back to the user.





What is a User Story?

- a tool to capture a description of a software feature from an end-user perspective
- helps to create a simplified description of a requirement
- short, simple descriptions of a feature told from the perspective of the person who desires the new capability, usually a user or customer of the system

User Stories Format & Examples

Format:

As an [actor / role] I want to [action / do something]
so that [achievement / business value]

Examples:

- As a *Flickr member*, I want to *set different privacy levels on my photos*, so I can *control who sees which of my photos*.



How to add detail to User Stories

1. By splitting a user story into multiple, smaller user stories.

Example: As a user, I can backup my entire hard drive.

- *As a user, I can specify files or folders to backup based on file size, date created and date modified.*
- *As a user, I can indicate folders not to backup so that my backup drive isn't filled up with things I don't need saved.*

How to add detail to User Stories

2. By adding "*acceptance criteria*"

- **Acceptance criteria** (AC) are the conditions that a software product must meet to be accepted by a user, a customer, or other system. They are unique for each user story and define the feature behavior from the end-user's perspective

Example:

"As a conference attendee, I want to be able to register online, so I can quickly get this task done, cut down on paperwork, and save trees."

Acceptance criteria:

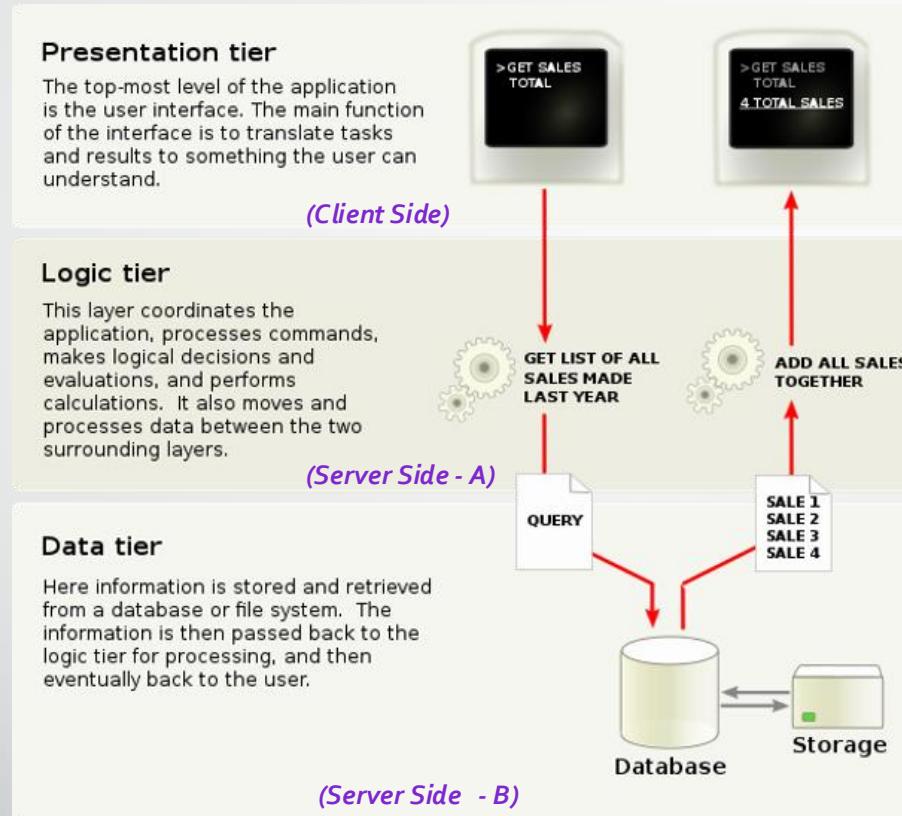
- A user cannot submit a form without completing all the mandatory fields.
- Form must be responsive (e.g., accessible via Mobile phone/tablets, should conform to screen size)
- Form information is stored in the registrations database (table XXXX)
- Protection against spam is active
- Users can pay registration fee by credit card
- An acknowledgment email is sent to the user after submitting the form
- Acknowledgement email should be received within 2-4 seconds after registration form is submitted

N-Tier Architecture

Example:
HTML5/JavaScript/CSS

Example:
C#, Java, Python, etc..

Example:
Oracle DB, MS DB, etc..



Wireframes: User Interface

Workflows, Use-Cases

Entity Relationship Diagrams
and Class diagrams

User Stories

Developers should breakdown User Stories into 3 main components

Example:

- As a User, I want to create an account
 - User Interface (**wireframe**): What should UI look like? What should we name the fields?
 - Logic (**workflow**): Check if the user exists before adding?
 - Data (**entity relationship diagram/class diagram**): What specific data fields (type, size/length) do we need to add to a table?

In Class Activity #3:

- The developer teams will orally present their previously conceptualized software ideas (which are considered challenging to create with current technology) to a customer/stakeholder team. The customer team will evaluate these ideas based on specific criteria without having access to any written documents.
- **Developer Team's Role:**
 1. **Oral Presentation of the Software Concept:**
 - Each developer team will use the software idea and specifications they created previously.
 - They must orally present their software concept, including a high-level description and specifications, without sharing any written documents.
 2. **Engage in Q&A:**
 - After the presentation, the developer team should be prepared to answer questions from the customer team, clarifying and expanding on their oral presentation as needed.
- **Customer/Stakeholder Team's Role:**
 1. **Evaluate the Presentation:**
 - Listen to the developer team's oral presentation.
 - Evaluate the presentation based on creativity, feasibility, understanding of user needs, and overall clarity, using the provided rubric.
 2. **Provide Oral Feedback:**
 - Offer constructive feedback to the developer team based on their evaluation. This feedback should focus on the strengths and areas for improvement in the concept and its presentation.
 3. **Submit to Canvas:**
 - Each customer/stakeholder team is required to prepare a written summary of their evaluation and feedback for submission. The write-up should encompass the following elements:
 - a. **Title/Name of the Software:** Include the title or name of the software concept as presented by the developer team.
 - b. **Paragraph Description:** Provide a brief description of the software idea based on your understanding from the oral presentation.
 - c. **List of 10 Features:** Note down 10 key features or specifications of the software as presented. These should be inferred from the oral presentation.
 - d. **Evaluation Score:** Assign an evaluation score based on the rubric provided. This score should reflect your assessment of the presentation in terms of creativity, user need understanding, clarity, practicality, feasibility, communication skills, and response to questions.

In-Class Activity #4

Reflecting on the 10 software specifications you compiled in Activity #2, please proceed with the following tasks:

1. Develop five user stories, each accompanied by its respective acceptance criteria.
2. Arrange a meeting with the client you interacted with in Activity #3. During this meeting, present your user stories and their associated acceptance criteria. Take note of any feedback or comments provided by the client.
3. Compile the client's feedback along with the user stories and acceptance criteria into a document, and submit this document to Canvas.



BACS385: 7th Day of Class

- Agenda -

- Review User Stories
- Breakout into your Teams: In Class Individual Activity #1

Client Request

Disney: Client requested that their IVR (interactive voice response) system greet each caller with the message "**Welcome [First Name of Caller]!**". However, if the caller-id (i.e., phone-number) is not recognized, then IVR must instead say "**Hello! I see this is your first time calling**". Additionally, if the caller has reserved tickets to a Walt Disney Resort, IVR must acknowledge this by responding with "**We can't wait to see you at [Name of Walt Disney Resort] on [Date of reservation]**" after the initial welcome greeting.



Client Request:

Disney: Client requested that their IVR (interactive voice response) system greet each caller with the message "Welcome [First Name of Caller]!". However, if the caller-id (i.e., phone-number) is not recognized, then IVR must instead say "*Hello! I see this is your first time calling*". Additionally, if the caller has reserved tickets to a Walt Disney Resort, IVR must acknowledge this by responding with "*We can't wait to see you at [Name of Walt Disney Resort] on [Date of reservation]*" after the initial welcome greeting.

How to attack this:

- **Must understand everything in the request.**
 - What is an IVR system? If you don't know, then Google is your friend.
 - Did we (i.e., your employer) develop this IVR? If so, who did? As a developer can I talk to them? Do we have existing documentation that I can review?
- **Extract type of users/roles and related data for each user/role**
- **Break down the existing request into separate parts**
- **Extract all parameters/conditions related to each part**

Client Request:

Disney: Client requested that their **IVR** (interactive voice response) system greet each **caller** with the message "Welcome [First Name of **Caller**]!". However, if the caller-id (i.e., phone-number) is not recognized, then **IVR** must instead say "*Hello! I see this is your first time calling*". Additionally, if the **caller** has reserved tickets to a Walt Disney Resort, **IVR** must acknowledge this by responding with "*We can't wait to see you at [Name of Walt Disney Resort] on [Date of reservation]*" after the initial welcome greeting.

- Extract type of users/roles and related data for each user/role
 - **IVR: Virtual Agent - AI**
 - Has access to caller information via caller-id
 - First Name
 - Has access before engaging the caller in conversation
 - Has knowledge of caller existing reservations
 - Date, Name of resort
 - IVR accepts NEW callers
 - IVR accepts REPEAT callers
 - **Caller: Human, customer**
 - Can call the IVR system
 - Can call multiple times
 - As a new/first-time caller
 - As a repeat caller
 - May have existing reservations in the system

Client Request:

Disney: Client requested that their IVR (interactive voice response) system greet each **caller** with the message "Welcome [First Name of **Caller**]!". However, if the caller-id (i.e., phone-number) is not recognized, then IVR must instead say "*Hello! I see this is your first time calling*". Additionally, if the **caller** has reserved tickets to a Walt Disney Resort, IVR must acknowledge this by responding with "*We can't wait to see you at [Name of Walt Disney Resort] on [Date of reservation]*" **after the initial welcome greeting**.

Break down the existing request into separate parts:

1. Greet each **caller** with the message "Welcome [First Name of **Caller**]!".
2. If caller-id (i.e., phone-number) is not recognized, then **IVR** must instead say "*Hello! I see this is your first time calling*".
3. If caller has reserved tickets to a Walt Disney Resort, **IVR** must acknowledge this by responding with "*We can't wait to see you at [Name of Walt Disney Resort] on [Date of reservation]*" after the initial welcome greeting.

Client Request:

Disney: Client requested that their IVR (interactive voice response) system greet each **caller** with the message "Welcome [First Name of **Caller**]!". However, if the caller-id (i.e., phone-number) is not recognized, then IVR must instead say "*Hello! I see this is your first time calling*". Additionally, if the **caller** has reserved tickets to a Walt Disney Resort, IVR must acknowledge this by responding with "We can't wait to see you at [Name of Walt Disney Resort] on [Date of reservation]" **after the initial welcome greeting**.

Extract all parameters/conditions related to each part

1. Greet each **caller** with the message "Welcome [First Name of Caller]!".
 - Must check if caller exists using caller-id
 - If caller exists, then get first-name of caller
 - Greet caller with the phrase: "Welcome [First Name of Caller]!".
2. If caller-id (i.e., phone-number) is not recognized, then **IVR** must instead say "*Hello! I see this is your first time calling*".
 - Must check if caller exist using caller-id
 - If caller is new, then IVR must greet caller with phrase: "*Hello! I see this is your first time calling*"
3. If caller has reserved tickets to a Walt Disney Resort, **IVR** must acknowledge this by responding with "We can't wait to see you at [Name of Walt Disney Resort] on [Date of reservation]" after the initial welcome greeting.
 - Must check if caller exists using caller-id
 - If caller exists, then get caller's existing reservations (a list containing Date and Name of Resort)
 - If caller has existing reservations, then
 - **Question to client: What happens if caller has multiple reservations?**
 - **Do we sort by most upcoming reservation date?**
 - Greet caller with the phrase: "We can't wait to see you at [Name of Walt Disney Resort] on [Date of reservation]"

User Stories

first draft

1. As an existing caller, I must be greeted with the message "**Welcome [First Name of Caller]!**".
 - **Acceptance criteria:**
 - Must check if caller exists using caller-id
 - If caller exists, then get first-name of caller
 - IVR must greet caller with the phrase: "**Welcome [First Name of Caller]!**"
 - Must call as a new caller and should **not** receive this message
2. As a new caller, I must be greeted with the message "**Hello! I see this is your first time calling**".
 - **Acceptance criteria:**
 - Must check if caller exist using caller-id
 - If caller is new, then IVR must greet caller with phrase: "**Hello! I see this is your first time calling**"
 - Must call as an existing caller and should **not** receive this message
3. As a caller with reservations, I must be greeted with the message "**We can't wait to see you at [Name of Walt Disney Resort] on [Date of reservation]**" after the initial welcome greeting.
 - **Acceptance criteria:**
 - Must check if caller exists using caller-id
 - If caller exists, then get caller's existing reservations (a list containing Date and Name of Resort)
 - If caller has existing reservations, then
 - IVR must greet caller with the phrase: "**We can't wait to see you at [Name of Walt Disney Resort] on [Date of reservation]**"
 - Must call as a new caller and existing caller without reservations and should **not** receive this message

ChatGPT's User Stories



You

Please read through the following list of client request, where each client operates on a different system for a different company. As a software project lead, you are initially required to independently create 3 user stories with acceptance criteria for each client, so that your developers can effectively work on them later.

Disney: Client requested that their IVR (interactive voice response) system greet each caller with the message "Welcome [First Name of Caller]!". However, if the caller-id (i.e., phone-number) is not recognized, then IVR must instead say "Hello! I see this is your first time calling". Additionally, if the caller has reserved tickets to a Walt Disney Resort, IVR must acknowledge this by responding with "We can't wait to see you at [Name of Walt Disney Resort] on [Date of reservation]" after the initial welcome greeting.



ChatGPT

Disney Client User Stories

User Story 1: Personalized Greeting Based on Caller-ID

- **As a** Disney IVR system,
- **I want to** greet callers with their first name if their phone number is recognized,
- **So that** the caller feels a personalized connection to the service.

Acceptance Criteria:

1. The system should cross-reference the incoming caller-ID with the database of known contacts.
2. If the caller-ID is recognized, the system should extract the caller's first name from the database.
3. The IVR system should then greet the caller with "Welcome [First Name of Caller]!".

User Story 2: Greeting for Unrecognized Caller-IDs

- **As a** Disney IVR system,
- **I want to** greet first-time callers with a specific message,
- **So that** new callers feel acknowledged and informed.

Acceptance Criteria:

1. The system should check if the incoming caller-ID is not present in the database.
2. If the caller-ID is not recognized, the IVR system should greet the caller with "Hello! I see this is your first time calling".



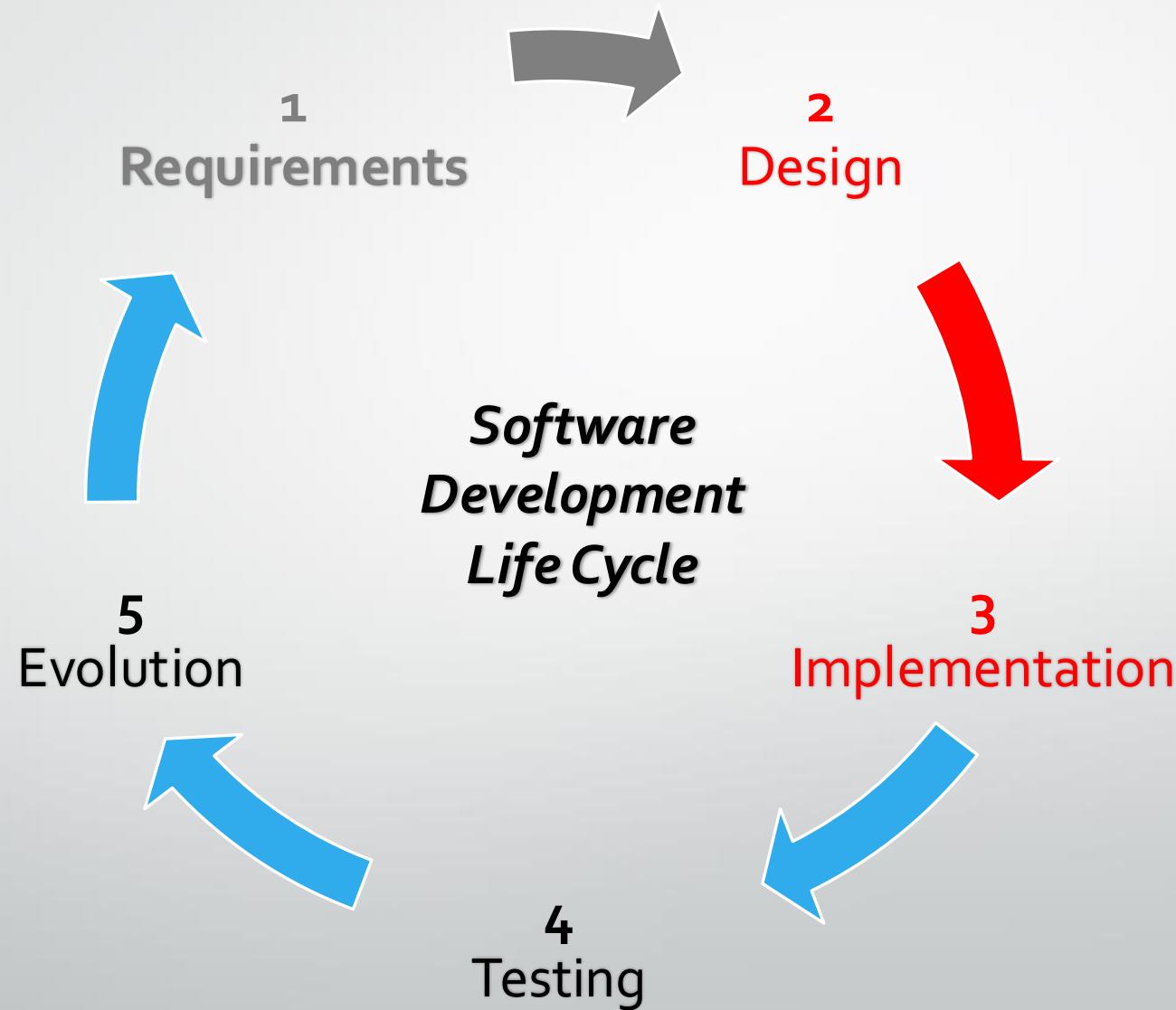
In-Class Individual Activity #1

- Please see Canvas.

BACS385: 7th Day of Class - Agenda -

- Software Design
 - Wireframes
- Breakout into your Teams: In Class Activity #5

Reminder:



Software Design

- Design: according to dictionary:
 - a plan or drawing produced to show the look and function or workings of a building, garment, or other object before it is built or made.
 - decide upon the look and functioning of (a building, garment, or other object), typically by making a detailed drawing of it.
- Software design is a creative activity in which you identify software components and their relationships, based on a customer's requirements.
 - Provide a technical description (blueprints) about how the system will implement the requirements

Software Design: Artifacts

Linking to Requirements Engineering (RE):

- Recall that in the RE process, we developed User Stories with acceptance criteria.
- In Software Design, these are transformed into detailed design documents.

Key Artifacts in Software Design:

- **User-Interface: Wireframes**

- Visual layouts translating user stories into graphical interfaces.

- **Logic Representation: Workflows**

- Diagrams showing the flow of operations and processes.

- **Data Representation: Entity Relationship Diagrams (ERD)**

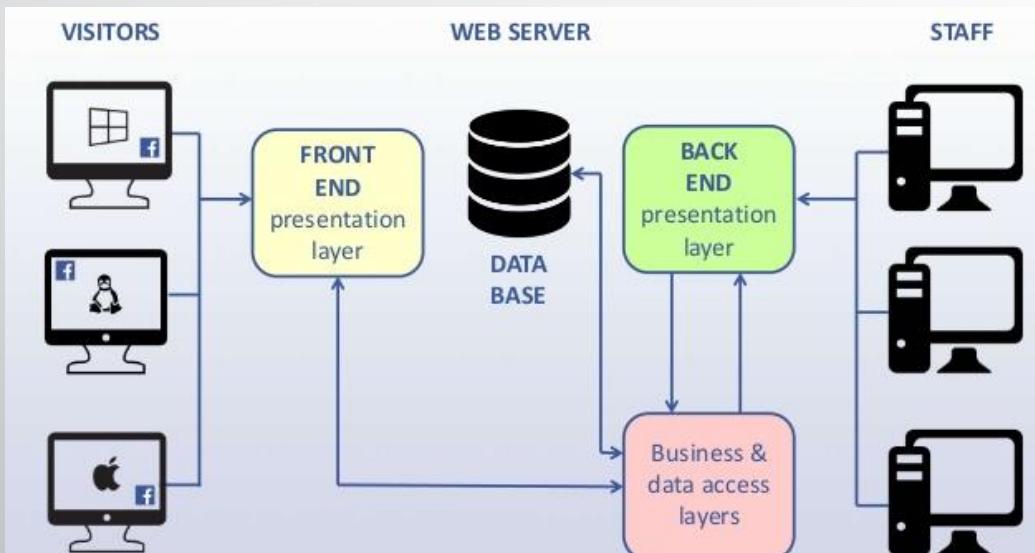
- Maps out data structures and their relationships.

From Concept to Blueprint:

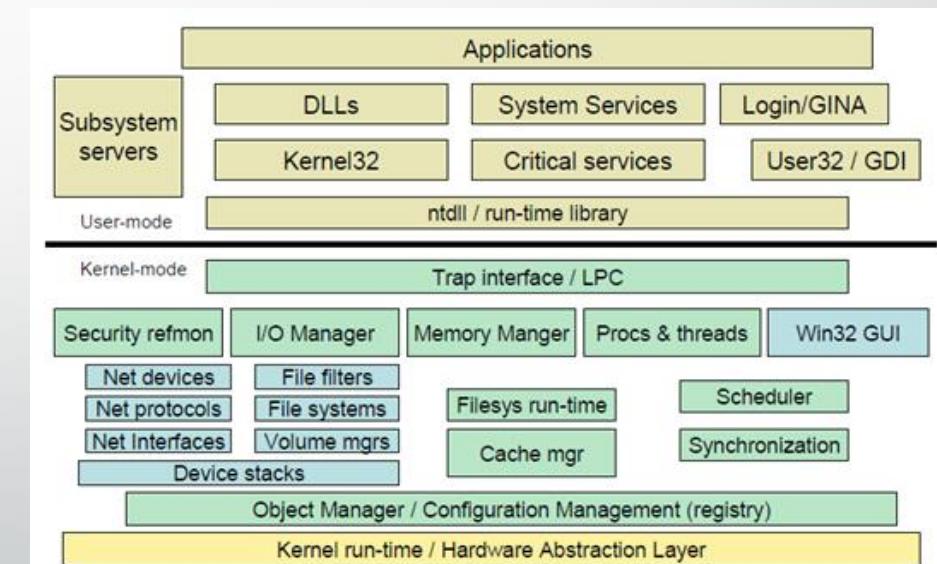
- These artifacts turn the conceptual requirements into detailed, visual blueprints for building the software.

Design Activities

- **Architectural design**, where one identifies the **overall structure** of the system, the **principal components** (sometimes called sub-systems or modules or layers, etc...), their **relationships** and how they are distributed.
 - Provides a view of the system at an abstract level. It shows how the major pieces of the finished application will fit together and interact with each other.



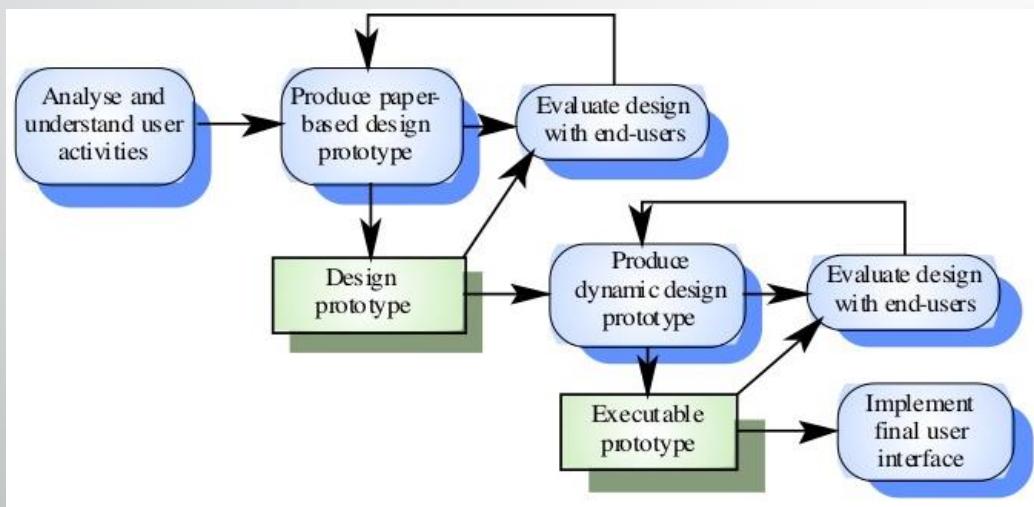
Facebook



Microsoft Windows

Design Activities

- **Interface design**, where one defines the interfaces between system components. (interface to controller; controller to database; external device (sensor) to analyzer...)
 - The **user interface (UI)**, in the industrial design field of human–computer interaction, is the space where interactions between humans and machines occur. The goal of this interaction is to allow effective operation and control of the machine from the human end, whilst the machine simultaneously feeds back information that aids the operators' decision-making process.
 - Recognize that to the end user, the UI **is** the application.



User Interface Design Process

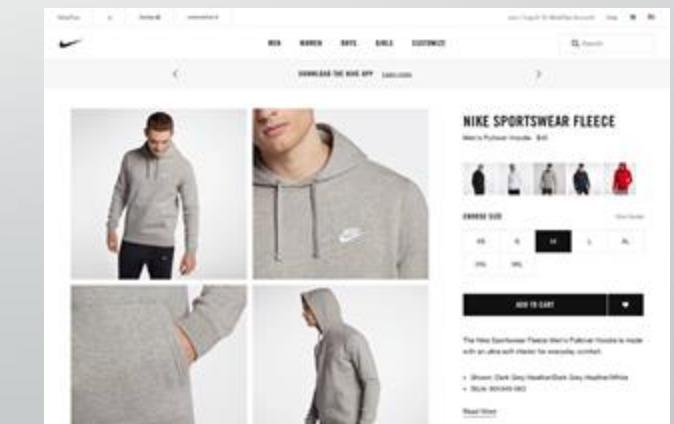
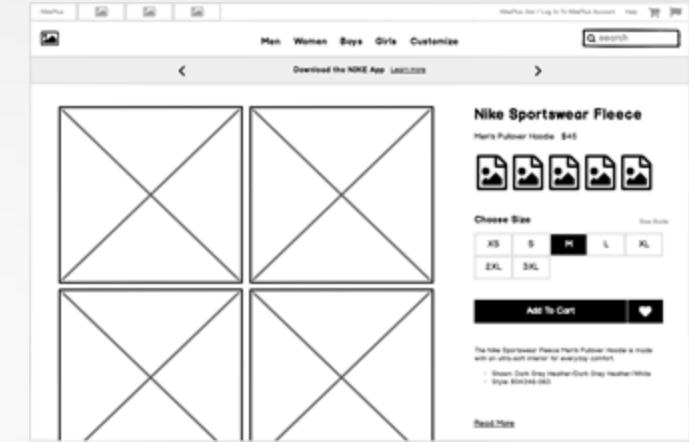
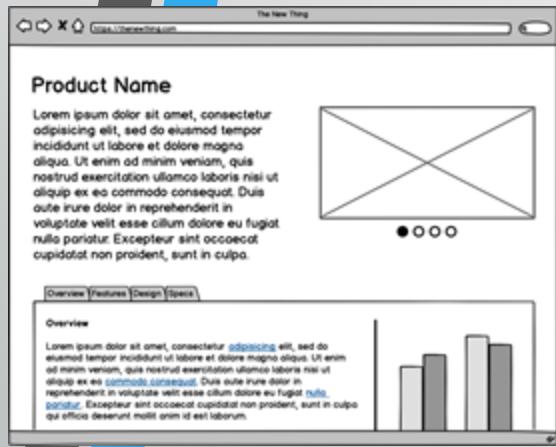
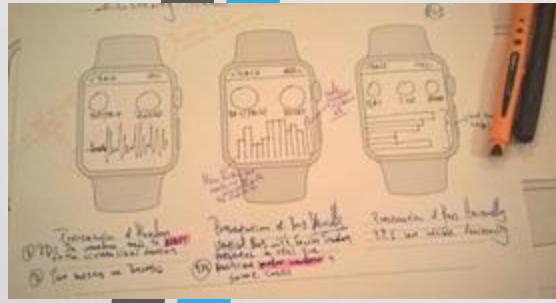


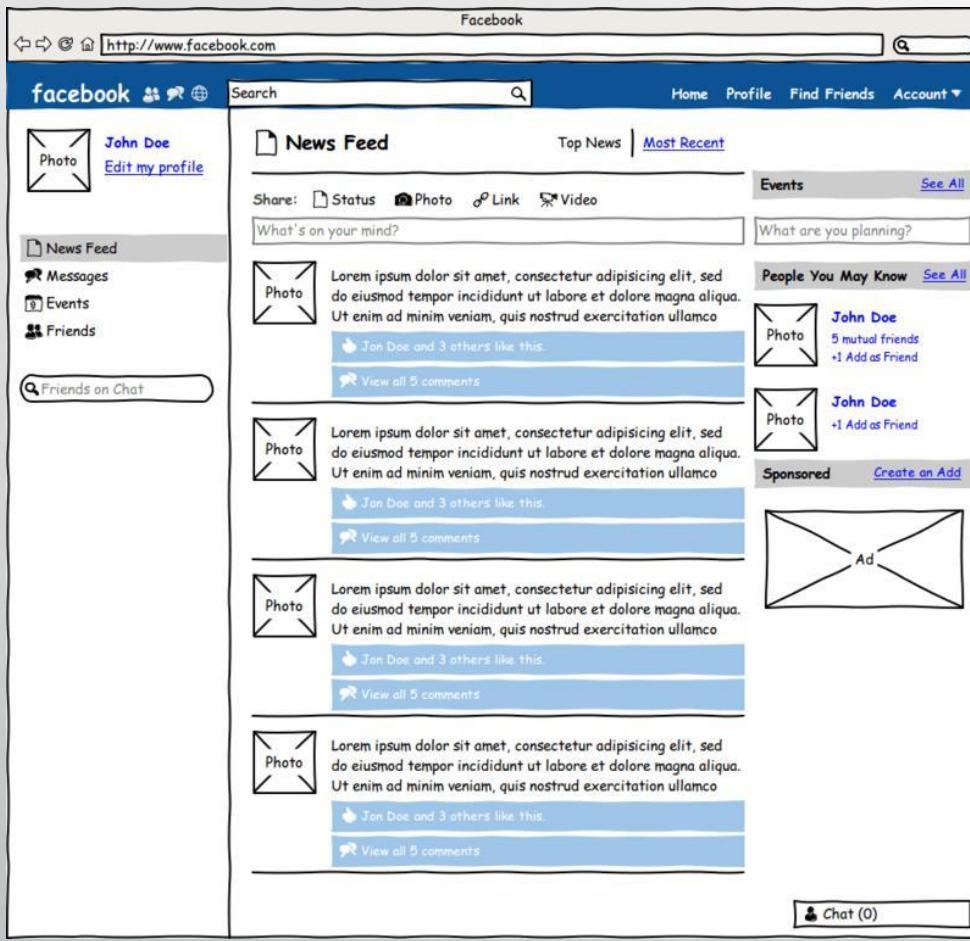
*American Airlines
Kiosk User Interface*

Design Activities: *Interface design*

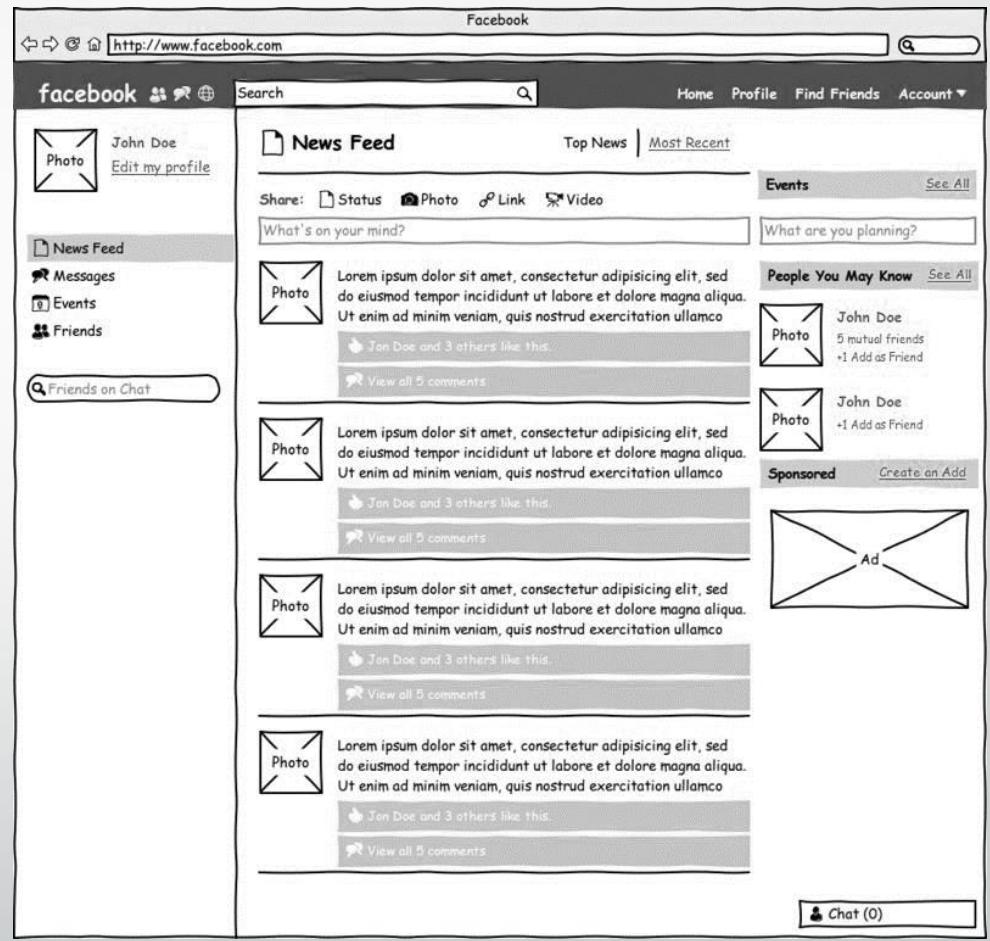
Wireframes

- Wireframes are sketchy, throw-away prototype of a Web page or application screen, a wireframe is used to identify the elements that will be displayed on the page or screen, such as
 - Navigation
 - Content sections
 - Imagery and/or media needs
 - Form elements
- typically created in black and white or shades of gray, use placeholders for images, and do not get into specifics of fonts.
- used to represent the site or application to client, designers, developer and other team members who have a stake in it at its very core level

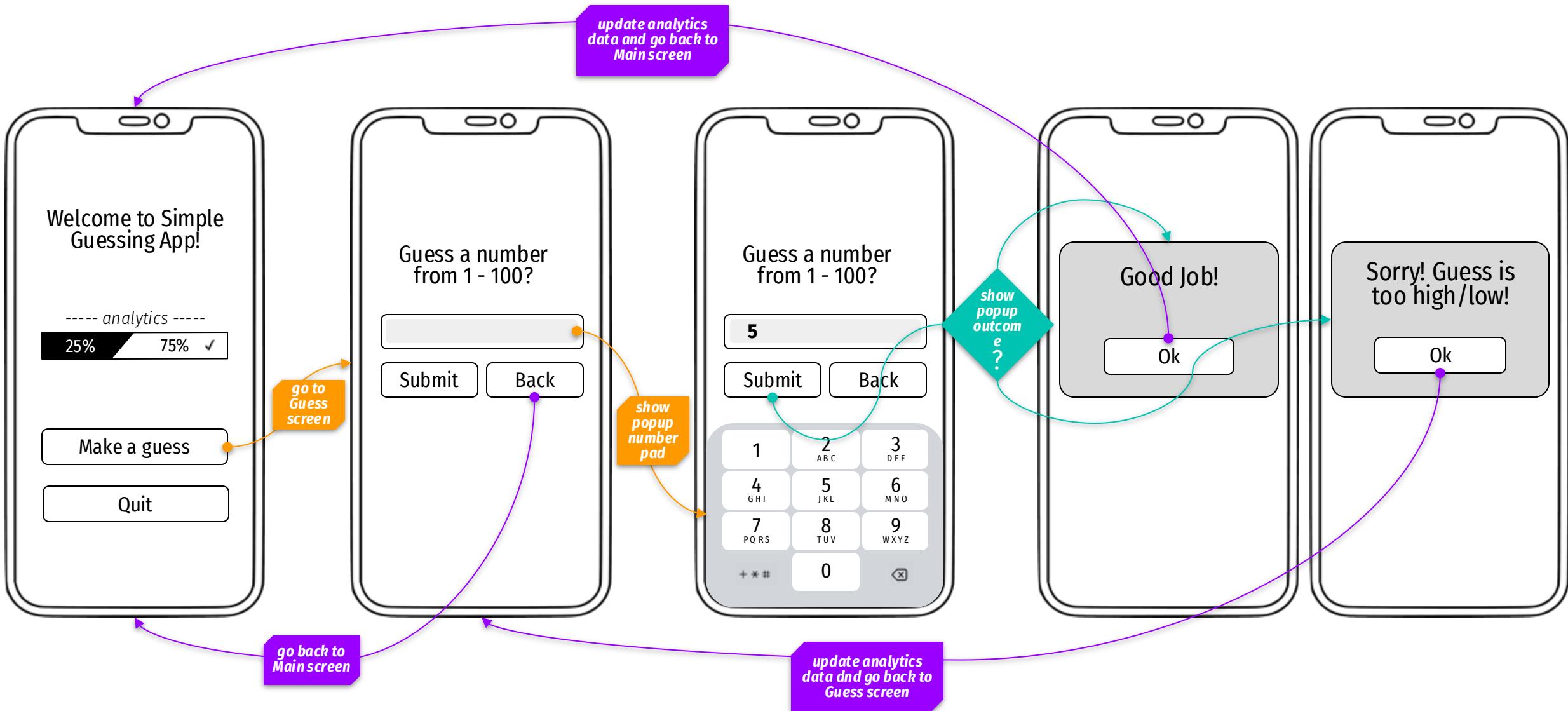




VS.



Guessing Game App Wireframe

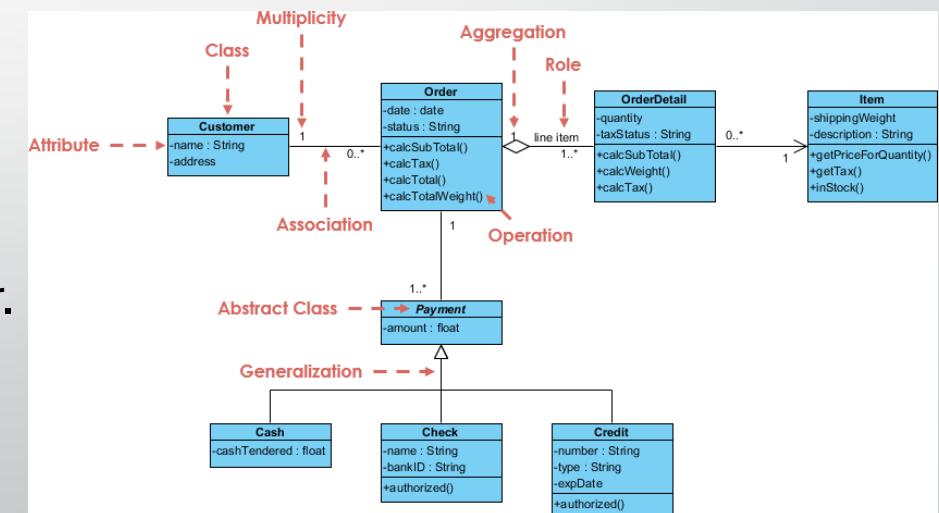


Design Activities

- **Component design**, where one takes each system component and design how it will operate.
- **Database design**, where one designs the system data structures and how these are to be represented in a database.

Notes:

- Fills in some of the gaps to provide extra detail that's necessary before developers can start writing code.
- It gives more specific guidance for how the parts of the system will work and how they will work together.
- It refines the definitions of the database, the major classes, and the internal and external interfaces.





In-Class Activity #5

- 1.** Select 2 user stories that have an acceptance criteria from in-class activity #4.
- 2.** Create wireframe(s) depicting each user-story:
 - You must use Figma for this task. If you haven't already, create a free account at figma.com.
 - A single user story may have one or many wireframes depicting the steps that users must take to demonstrate its functionality.
- 3.** Meet with your client and present your wireframe:
 - Record their comments and feedback during this presentation.
- 4.** Submission to Canvas:
 - Submit a document that includes:
 - Each user story and its corresponding acceptance criteria.
 - The wireframe(s) created for each user story (links to Figma projects or screenshots).
 - A summary of the feedback received from your client.
 - Make sure to associate each user story with its corresponding wireframe as a point of reference.

BACS385: 9th Day of Class - Agenda -

- Guess the User-Story
- Breakout into your Teams: In Class Activity #6

Guess the User-Story

The image shows a globe with various location markers. A black dot represents the current location at TeleHub 17 in Los Angeles. A line connects this dot to a selected destination marked with a checkmark. Preferred locations are marked with stars. Other offered destinations are indicated by open circles. The globe is set against a background of a world map.

Current Location:

TeleHub 17
Los Angeles, California, 90005

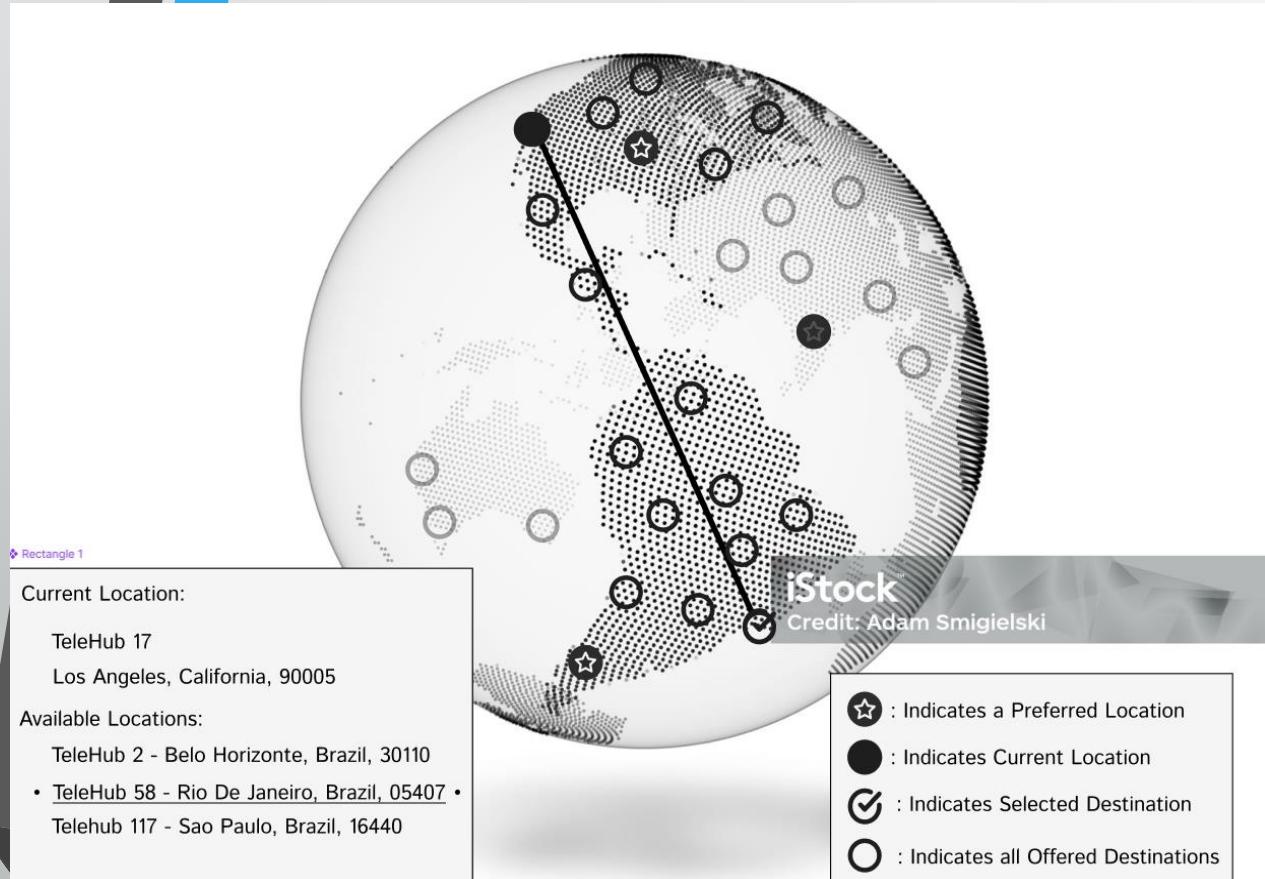
Available Locations:

- TeleHub 2 - Belo Horizonte, Brazil, 30110
- TeleHub 58 - Rio De Janeiro, Brazil, 05407
- Telehub 117 - Sao Paulo, Brazil, 16440

iStock
Credit: Adam Smigelski

- ★ : Indicates a Preferred Location
- : Indicates Current Location
- ✓ : Indicates Selected Destination
- : Indicates all Offered Destinations

Guess the User-Story



As a passenger, I would like to teleport from the USA to Paris. There should be a mapping system to tell what hub I need to go and to what hub will be teleported to the destination if there are many options, so that I don't have to travel for a long distance to a hub while I have one close to me.

Acceptance Criteria:

- Mapping system showing the nearby hub and all available hub at destination
- Passengers being able to choose by preference

Guess the User-Story

The screenshot shows a user interface for a dream analysis app. At the top, there is a blue cloud icon containing the text "DreamScape" and three small "z" symbols indicating sleep. Below this, a message box contains the following text:

Last night's dream has been recorded!

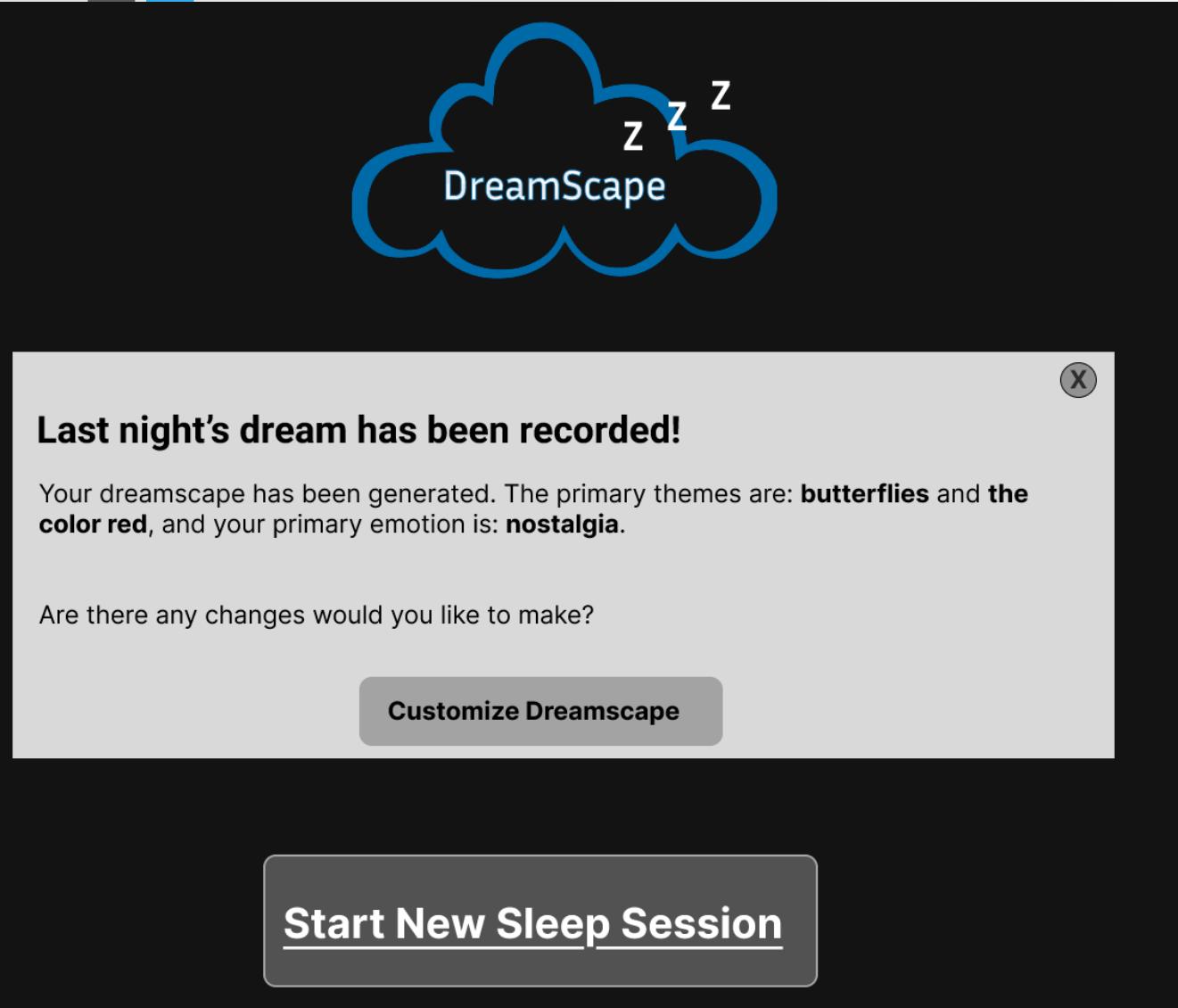
Your dreamscape has been generated. The primary themes are: **butterflies** and the **color red**, and your primary emotion is: **nostalgia**.

Are there any changes would you like to make?

[Customize Dreamscape](#)

At the bottom, there is a large button labeled [Start New Sleep Session](#).

Guess the User-Story

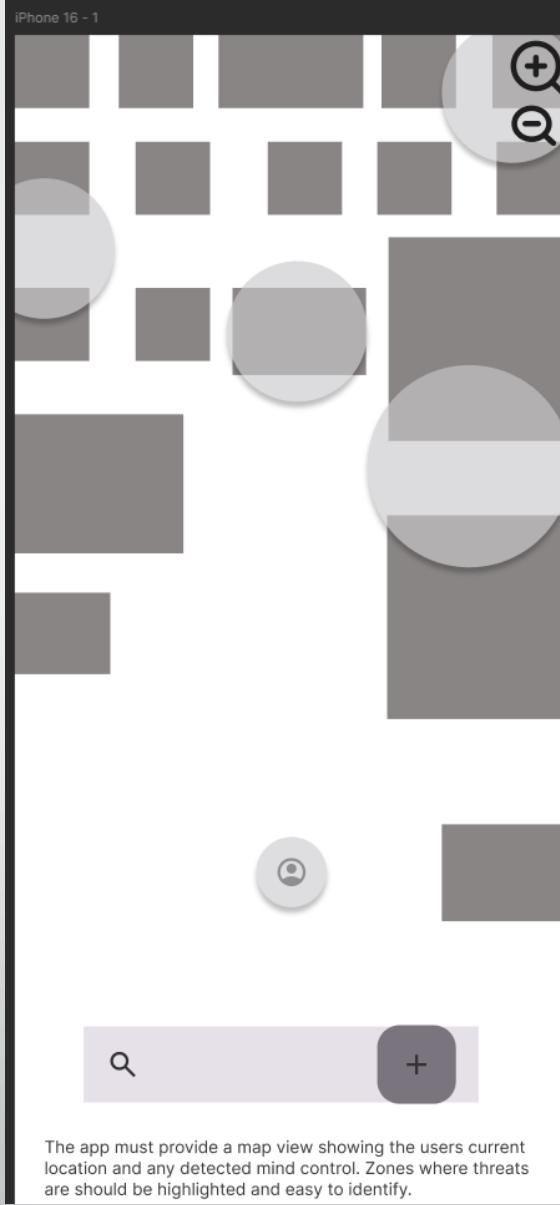


As a user, I want Dreamscape to capture and analyze my brainwave patterns during sleep, so that my dreams can be accurately reconstructed.

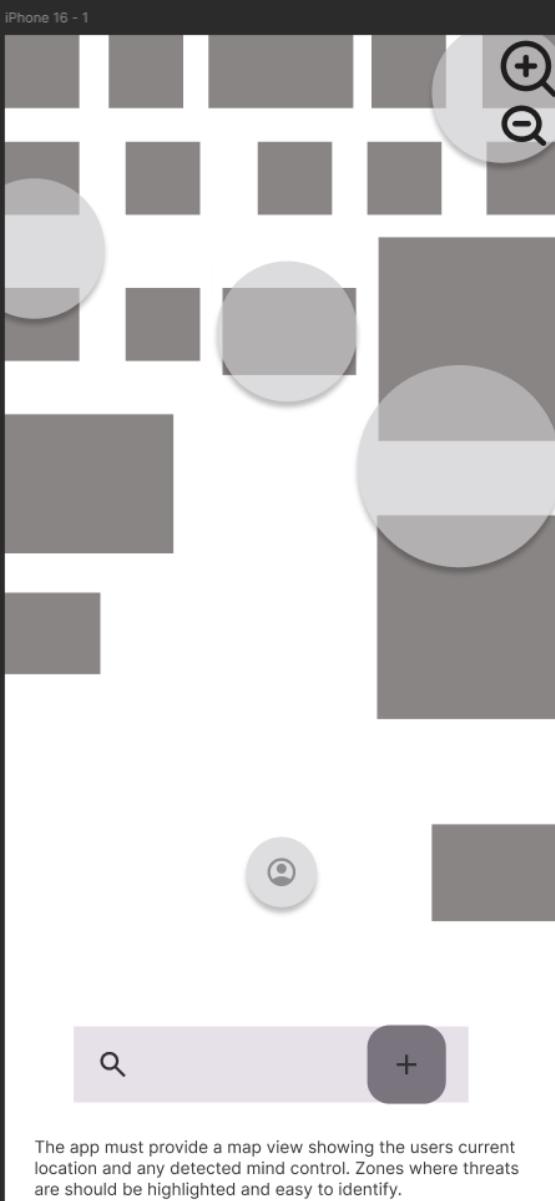
Acceptance Criteria:

- The application integrates with EEG devices to monitor and record brain activity.
- Captured data is processed with an accuracy of 95% or higher by the AI Dream Recognition Engine.
- Users receive a notification confirming data capture and initial analysis.

Guess the User-Story



Guess the User-Story



As a user, I want to be able to see a real-time map that displays areas where mind control is detected or where potential threats are in my vicinity.

Acceptance criteria - The app must provide a map view showing the users current location and any detected mind control. Zones where threats are should be highlighted and easy to identify.

Guess the User-Story

Auto Detection of Language

The screenshot shows a mobile application interface. At the top, there is a navigation bar with three horizontal lines on the left and a search bar containing the text "Search For Langauge" with a magnifying glass icon on the right. Below the navigation bar is a section titled "Favorite Langauges" enclosed in a purple-bordered box. This section contains a descriptive text and a list of three languages: English, Spanish, and French, each with a star icon next to it. At the bottom of the screen is a dark blue button labeled "Auto Detect Langage" with a white arrow pointing to the right.

≡ Search For Langauge

Favorite Langauges

Stored here is your "favorite" Languages and the previous ones you used in the translator. To favorite a language, use the search bar and then hit the star next to the language you want to favorite.

E	English	★
S	Spanish	★
F	French	★

Auto Detect Langage →

Guess the User-Story

Auto Detection of Language

Search For Langauge

Favorite Langauges

Stored here is your "favorite" Languages and the previous ones you used in the translator. To favorite a language, use the search bar and then hit the star next to the language you want to favorite.

E	English	★
S	Spanish	★
F	French	★

Auto Detect Langage →

As a user of the application, I would like to be able to easily navigate the user interface allowing for auto detection of the language being spoken to me without having to manually input it

Acceptance Criteria:

- Users should be able to easily select the “auto detection” feature in user interface.
- Auto detection feature should be able to quickly identify what language is being communicated and begin translation almost immediately.
- Users should be informed what language is being spoken to them as well as a breakdown of every nuance.
- Application should automatically default to auto-detection unless input language is otherwise specified.

Guess the User-Story

Dream Recordings

Filter

Happy

Sad

Nightmares

Happy Dreams

Nightmares

Sad Dreams

Title

Title

Title

Title

Title

Title

Monday Feb 3
Nightmare

Tuesday Feb 4
Floating in space

Wednesday Feb 5
Video game idea

Play Video....

Guess the User-Story

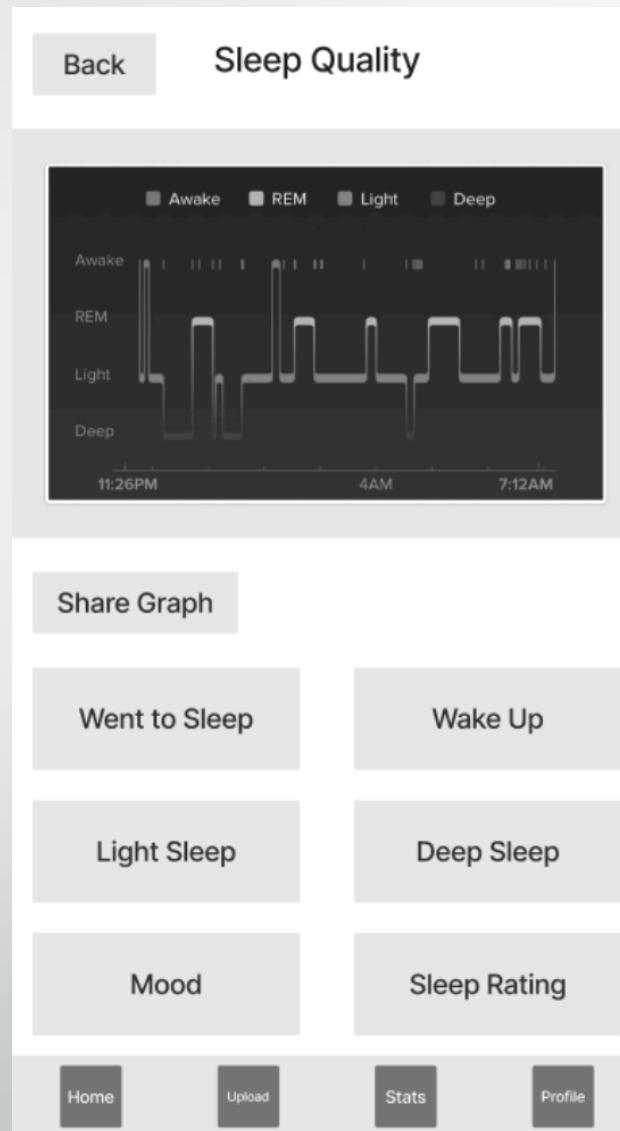


As a user I want to have a portal that has my dreams saved so that I can access and replay my dreams.

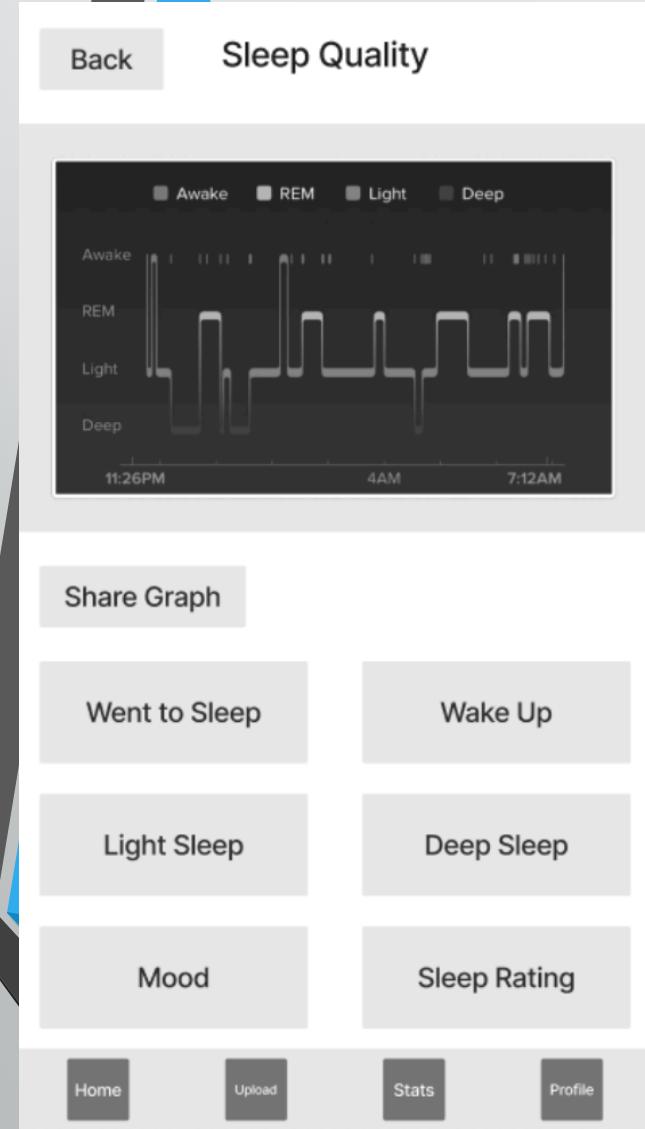
Acceptance Criteria:

- Must have a functional dream dashboard if clicked and show dream categories.
- Once clicked on a specific video, it must be accessible for replay.

Guess the User-Story



Guess the User-Story

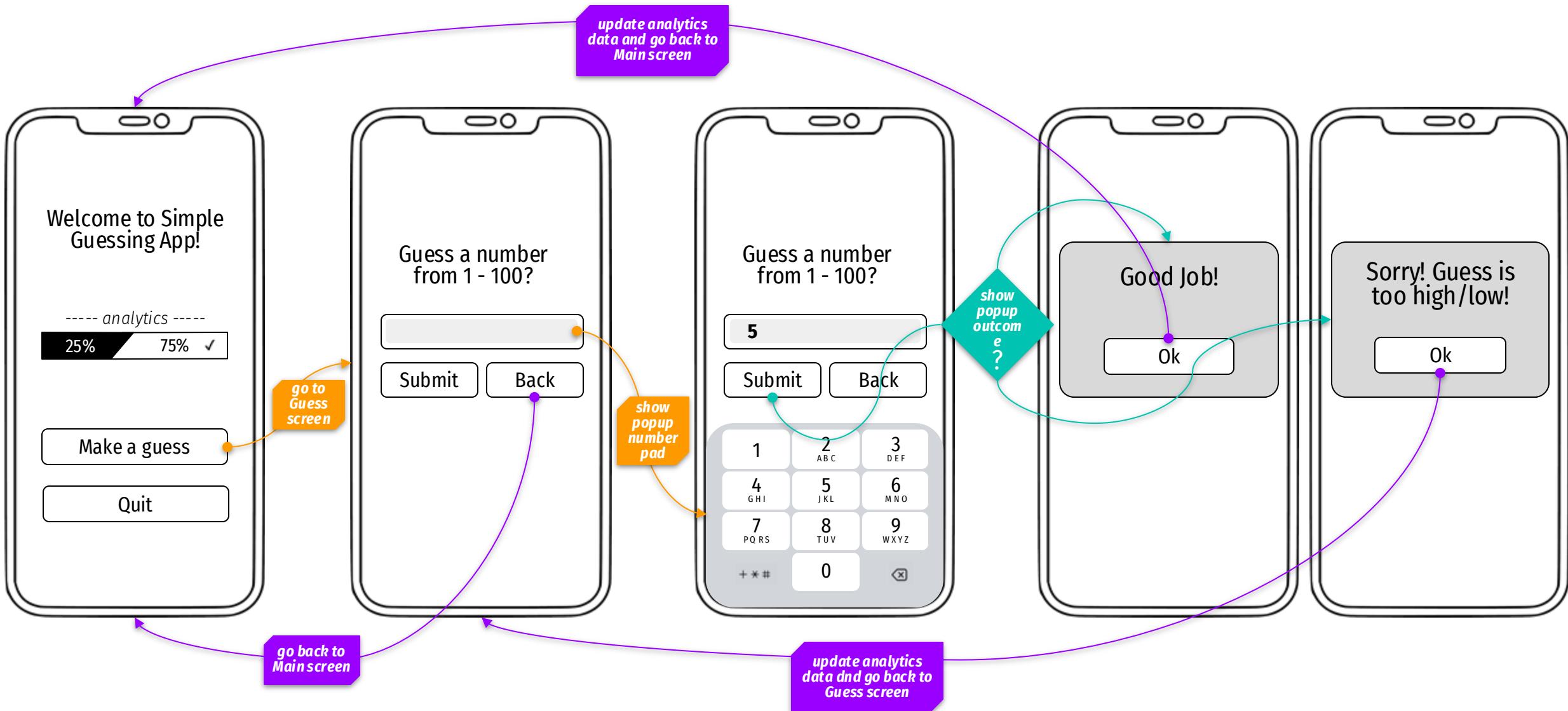


As a dream capture user, I want to be able to monitor my sleep, so that I can better understand my sleep quality.

Acceptance Criteria:

- i. System must track sleep patterns
- ii. Integration with wearable devices must be available for accurate data collection
- iii. Users can see detailed sleep reports via visual charts and trends
- iv. Provide user with personalized sleep improvement suggestions based off

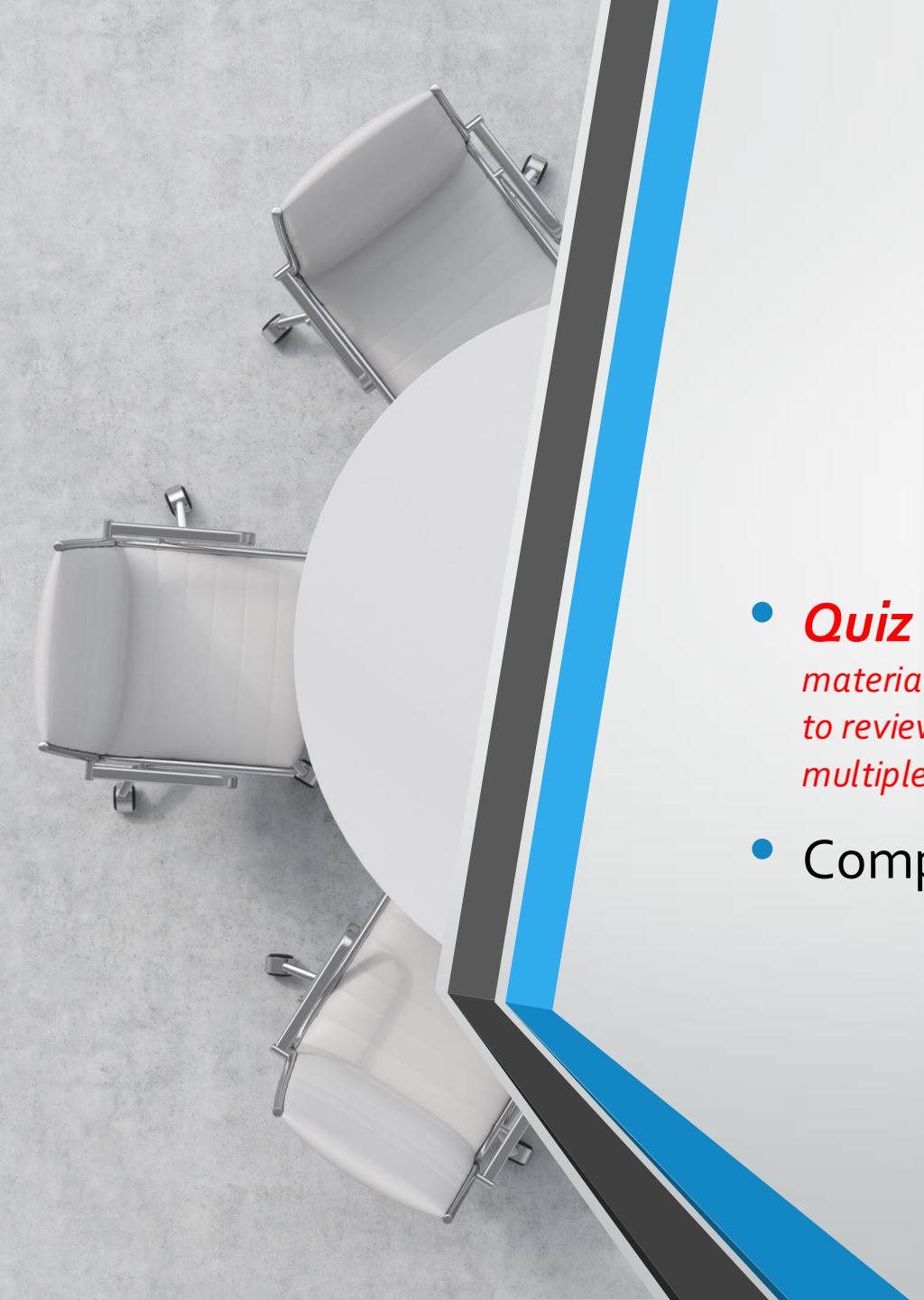
Guessing Game App Wireframe





Complete In-Class Activity #6

- Select 2 user stories (not selected in activity #5) and repeat activity 5 steps.
 - Must use uniform design and consistency
 - Think of a platform for your app (e.g., iPhone, desktop, etc..)
 - Must add annotations helping the user understand how to interact with UI



BACS385: 10th Day of Class

- Agenda -

- **Quiz 1:** *Scheduled for Friday, February 16, during class time, will encompass all material up to and including the sections on User Stories and Wireframes. Please ensure to review all related group and individual activities. Note that the exam will be in a multiple-choice format, administered on Canvas, and will be closed notes/book.*
- Complete Individual Activity #2

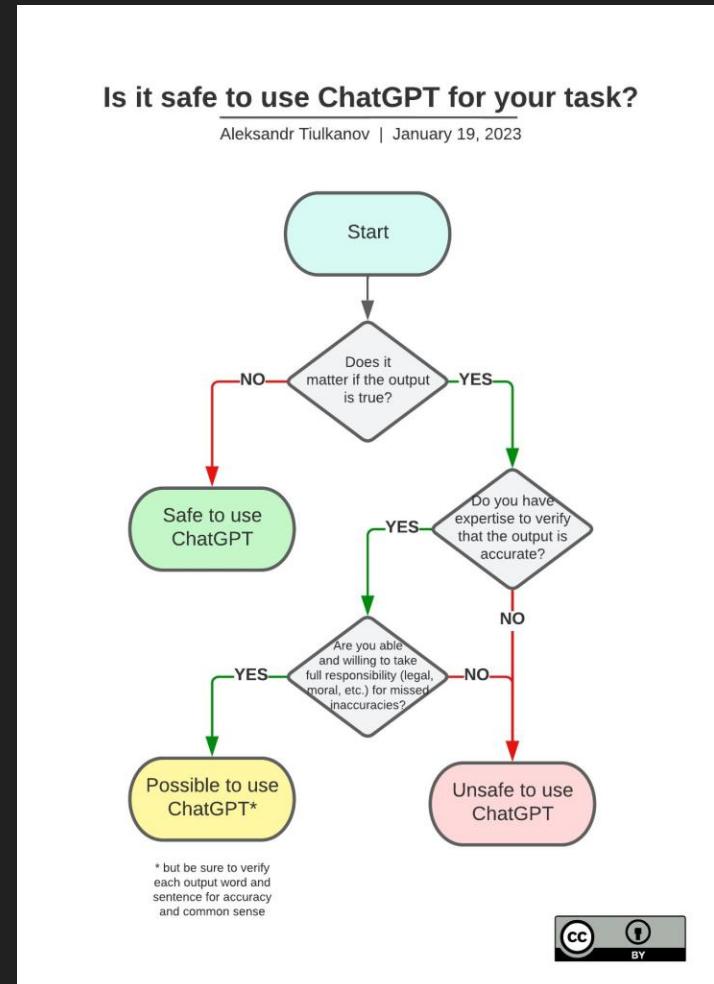


BACS385: 11th Day of Class

- Agenda -

- Software Design: Workflow Diagrams
- Breakout into your Teams: In Class Activity #7

What is a Workflow?



- **Definition:**

- A workflow is the repeatable pattern of activities or steps that take place to complete a task regularly.

- **Characteristics:**

- Involves tasks, documents, or information passed to a participant for action in a sequence defined by business rules.
- A predefined set of activities yielding a desired outcome.
- Can range from simple, sequential steps to complex, parallel series of events with dependencies and requirements.
- Consistency in results upon each execution.
- Comprises steps, necessary resources, and their interactions.

Workflows in Software Design

- **Recap:**

- User Stories define the functionalities and outcomes from the user's perspective, Wireframes sketch the user interface and interactions, and Acceptance Criteria specify the conditions under which a feature is considered complete and functioning correctly.
- User Stories guide the identification of essential steps in a workflow, highlighting what users aim to accomplish, ensuring all necessary functionalities are covered.
- Wireframes provide a visual draft that aids in plotting the flow of actions, detailing the sequence of user interactions and the interface elements involved.
- Acceptance Criteria add a layer of precision, outlining the specific conditions that must be met for each step or feature within software.

Question: *How can we use User Stories, Wireframes and Acceptance Criteria in Workflows?*

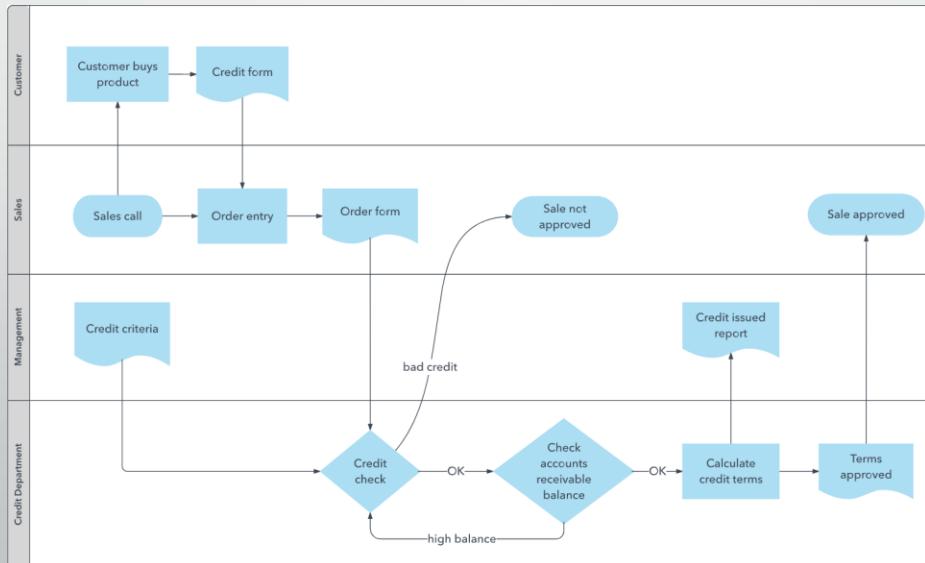
Why Create Workflows?

Workflow diagrams are a vital component of the software development lifecycle. They provide a visual representation of the sequence of operations or steps required to complete a software process, which is crucial for understanding, analyzing, and optimizing that process. By clearly outlining how different parts of the system interact and the flow of data and control between them, workflow diagrams help in:

- **Improving Communication:** They serve as a common language that can be understood by both technical and non-technical stakeholders, facilitating better communication and collaboration.
- **Identifying Bottlenecks and Inefficiencies:** By visually mapping out the process, it's easier to spot redundancies, unnecessary steps, or bottlenecks that could be streamlined.
- **Enhancing System Design:** They allow designers and developers to see the big picture and the details of system interactions, helping in making informed design decisions that align with user stories and acceptance criteria.
- **Supporting Testing and Debugging:** Workflow diagrams can guide the creation of test cases and help in identifying potential points of failure.
- **Facilitating Change Management:** They make it easier to understand the impact of proposed changes on the system's workflow and to plan accordingly.

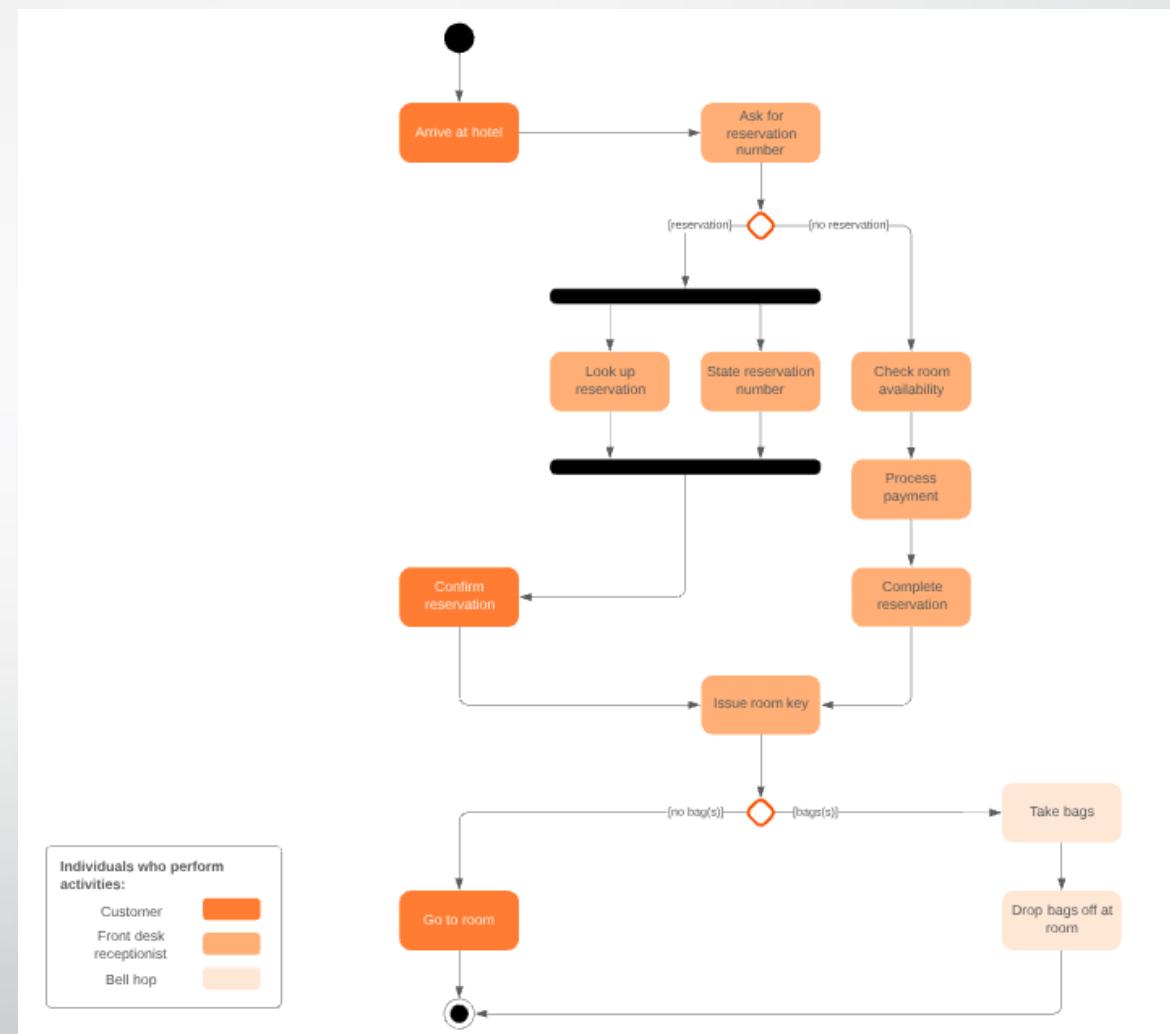
Types of Workflow Diagrams

- **Swimlane Diagrams:** Separate tasks or activities by department or team, highlighting interactions and potential inefficiencies.
- **Example Scenario:** Order Processing System
 - **Sales Department:** Receives order and checks customer details.
 - **Credit Department:** Verifies payment information and approves credit.
 - **Warehouse:** Manages inventory check and dispatches the order.
 - **Delivery Team:** Handles shipping and confirms delivery to the customer.
- **Benefit:** Clearly shows how each department interacts within the process, identifying bottlenecks or inefficiencies in cross-departmental workflows.



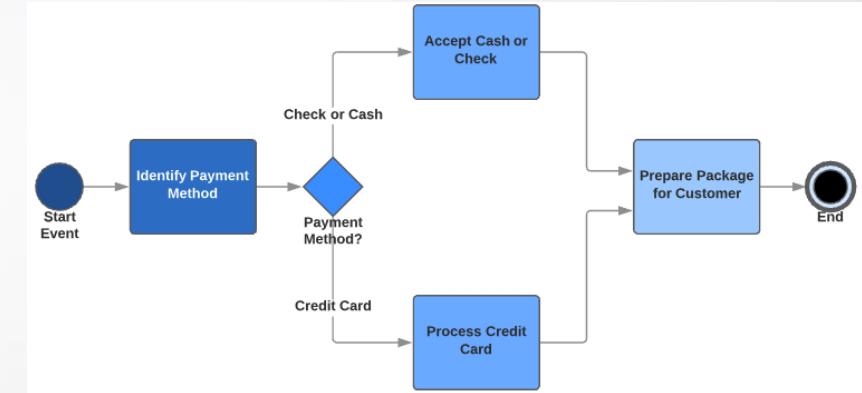
Types of Workflow Diagrams

- **UML Activity Diagrams:** Use Unified Modeling Language to represent the sequence of steps and the flow of control in complex systems.
- **Example Scenario:** User Login and Authentication
 - **Start:** User accesses the login screen.
 - **Action:** User enters username and password.
 - **Decision Point:** System verifies credentials.
 - **If valid, proceed to the user dashboard.**
 - **If invalid, return to login screen with an error message.**
 - **Benefit:** Demonstrates the flow of control through various decision points, offering a clear visualization of logical steps in software processes.



Types of Workflow Diagrams

- **BPMN (Business Process Modeling Notation):** Emphasizes business processes using a standardized flowchart method.
- **Example Scenario:** Customer Service Request Handling
 - **Start Event:** Customer submits a service request.
 - **Tasks:**
 - Registration of request.
 - Initial assessment by service team.
 - Assignment to the appropriate department.
 - Resolution of the request.
 - **End Event:** Customer receives resolution and feedback is collected.
 - **Benefit:** Facilitates a common language for describing the steps in business processes, making it easier for teams to design, document, and optimize workflows.



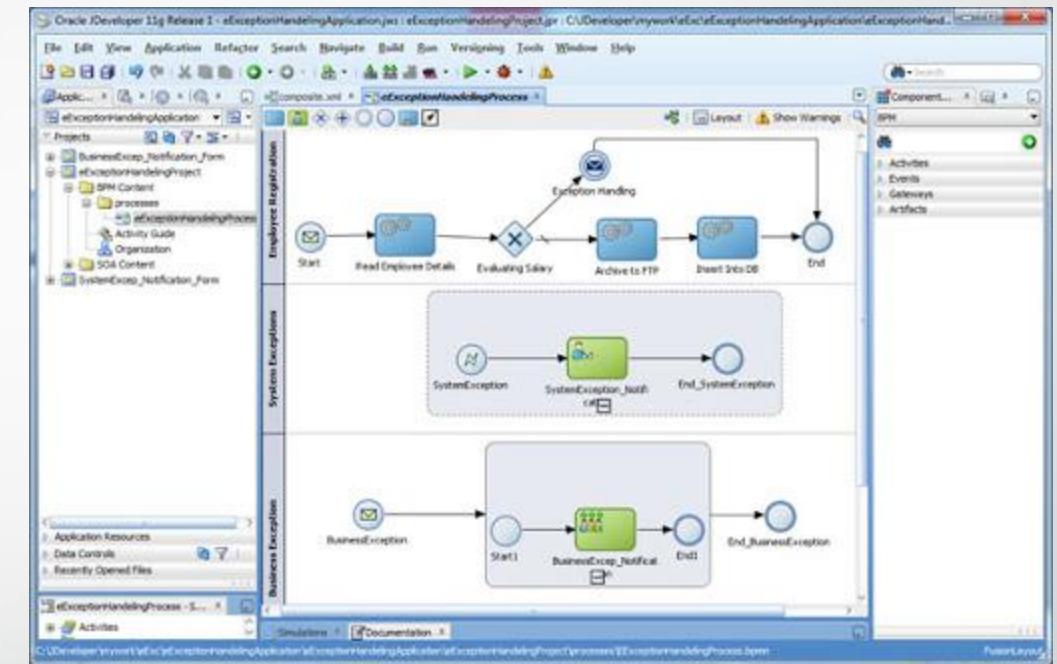
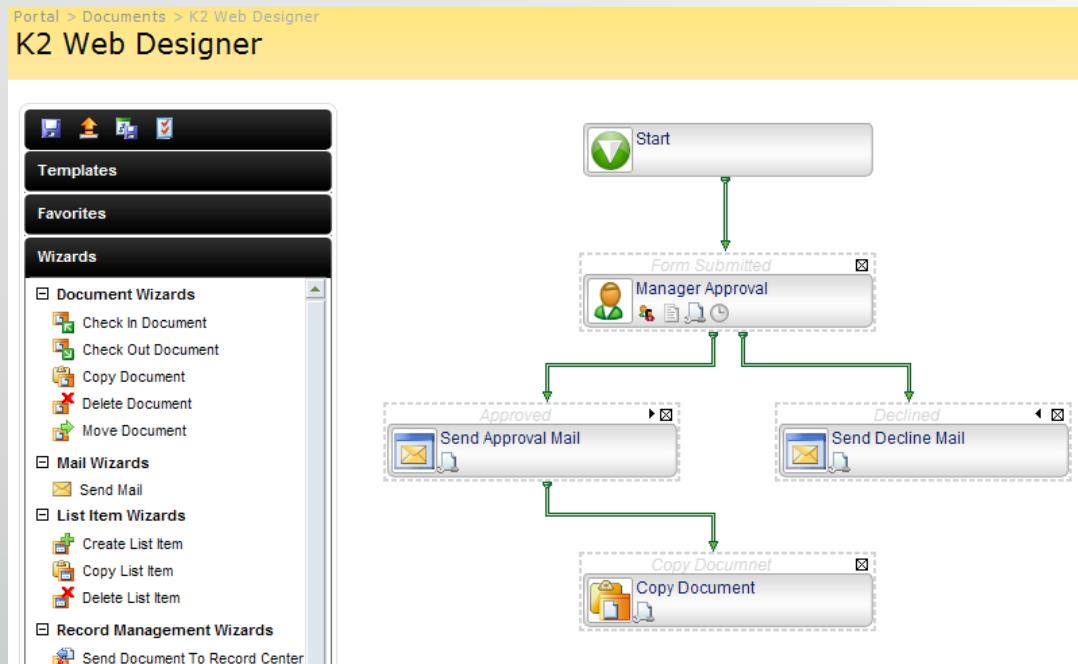
BlueworksLive



PNM**SOFT**
Business Processes in Motion™



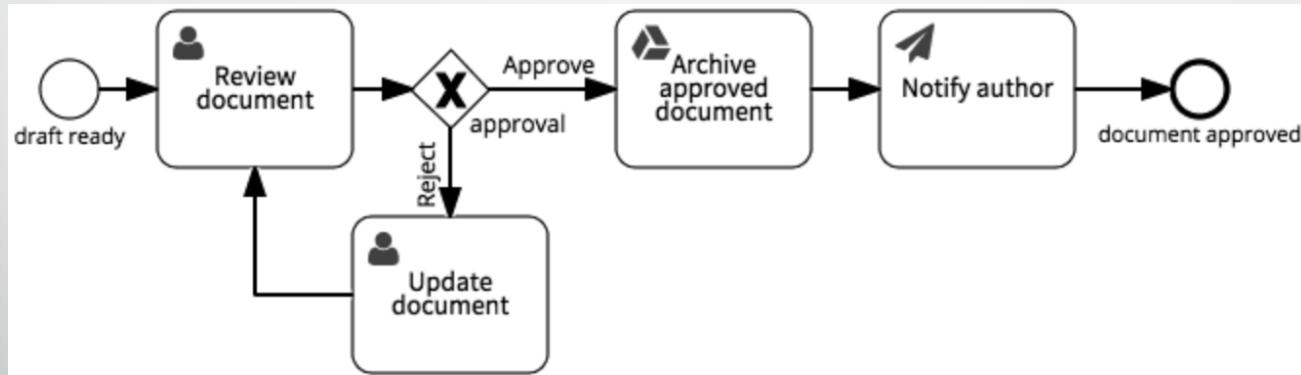
ORACLE®
BPM



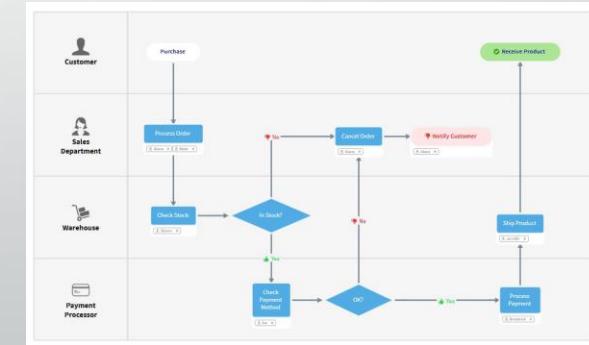
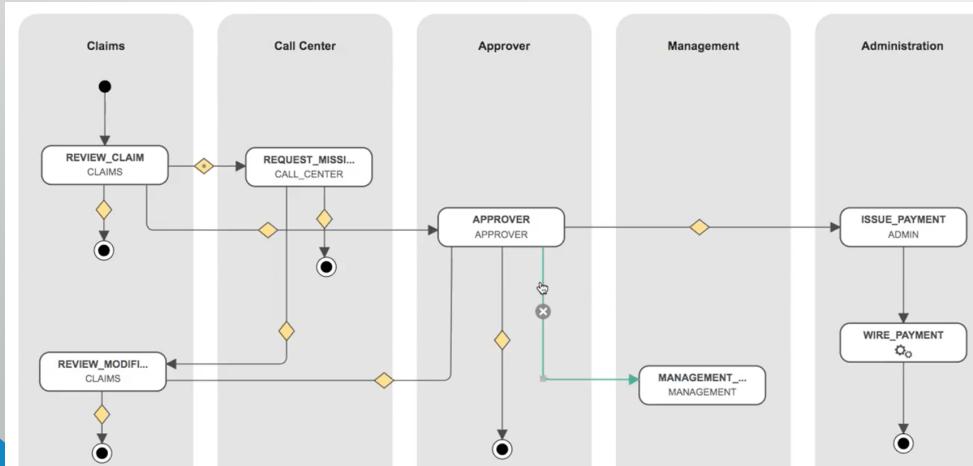
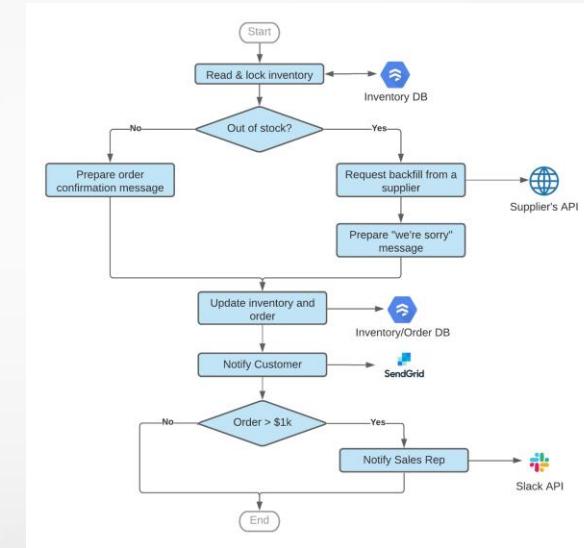
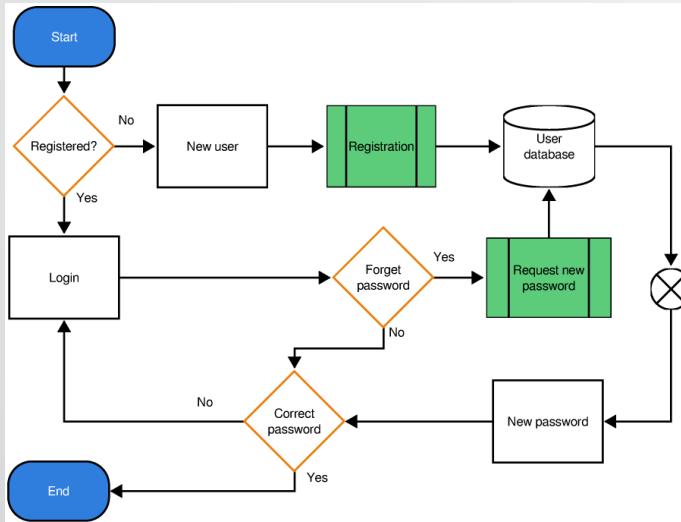
ORACLE®
BPM

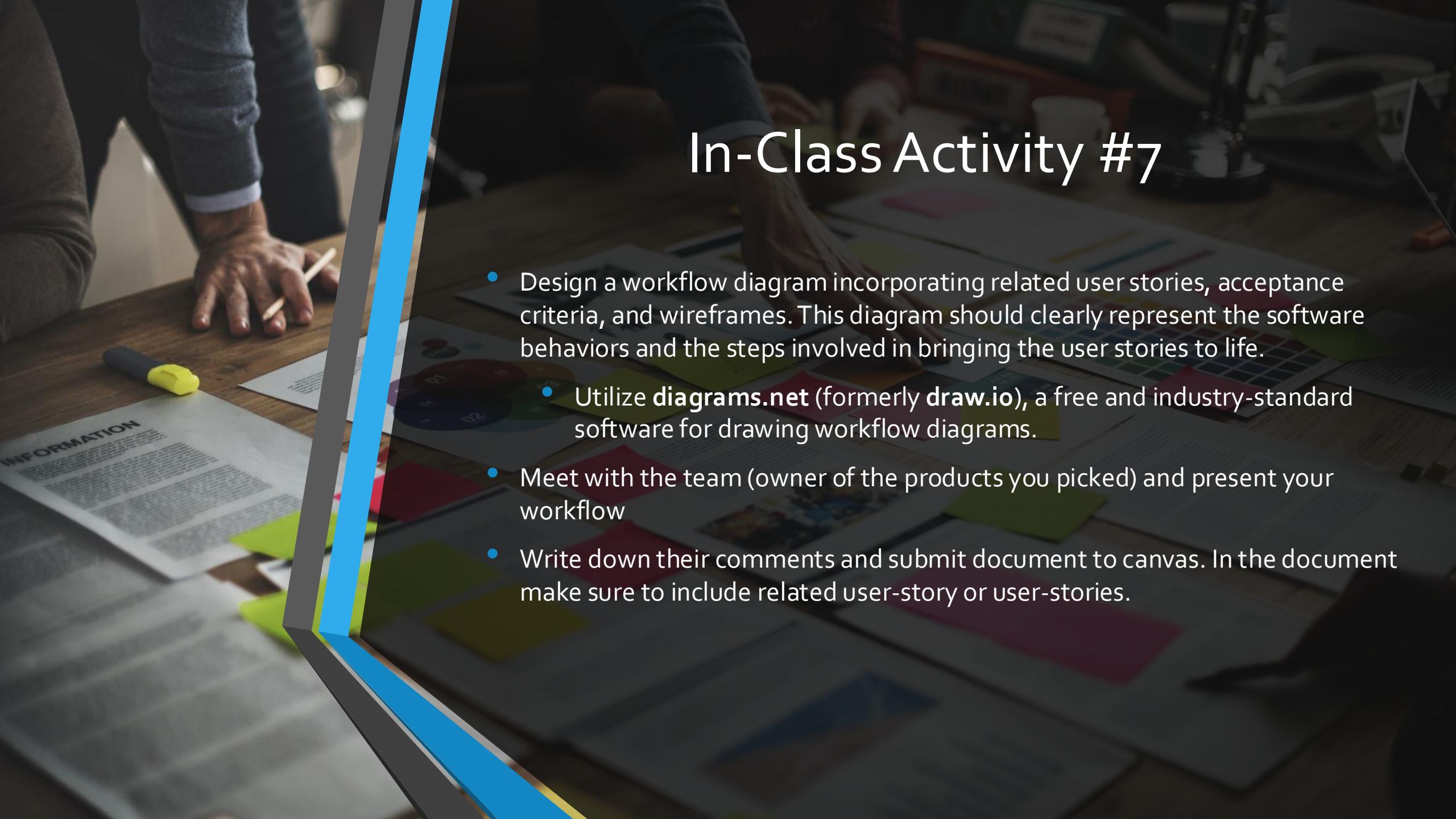
Workflow Process Components

- **Circle:** Marks the start and end points of the workflow sequence.
- **Rectangle:** Denotes instructions or actions.
- **Diamond:** Indicates a yes or no decision, breaking the workflow into two distinct paths.
- **Arrow:** Directs to the next step in the sequence.



Examples



A photograph showing a person's hands working on a wooden desk. On the desk are several pieces of paper, one of which has the word "INFORMATION" printed on it. A yellow highlighter is also visible. The background is slightly blurred.

In-Class Activity #7

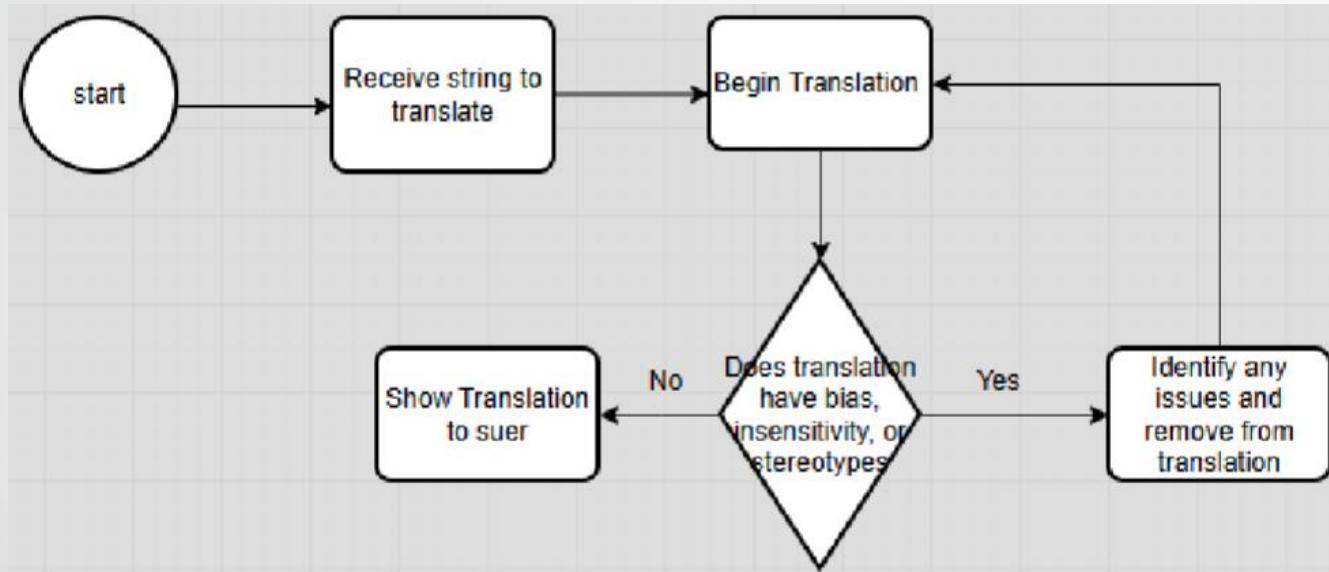
- Design a workflow diagram incorporating related user stories, acceptance criteria, and wireframes. This diagram should clearly represent the software behaviors and the steps involved in bringing the user stories to life.
 - Utilize **diagrams.net** (formerly **draw.io**), a free and industry-standard software for drawing workflow diagrams.
- Meet with the team (owner of the products you picked) and present your workflow
- Write down their comments and submit document to canvas. In the document make sure to include related user-story or user-stories.

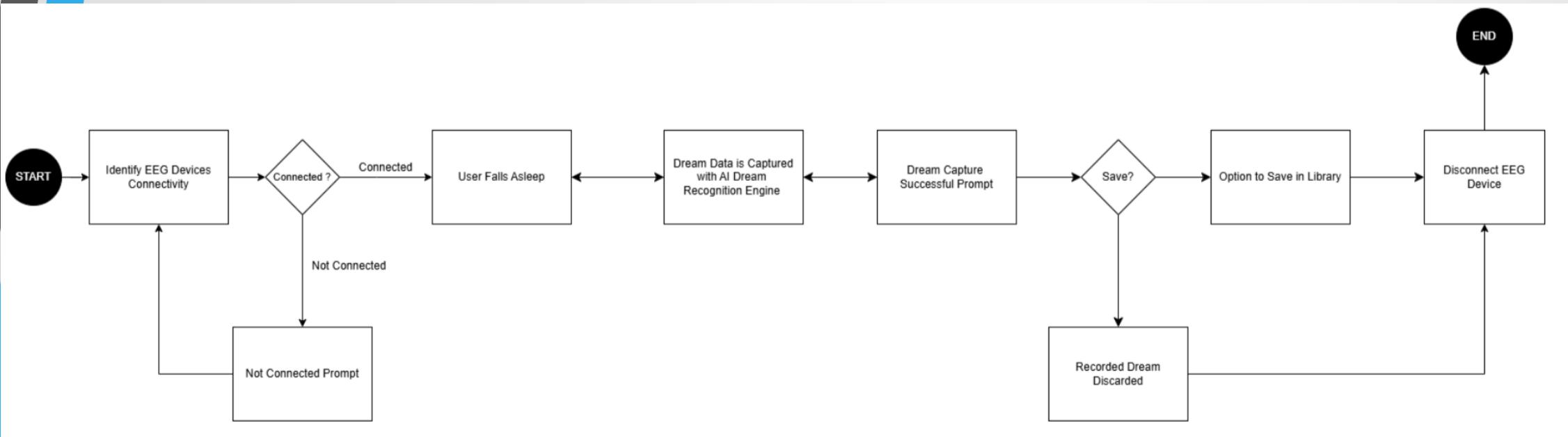
BACS385: 12th Day of Class

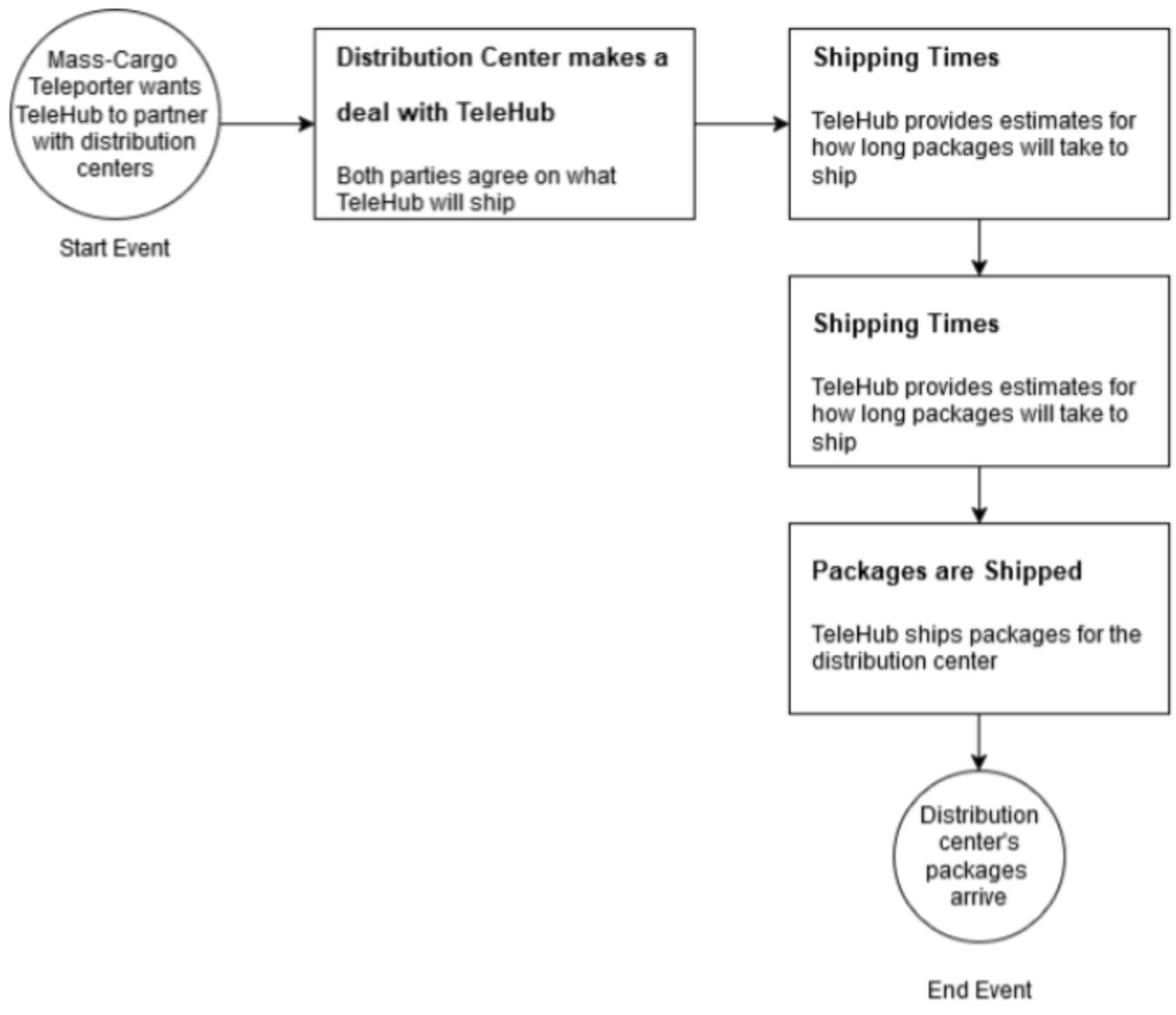
- Agenda -

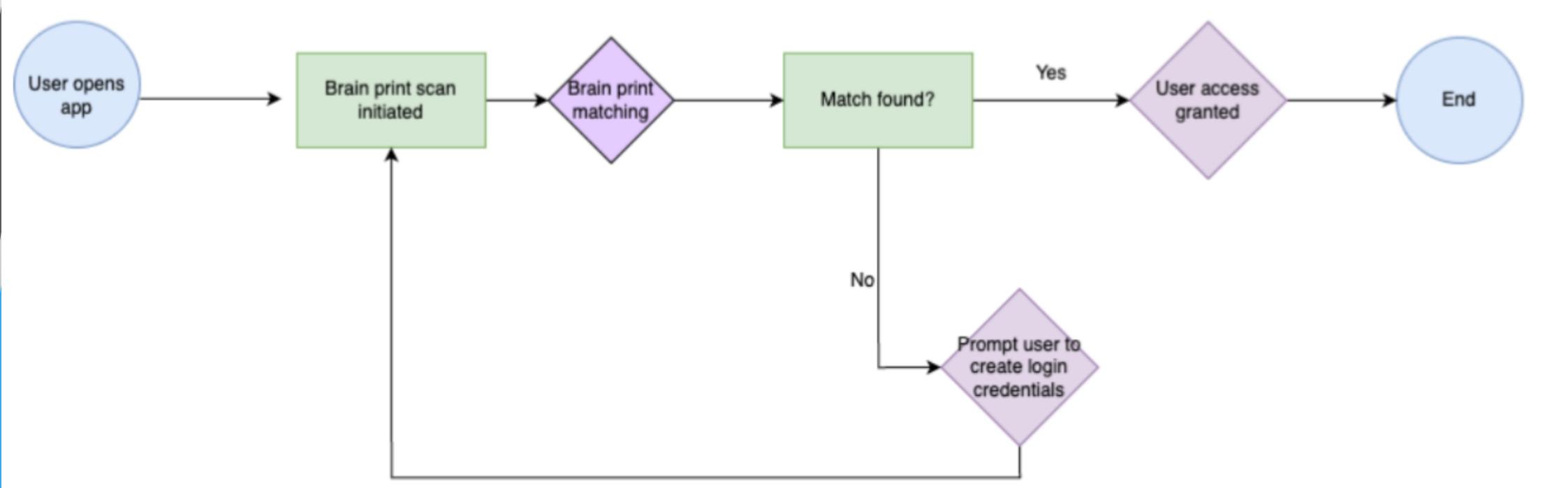
- More Workflows
- Breakout into your Teams: In Class Activity #8

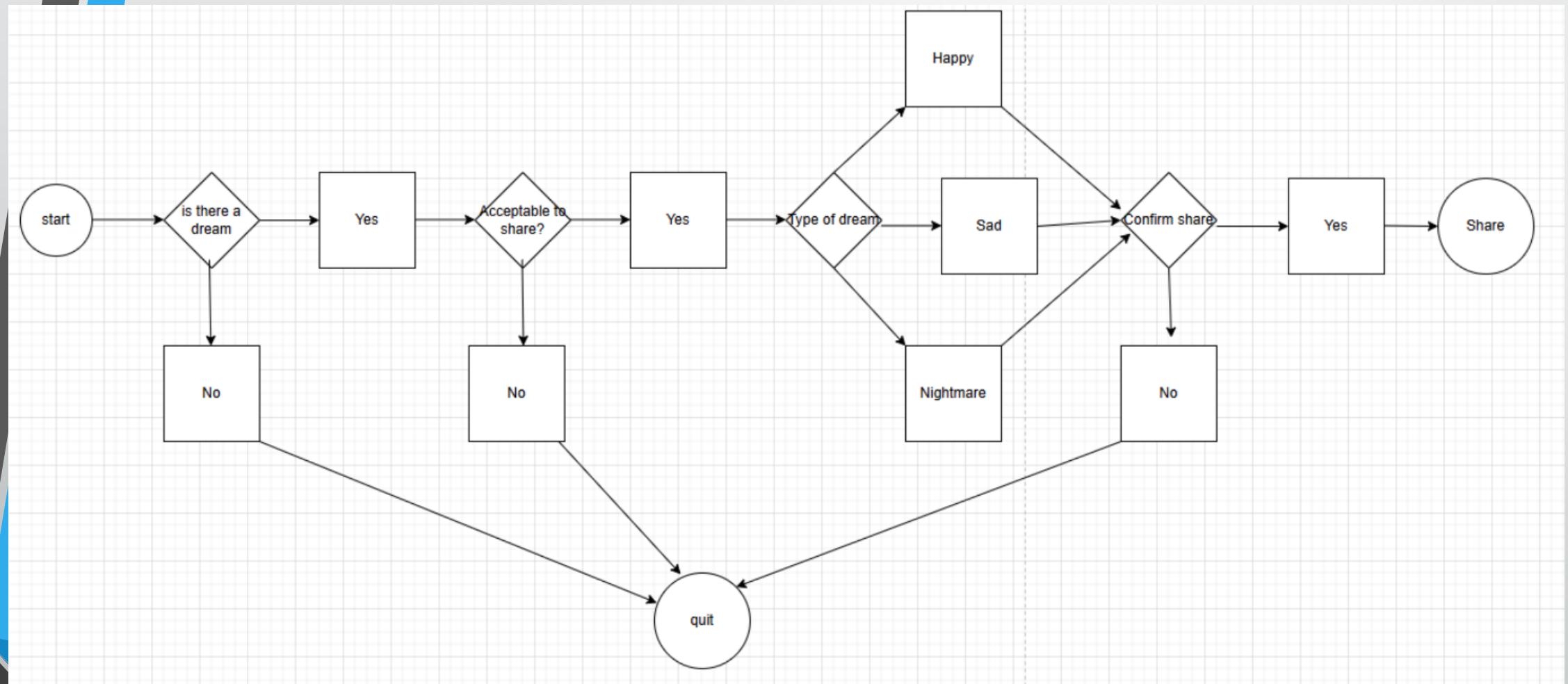


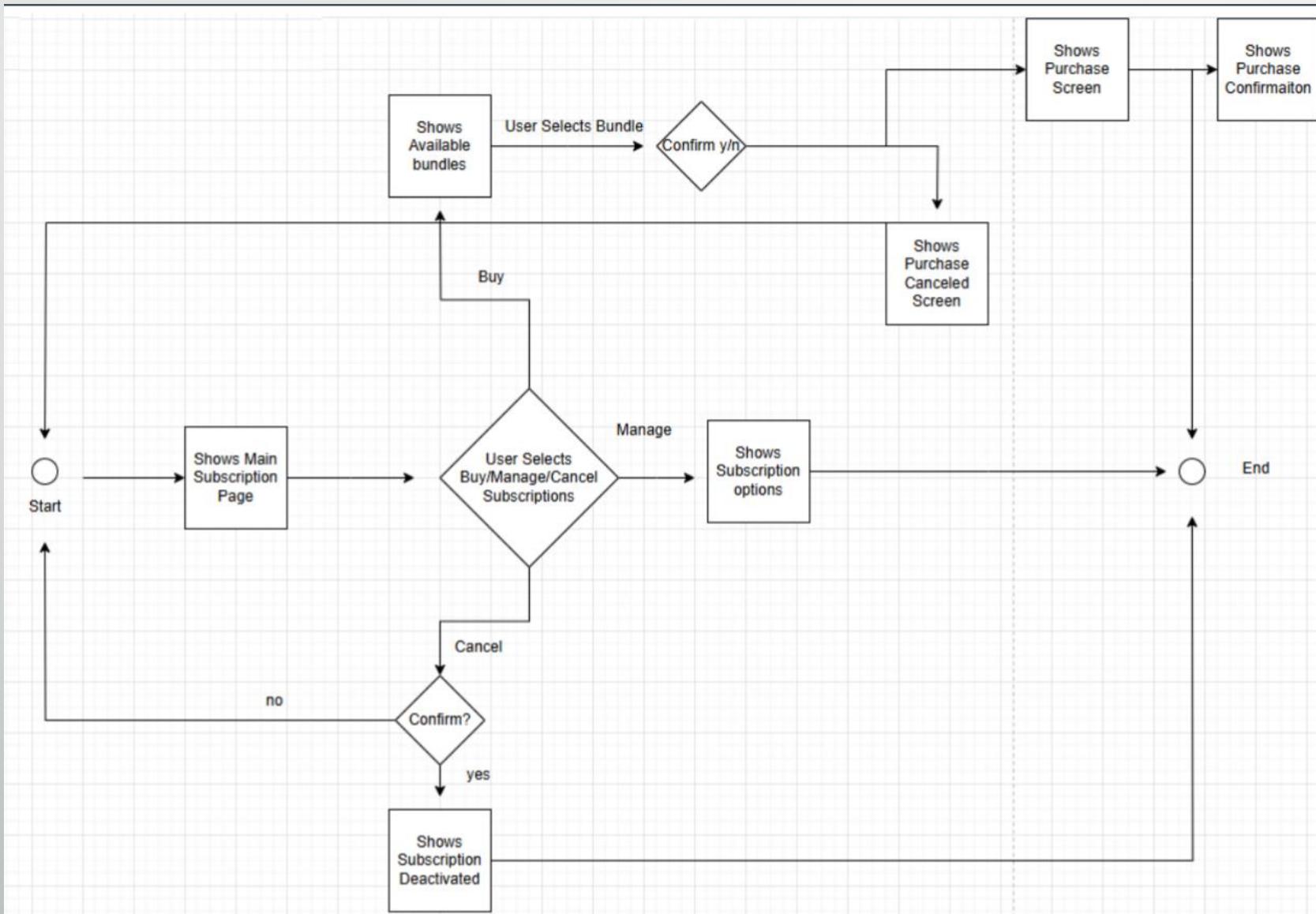












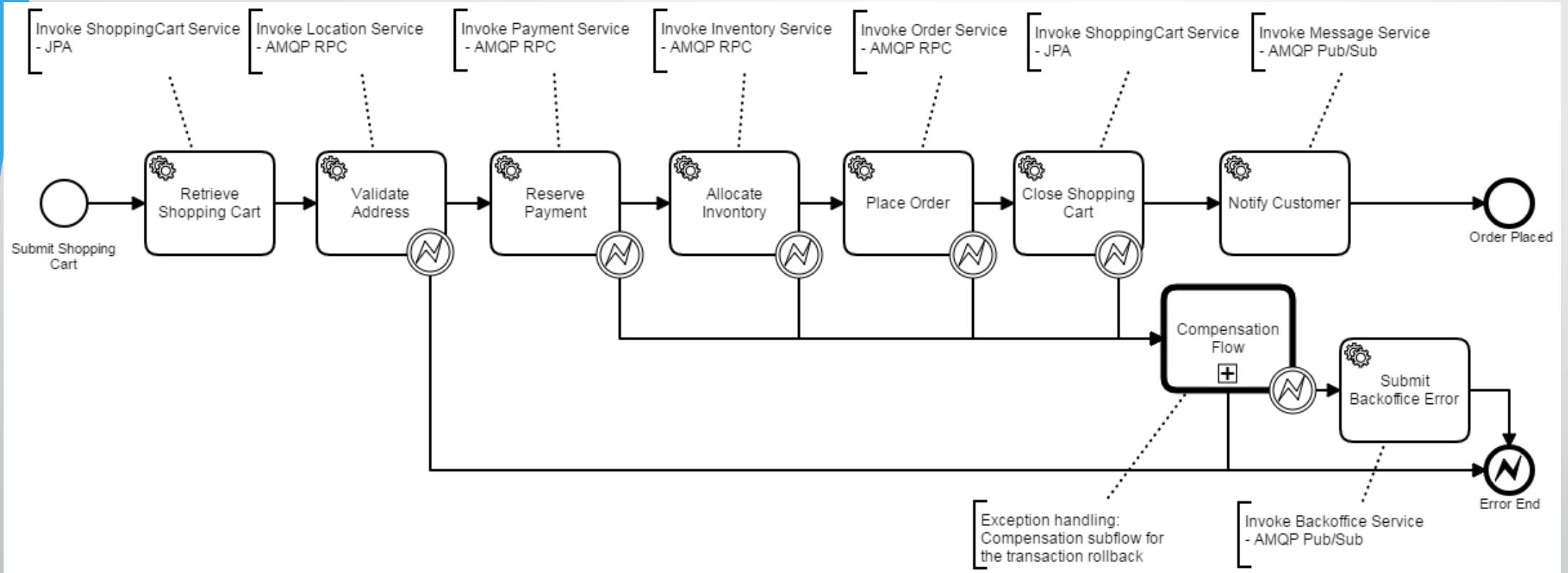
Full Journey vs. Action-Specific Processes

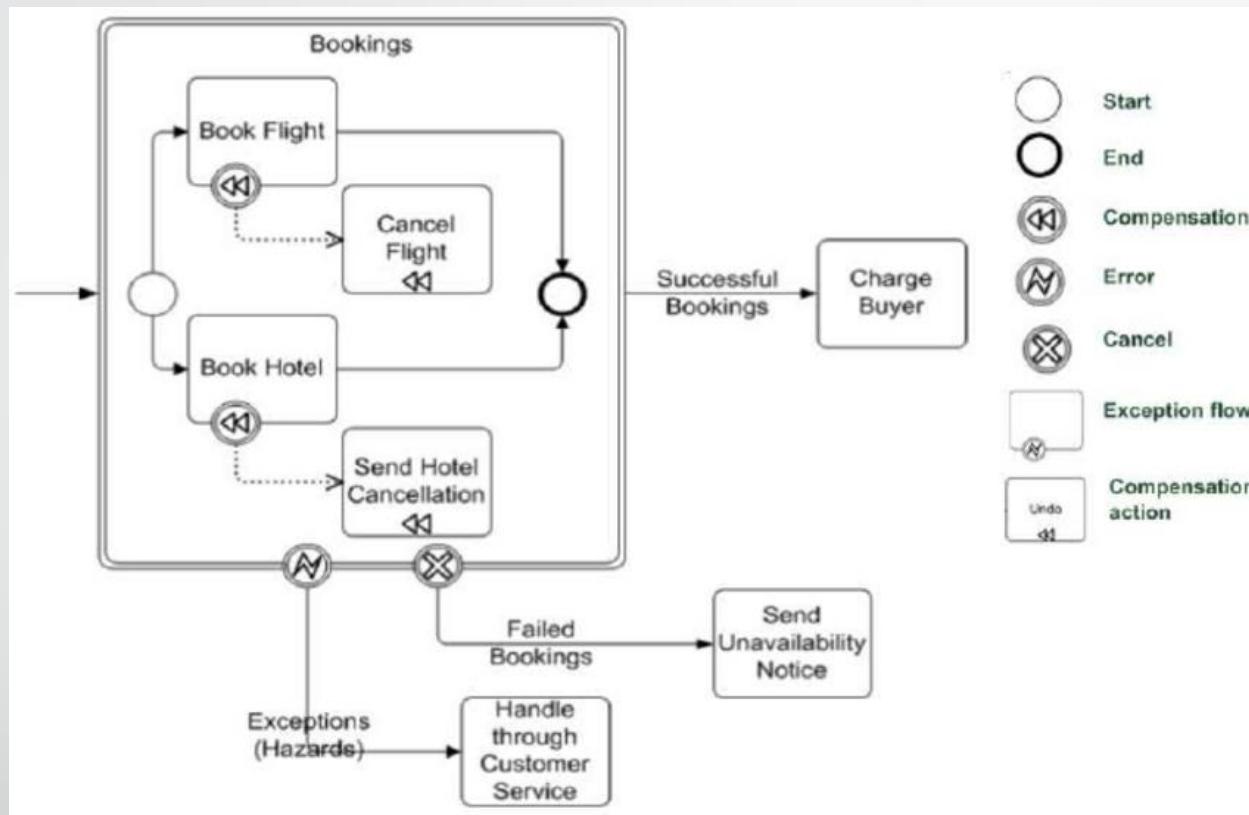
Workflows can be designed to illustrate the complete journey of a user through the entire interface or to detail the behind-the-scenes processes triggered by specific user actions. Here are the two distinct uses:

- **Full User Journey Workflows:** These depict the user's navigation through the entire application, from screen to screen. They are often used for understanding and testing the overall user experience and interface design.
- **Specific Action Workflows:** These focus on the detailed processes that occur as a result of a single user action, such as clicking a button. These workflows typically include system operations that are not visible to the user but are essential for the application's logic, such as data validation, API calls, and conditional logic.
 - Pivotal for developers and system architects as they outline the system-level operations that occur in response to a single user action. These workflows delve into the backend processes like data validation, API interactions, and conditional operations that, while invisible to the user, are crucial for the accurate functioning of the application upon user interaction, such as clicking a button.

Exception Handling in Workflows

- **Exception handling** is the process of responding to the occurrence, during computation, of **exceptions** – anomalous or exceptional conditions requiring special processing – often disrupting the normal flow of program execution.
 - Look at each action block in a workflow and think about how it could fail?
 - Server request gives a warning/error
 - Server credentials changed
 - Connection string (user-name/password) has changed
 - Server is down
 - Connection to Server Timeout – how long is the duration?





In-Class Activity #8

- Create another workflow representing a collection of related user-stories. Please make sure that every step in the workflow should account for potential errors or unexpected user behavior.
- Meet with the team (owner of the products you picked) and present your workflow
- Write down their comments and submit document to canvas. In the document make sure to **include related user-stories**.

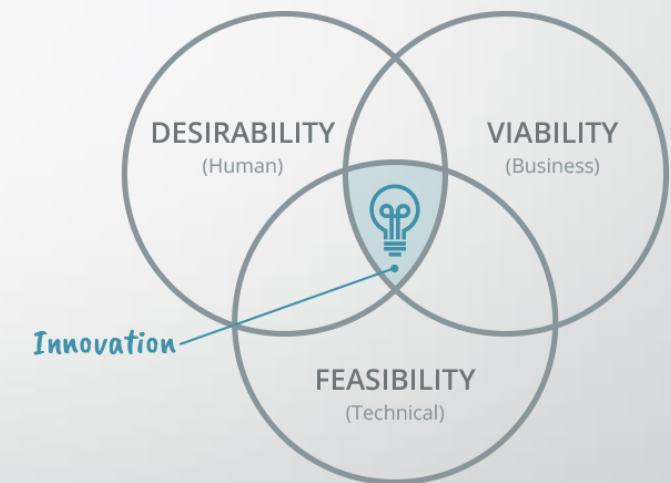
BACS385: 13th Day of Class

- Agenda -

- Feasibility Analysis
- Breakout into your Teams: In Class Activity #9

Feasibility Analysis

- Feasibility analysis is a crucial step in the software development life cycle that ***usually takes place right before the software implementation*** phase.
 - Evaluate the practicality, viability, and potential success of a proposed software project.
 - Assess various factors, such as technical feasibility, economic viability, operational feasibility, schedule feasibility, and legal feasibility.
 - Identify potential obstacles and risks that may impact the success of the project and determine whether the project is worth pursuing.

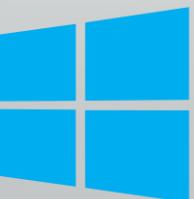


Feasibility in terms of Software Development

- Technical feasibility is the assessment of whether a proposed software project is possible to develop with the available technology and resources.
- Evaluates the technical requirements of the project:
 - Hardware (including Network, Memory, GPU, CPU, etc...)
 - Software (including SDK/API, DB, OS, etc...)
 - Programming Languages Expertise

Examples of Technical Feasibility

- **Compatibility with existing systems:** Technical feasibility analysis assesses whether a proposed software project can integrate with existing systems and software applications.
 - For example, if a new software application is being developed, technical feasibility analysis will evaluate whether the new application is compatible with the existing operating system, database management system, or web server.
 - if the existing system is using both iOS and Android mobile devices, the technical feasibility analysis will evaluate whether the proposed mobile application can run on both operating systems, or whether it needs to be developed separately for each operating system.



Examples of Technical Feasibility

- **Resource availability:** Technical feasibility analysis evaluates whether the necessary resources, such as hardware, software, and programming tools, are available to develop and implement the software project.
 - For example, if a software project requires a particular programming language or software development tool, technical feasibility analysis will evaluate whether those resources are available.



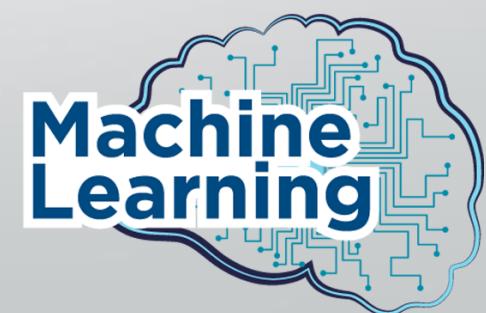
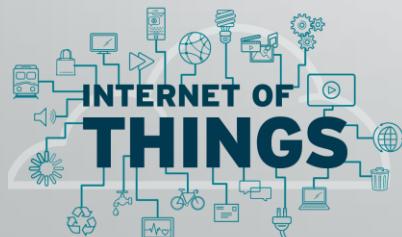
Examples of Technical Feasibility

- **Scalability:** Technical feasibility analysis assesses whether a software project can scale to accommodate future growth and expansion.
 - For example, if a software application is expected to have a large number of users, technical feasibility analysis will evaluate whether the software architecture can support this level of traffic and usage.



Examples of Technical Feasibility

- **Technical expertise:** Technical feasibility analysis assesses whether the development team has the necessary technical expertise to complete the software project.
 - For example, if a software project requires expertise in a specific programming language or software development methodology, technical feasibility analysis will evaluate whether the development team has the necessary skills and experience to successfully complete the project.



Scenario

- You just started a software development company with 3 of your friends running exclusively on Linux operating system. You and your friends are entry level developers (graduated last year) who have programming experience in Java.

A client is ready to give you \$50k to complete the following task in **1 month**:

*"I have a **large CSV file** with **2 million records** that I would like to **analyze sentiment for**, and I prefer to **use Microsoft Excel**. I need a **feasible solution** that **can handle large volumes of data** and **be scaled for future growth**.*

Please provide a proposal outlining the most efficient way to perform sentiment analysis on this data using Excel, with recommendations for optimizing speed, accuracy, and scalability. Please also provide an estimated timeline and budget, including ongoing maintenance and support for future data processing."

What do you do?

Scenario

- You just started a software development company with 3 of your friends running exclusively on Linux operating system. You and your friends are entry level developers (graduated last year) who have programming experience in Java.

A client is ready to give you \$50k to complete the following task in **1 month**:

*"I have a **large CSV file** with **2 million records** that I would like to **analyze sentiment for**, and I prefer to use Microsoft Excel. I need a **feasible solution** that **can handle large volumes of data** and **be scaled for future growth**.*

Please provide a proposal outlining the most efficient way to perform sentiment analysis on this data using Excel, with recommendations for optimizing speed, accuracy, and scalability. Please also provide an estimated timeline and budget, including ongoing maintenance and support for future data processing."

What do you do?

Compatibility

- How can we run Microsoft Excel on Linux?

Resource availability

- Can Microsoft Excel handle a file with 2 million records?
- Does Microsoft Excel have Sentiment Analysis tools?
- Do we have adequate hardware to handle memory and processing speeds to run Sentiment Analysis?

Scalability

- Can Microsoft Excel scale to larger files?
- If not, how can we scale to process larger files?

Technical expertise

- Do we have expertise in Sentiment Analysis (AI/Machine Learning)?
- Can Java help us? Is it the appropriate programming language for Machine Learning?

**Bottom line, can you do this with current in-house expertise and constraints?
Can you complete this in 1 month?**

Group Activity #9

- Sabre Hospitality Solutions is a leading technology provider for the hospitality industry, serving more than 40,000 hotels, resorts, and chains spanning 160 countries. As a project lead, you have been tasked with creating a mobile app using the Sabre APIs (create a dev account by visiting <https://developer.sabre.com/home>) to allow consumers to ***book a hotel, modify bookings, and cancel bookings.***
- To begin the project, you should work with your team of developers to create user-stories that are focused exclusively on the required features of the mobile app. This should include user-stories that cover the booking process, modifying bookings, and canceling bookings.
- Once the user-stories have been created, you should execute a feasibility analysis to determine whether the current set of Sabre APIs will allow you to develop the app and satisfy the user-stories. This analysis should evaluate the technical feasibility of using the Sabre APIs to create the app. To provide proof of feasibility, you should provide an appropriate set of Sabre APIs required for each user-story. This should include a detailed description of the API function and how it will be used to fulfill the requirements of each user-story. You should also include any additional information that may be needed, such as documentation or technical specifications, to support the use of the APIs.
- By executing this feasibility analysis and providing the appropriate set of Sabre APIs required for each user-story, you can ensure that the mobile app development project is technically feasible and can be completed within the specified time frame and budget.

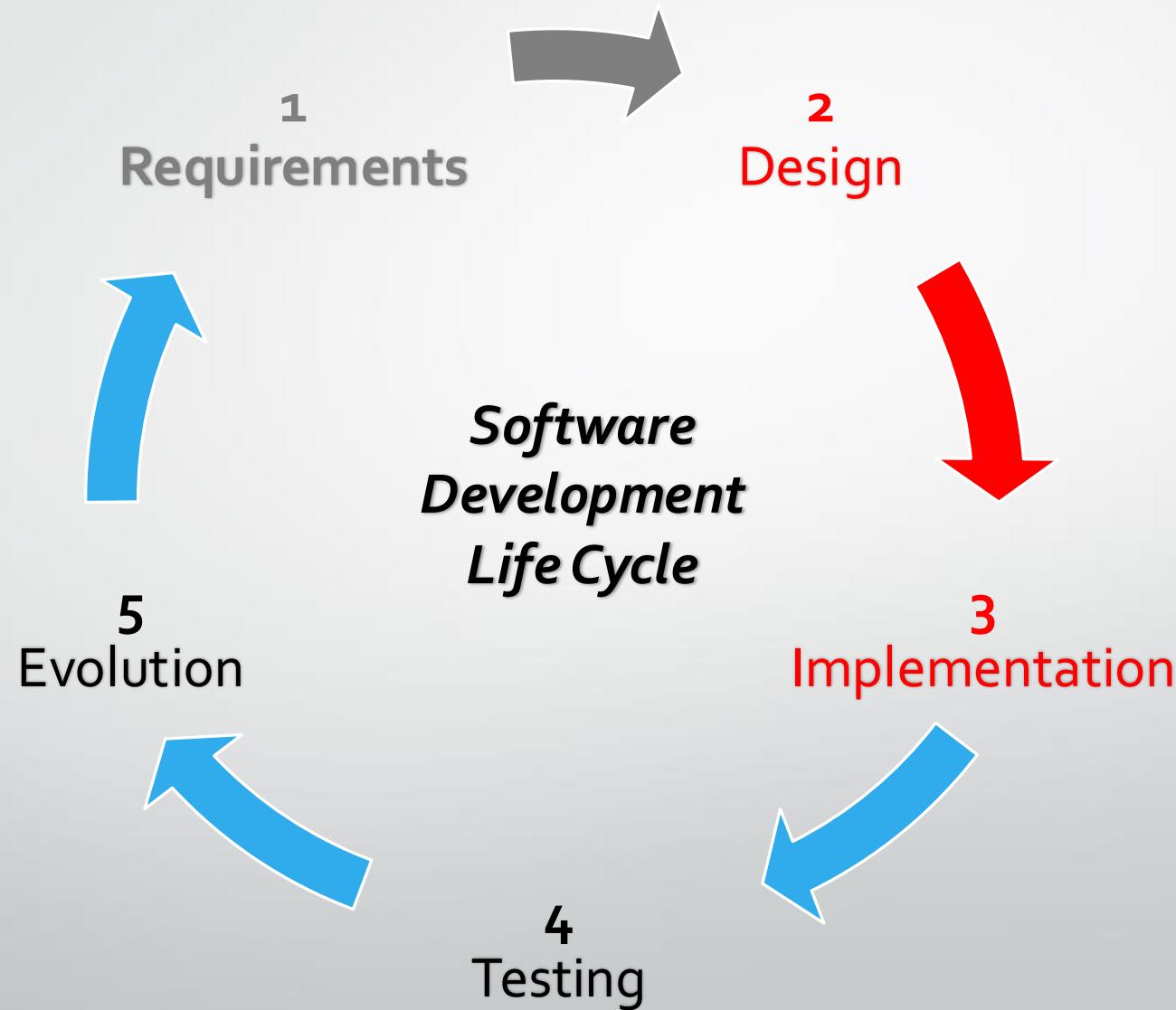


BACS385: 14th Day of Class

- Agenda -

- Software Implementation
- Individual Activity #6

Reminder:



Software Implementation

- “**Implementation**” according to dictionary:
 - the process of putting a decision or plan into effect; execution.
- The process of realizing the software design as a program/software/application/product
- **Programming is the implementation of the design.**
- During the software implementation phase developers create the software
 - Sometimes called ***construction*** phase
- It includes:
 - Internal method design
 - Writing the source code
 - Writing unit tests (sometimes)
 - Testing and debugging code
 - Integration with other components

Scenario

"I am in need of a mobile app that can provide me with real-time updates on the status of my vehicle while it is being serviced at the shop. It should send me alerts on my mobile device to keep me informed of the progress of my vehicle and any changes to the estimated timeline for completion."

As the project lead, you have been tasked with assessing the feasibility of building the mobile app that provides real-time updates on the status of the client's vehicle while it is being serviced at the shop. You have been provided with a list of API sets that your company is currently using.

Scenario

"I am in need of a mobile app that can provide me with real-time updates on the status of my vehicle while it is being serviced at the shop. It should send me alerts on my mobile device to keep me informed of the progress of my vehicle and any changes to the estimated timeline for completion."

As the project lead, you have been tasked with assessing the feasibility of building the mobile app that provides real-time updates on the status of the client's vehicle while it is being serviced at the shop. You have been provided with a list of API sets that your company is currently using.

Which API set would you select?

- 1. GPS tracking APIs:** Track the location of the vehicle and provide updates.
- 2. Messaging APIs:** Send text messages or push notifications to the client's mobile device with updates.
- 3. Payment APIs:** Facilitate payment transactions for services rendered on the vehicle.
- 4. Weather APIs:** Provide updates on weather conditions in the area.
- 5. Social media APIs:** Allow clients to share experiences on social media platforms.
- 6. Parts inventory APIs:** Manage the shop's parts inventory and provide real-time updates on availability.
- 7. Service status APIs:** Allow clients to receive real-time status updates on their vehicle.
- 8. Online booking APIs:** Allow clients to book services online and receive status updates through the app.
- 9. Maintenance reminder APIs:** Send reminders when the vehicle is due for routine maintenance services.
- 10. Rating and review APIs:** Allow clients to rate and review their experiences.

Scenario

"I am in need of a mobile app that can provide me with real-time updates on the status of my vehicle while it is being serviced at the shop. It should send me alerts on my mobile device to keep me informed of the progress of my vehicle and any changes to the estimated timeline for completion."

As the project lead, you have been tasked with assessing the feasibility of building the mobile app that provides real-time updates on the status of the client's vehicle while it is being serviced at the shop. You have been provided with a list of API sets that your company is currently using.

Which API set would you select?

1. **GPS tracking APIs:** Track the location of the vehicle and provide updates.
2. **Messaging APIs:** Send text messages or push notifications to the client's mobile device with updates.
3. **Payment APIs:** Facilitate payment transactions for services rendered on the vehicle.
4. **Weather APIs:** Provide updates on weather conditions in the area.
5. **Social media APIs:** Allow clients to share experiences on social media platforms.
6. **Parts inventory APIs:** Manage the shop's parts inventory and provide real-time updates on availability.
7. **Service status APIs:** Allow clients to receive real-time status updates on their vehicle.
8. **Online booking APIs:** Allow clients to book services online and receive status updates through the app.
9. **Maintenance reminder APIs:** Send reminders when the vehicle is due for routine maintenance services.
10. **Rating and review APIs:** Allow clients to rate and review their experiences.

Messaging APIs

Which API methods would you select?

1. ***sendMessage(message: string, recipient: string)*** - Sends a text message to the specified recipient's mobile device.
2. ***sendMediaMessage(media: object, recipient: string)*** - Sends a rich media message, such as an image or video, to the specified recipient's mobile device.
3. ***getMessageHistory(client: string, conversation: string)*** - Retrieves the message history for the specified client or conversation.
4. ***getMessageStatus(message: string)*** - Retrieves the status of the specified message, such as whether it has been delivered or read.
5. ***sendPushNotification(notification: object, recipient: string)*** - Sends a push notification with the specified content to the specified recipient's mobile device.
6. ***implementSecureAuthentication(username: string, password: string)*** - Implements secure authentication and authorization mechanisms using the specified username and password.

Service status APIs

Which API methods would you select?

1. *getServiceStatus(vehicleId: string)* - Retrieves the current service status of the specified vehicle, such as whether it is being serviced or ready for pickup.
2. *subscribeToServiceStatusUpdates(vehicleId: string, client: string)* - Allows the specified client to subscribe to real-time updates on the specified vehicle's service status.
3. *getEstimatedCompletionTime(vehicleId: string)* - Retrieves the estimated completion time for the specified vehicle service.
4. *requestStatusChange(vehicleId: string, newStatus: string)* - Allows clients to request changes to the specified vehicle's service status, such as rescheduling an appointment.
5. *getVehicleLocation(vehicleId: string)* - Retrieves the location of the specified vehicle, if available.
6. *getPaymentStatus(vehicleId: string)* - Retrieves the payment status of the specified vehicle service, such as whether payment has been received or is outstanding.

Individual Activity #5

As a project lead, your objective is to select API methods suitable for the Bowling Ally mobile app that meet the following categories:

- **Third-Party Authentication API:** This API should enable users to log in using their Google, Facebook, or other third-party accounts. It is critical that this API supports OAuth 2.0 or equivalent protocols to ensure secure user authentication.
- **Instagram API:** This API should have authentication capabilities that allow users to authenticate their Instagram accounts and post pictures of their scores to their Instagram profiles.

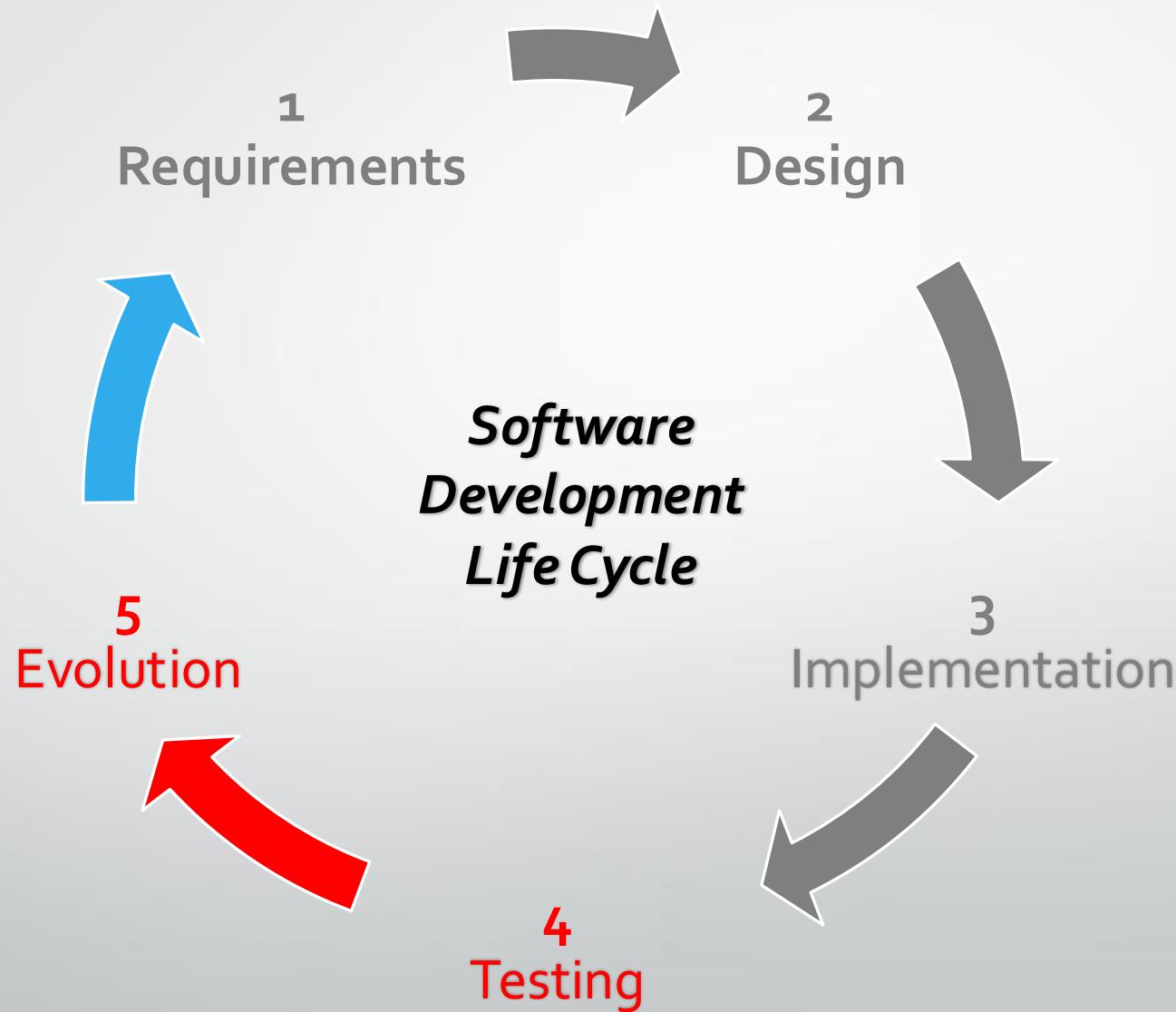
To meet these requirements, you need to identify the required API methods that you envision using in code. Based on your understanding of user stories, acceptance criteria, wireframes, and workflows, choose appropriate API methods for each category (i.e., items 1 and 2) and provide an explanation for your decision.

BACS385: 15th Day of Class

- Agenda -

- Software Testing and Evolution
- Breakout into your Teams: In Class Activity #10

Reminder:

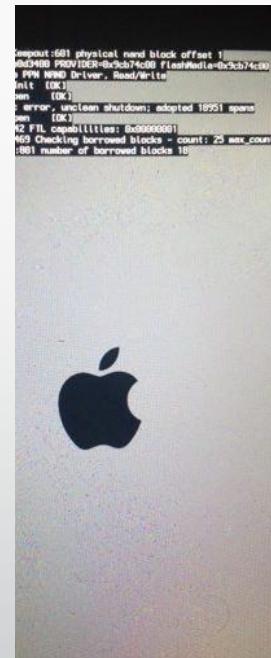


Software Testing

- The process of exercising a program with the specific intent of finding errors prior to delivery to the end user
- Must be planned carefully to avoid wasting development time and resources, and conducted systematically
- Intended to show that a system conforms to its specification and meets the requirements of the system customer.
- System testing involves executing the system with *test cases based on* user-stories + acceptance criteria, workflows and user-interface design that are **derived from the specification** of the real data to be processed by the system.
- An early start to testing reduces the cost, time to rework and error free software that is delivered to the client.

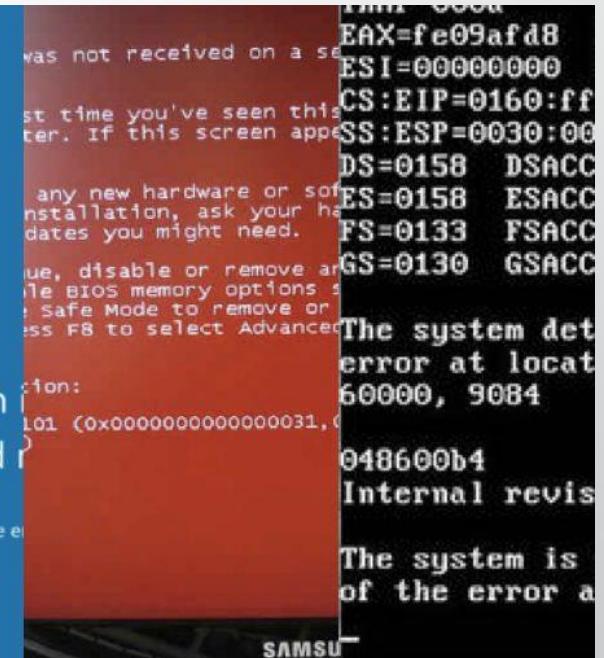
What Testing shows

- Errors/defects/bugs
- Requirements Conformance
- Performance
 - An Indication of Quality



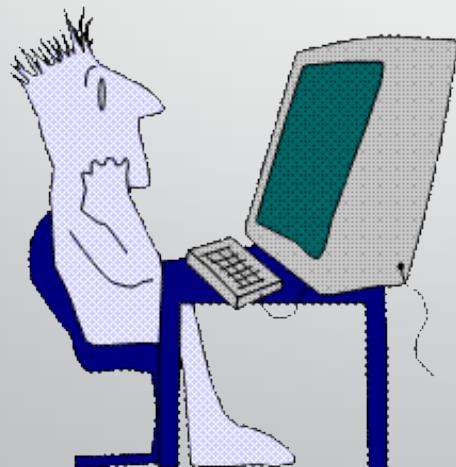
Your PC ran into a problem and needs to restart.
We're sorry this happened.

You can search for the error online or contact support.



Who Tests the Software?

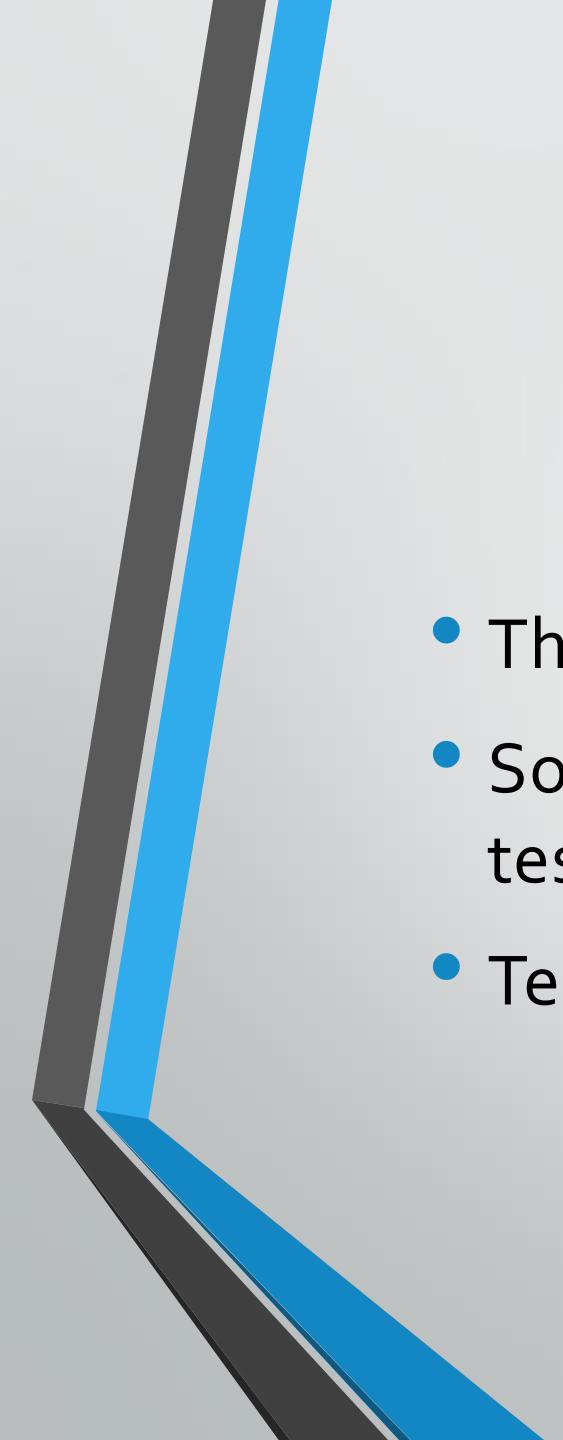
- ❖ Developer
 - ❖ Understands system but, will test “gently”
 - ❖ Driven by “delivery”
- ❖ Independent Tester
 - ❖ Must learn about the system
 - ❖ Will attempt to break it
 - ❖ Driven by quality



Who Tests the Software?

In most cases, following professionals are involved in testing of a system within their respective capacities:

- Programmer/Software Developer
- Software Tester
- Project Leader/Manager
- End User/Customer



Misconceptions:

- The developer should do no testing at all
- Software should be “tossed over the wall” to stranger who will test it mercilessly
- Testers are not involved with the project until it is ready

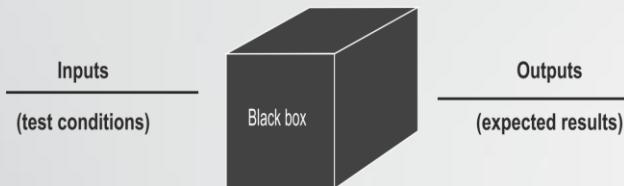
When to Stop Testing?

When to Stop Evolving?

- Software is inherently flexible and can change.
- As requirements change through changing business circumstances, the software that supports the business **must** also evolve and change.

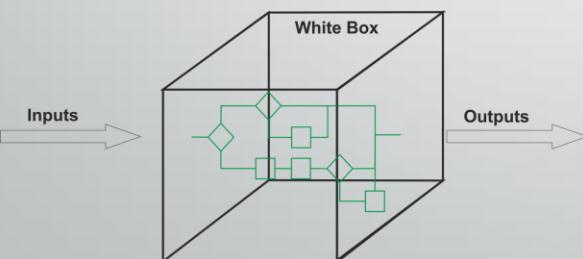
- Testing is potentially endless
 - We cannot test until all the defects are unearthed and removed – which is impossible
- At some point, we have to stop testing and ship the software
 - The question is, When?
- Realistically, testing is a trade-off between budget, time and quality
- It is driven by profit models

Testing Techniques



Black-box testing:

- The technique of testing without having any knowledge of the interior workings of the application is Black Box testing.
- The tester does not have access to the source code.
- A tester will interact with the system's user interface by providing inputs and examining outputs without knowing how and where the inputs are worked upon.



White-box testing:

- Detailed investigation of internal logic and structure of the code.
- A tester needs to have knowledge of the internal working of the code.
- A tester needs to have a look inside the source code and find out which unit/chunk of the code is behaving inappropriately.
- Test case derived from software specification and requirements

What do you need to start testing?

- “Test Cases” are required to perform testing
- A **TEST CASE** is a set of conditions or variables under which a tester will determine whether a system under test satisfies requirements or works correctly.

Test Scenario ID	Login-1		Test Case ID	Login-1A			
Test Case Description	Login – Positive test case		Test Priority	High			
Pre-Requisite	A valid user account		Post-Requisite	NA			
Test Execution Steps:							
S.No	Action	Inputs	Expected Output	Actual Output	Test Browser	Test Result	Test Comments
1	Launch application	https://www.facebook.com/	Facebook home	Facebook home	IE-11	Pass	[Priya 10/17/2017 11:44 AM]: Launch successful
2	Enter correct Email & Password and hit login button	Email id : test@xyz.com Password: *****	Login success	Login success	IE-11	Pass	[Priya 10/17/2017 11:45 AM]: Login successful

Test Case Elements

A test case can have the following elements. Note, however, that a test management tool is normally used by companies and the format is determined by the tool used.

Element	Definition
Test Suite ID	The ID of the test suite to which this test case belongs.
Test Case ID	The ID of the test case.
Test Case Description	The summary / objective of the test case.
Prerequisites	Any prerequisites or preconditions that must be fulfilled prior to executing the test.
Date of Execution	The date of execution of the test.
Executed By	The name of the person who executed the test.
Execution Steps	Step-by-step procedure to execute the test.

Test Case Elements - Execution Steps

- The following is a list of elements that are part of each execution step

Elements	Definition
Order/Index	The ID of the test suite to which this test case belongs.
Action	Description on how to achieve this step
Input	Input data needed (e.g., URL, field data, attachment, etc...)
Expected Output	A screenshot or a description of what should be the expected output of the action
Actual Output	A screenshot or a description of the actual output of the action
Test Environment	The environment (Hardware/Software/Network) in which the test was executed (e.g., Operating System, Mobile Platform, Web Browser, Screen-Resolution, etc...)
Results	Pass or Fail. Other statuses can be 'Not Executed' if testing is not performed and 'Blocked' if testing is blocked.
Comments	Any comments on the test case or test execution step

Test Case Example

Project Name:	Google Email					 www.SoftwareTestingMaterial.com			
Module Name:	Login								
Reference Document:	If any								
Created by:	Rajkumar								
Date of creation:	DD-MMM-YY								
Date of review:	DD-MMM-YY								
TEST CASE ID	TEST SCENARIO	TEST CASE	PRE-CONDITION	TEST STEPS	TEST DATA	EXPECTED RESULT	POST CONDITION	ACTUAL RESULT	STATUS (PASS/FAIL)
TC_LOGIN_001	Verify the login of Gmail	Enter valid User Name and valid Password	1. Need a valid Gmail Account to do login	1. Enter User Name 2. Enter Password 3. Click "Login" button	<Valid User Name> <Valid Password>	Successful login	Gmail inbox is shown		
TC_LOGIN_001	Verify the login of Gmail	Enter valid User Name and invalid Password	1. Need a valid Gmail Account to do login	1. Enter User Name 2. Enter Password 3. Click "Login" button	<Valid User Name> <Invalid Password>				
TC_LOGIN_001	Verify the login of Gmail	Enter invalid User Name and valid Password	1. Need a valid Gmail Account to do login	1. Enter User Name 2. Enter Password 3. Click "Login" button	<Invalid User Name> <Valid Password>	A message "The email and password you entered don't match" is shown			
TC_LOGIN_001	Verify the login of Gmail	Enter invalid User Name and invalid Password	1. Need a valid Gmail Account to do login	1. Enter User Name 2. Enter Password 3. Click "Login" button	<Invalid User Name> <Invalid Password>				

Test Case Example

Test Case ID	BU_001	Test Case Description	Test the Login Functionality in Banking		
Created By	Mark	Reviewed By	Bill	Version	2.1
QA Tester's Log	Review comments from Bill incorporate in version 2.1				
Tester's Name	Mark	Date Tested	1-Jan-2017	Test Case (Pass/Fail/Not)	Pass
S #	Prerequisites:		S #	Test Data	
1	Access to Chrome Browser		1	Userid = mg12345	
2			2	Pass = df12@434c	
3			3		
4			4		
Test Scenario	Verify on entering valid userid and password, the customer can login				
Step #	Step Details	Expected Results	Actual Results	Pass / Fail / Not executed / Suspended	
1	Navigate to http://demo.guru99.com	Site should open	As Expected	Pass	
2	Enter Userid & Password	Credential can be entered	As Expected	Pass	
3	Click Submit	Customer is logged in	As Expected	Pass	
4					

In-Class Activity #10

- Select one of your workflows, and create a test-case based on each workflow step with keeping its associated user-story and acceptance criteria in mind.
- Meet with the team (owner of the products you picked) and present your testcase
- Write down their comments and submit document to canvas. In the document make sure to include related workflow, user-story or user-stories.

BACS385: 19th
Day of Class

Agenda

- Finish In-Class Activity #10

Agenda

- In-Class Activity #11
- ***Quiz 2: Monday March 25th***
 - Covers:
 - *Software Design*
 - *Workflows*
 - *Software Implementation*
 - *Software Testing & Evolution*
 - *Test-Cases*

Group Activity #11

Your team will present your system's product unveiling, utilizing Figma and Google Slides. The purpose of this presentation is to offer an informal overview of your work on the "*hypothetically impossible system*." This is an opportunity to share your efforts with the class and compile all created artifacts into a cohesive 15-minute presentation. Your objective is to convincingly present and sell the concept to the audience.

- **Outline of Presentation:**

- **Slide 1:** Highlight the necessity of the app and its significance.
- **Slide 2:** Detail what the app is and its purpose.
- **Slide 3:** The app's target users and roles.
- **Multiple Slides:** A specific scenario/use-case, demonstrating the app's functionality screen by screen, including user interactions and outcomes.
 - Attach relevant user stories at each screen's end (acceptance criteria excluded).
- **Final Slide:** Cover the N-Tier system architecture (technology stack) of your product, divide your system into the following tiers:
 - **Presentation Tier:** Specify technologies used on the client side, such as HTML5, JavaScript, CSS, etc. **Indicate if the system is a mobile or desktop application and the operating system it supports (e.g., Microsoft Windows, Linux, Android, Apple OS).**
 - **Logic Tier:** Detail the server-side programming language and frameworks used (e.g., C#, Java, Python).
 - **Data Tier:** State the database technologies utilized (e.g., Oracle DB, MS DB, MongoDB).

- **Presentation Guidelines:**

- Limit to **15 minutes**.
- Mandatory participation from all team members; non-participation results in a **0%** grade.
- Submit and present live during class at the assigned times.

- **Live Presentation Schedule (random selection on Friday March 1st):**

- Monday March 18, 10:10 AM: TBD
- Monday March 18, 10:40 AM: TBD
- Wednesday March 20, 10:10 AM: TBD
- Wednesday March 20, 10:40 AM: TBD
- Friday March 22, 10:10 AM: TBD
- Friday March 22, 10:40 AM: TBD

- **Submission:**

- Share the link to your Google Slides on Canvas.

BACS385: 21st Day of Class

Agenda

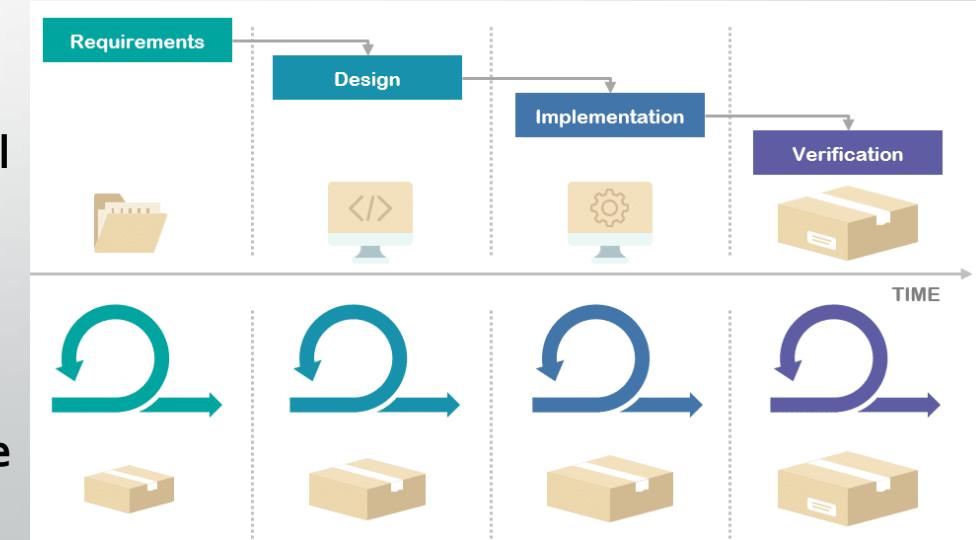
- Intro to Iterative development
- Intro to Scrum
- In-Class Activity #12

Iterative development

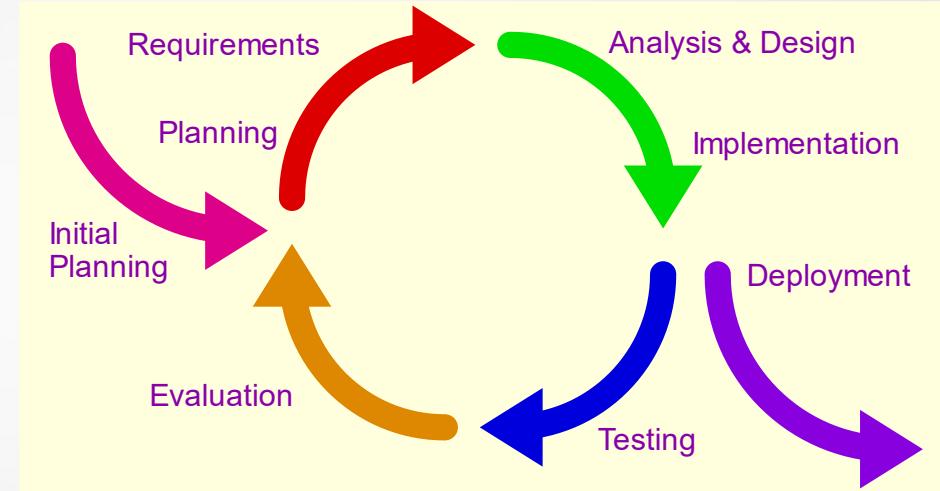
An iteration is a distinct sequence of activities based on an established plan and evaluation criteria, resulting in a executable release

- System requirements **ALWAYS evolve in the course of a project** so process iteration where earlier stages are reworked is always part of the process for large systems
- **Iteration** can be applied to any of the generic process models

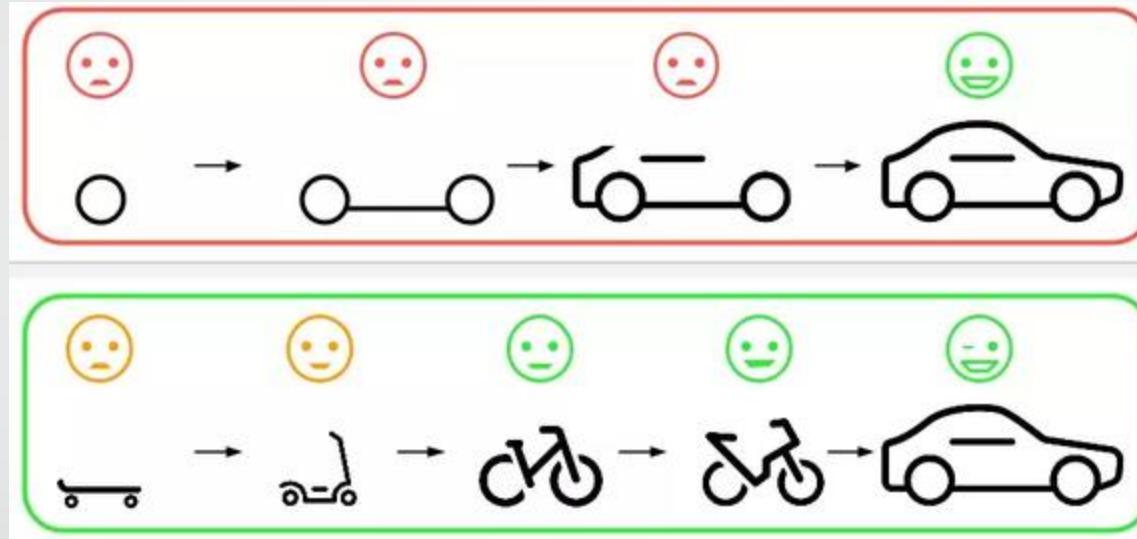
Waterfall



Iterative

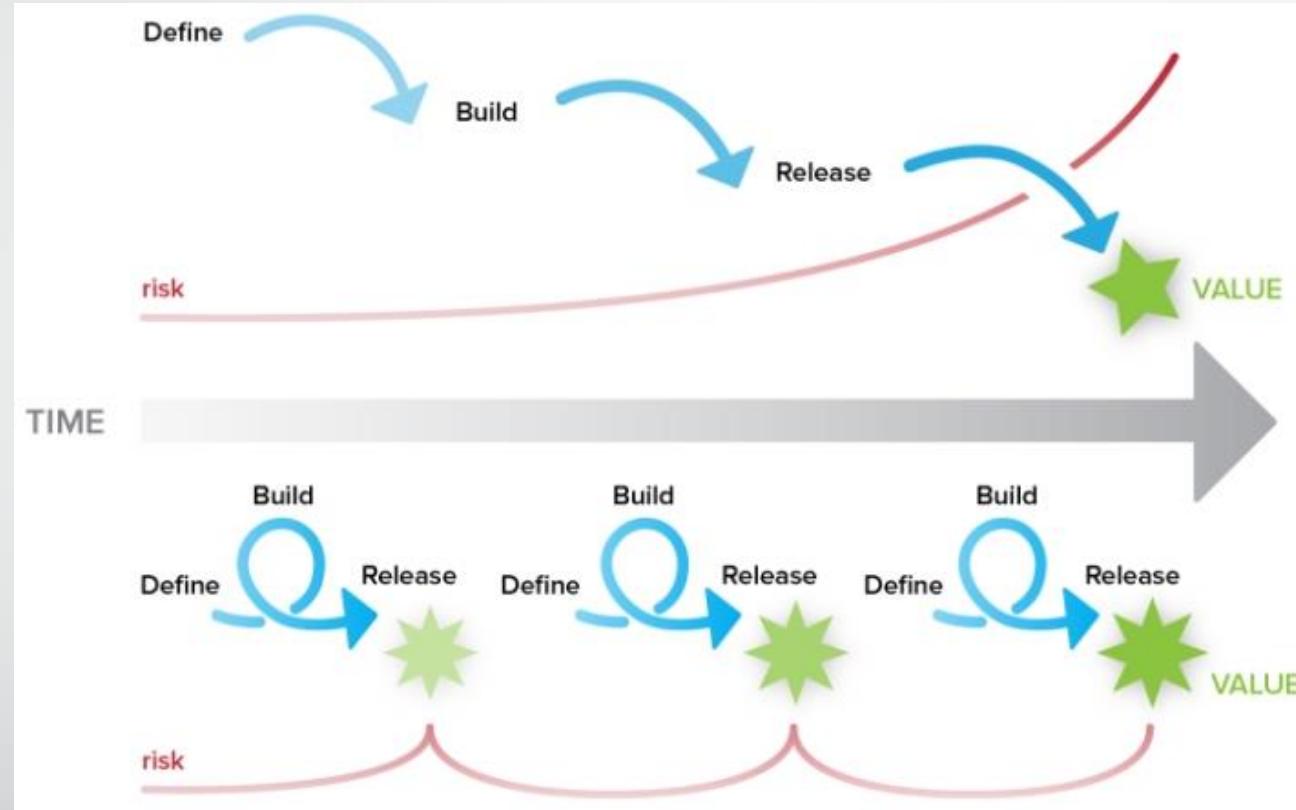


Waterfall Model is a traditional incremental development approach, where the stages (Requirements gathering, design, implement, test, maintain) increment after one another which eventually transforms the concept into a final outcome/product.



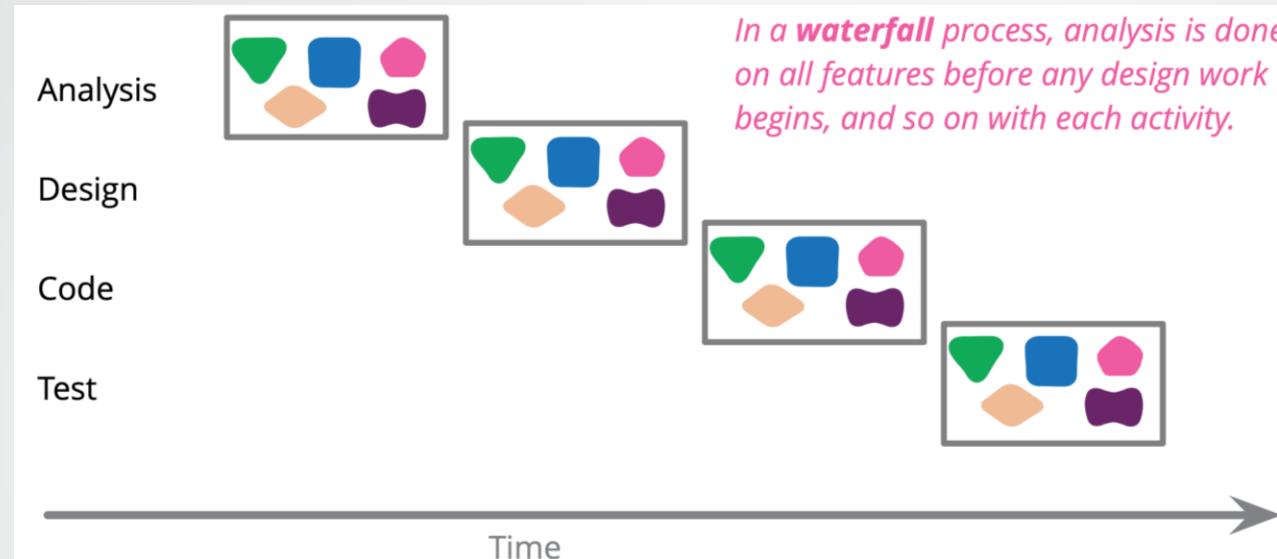
Iterative model is an approach where the focus is put on creating a working prototype first and adding features in development cycles where the Incremental Development stages (Waterfall model) are done for every cycle

Waterfall Model

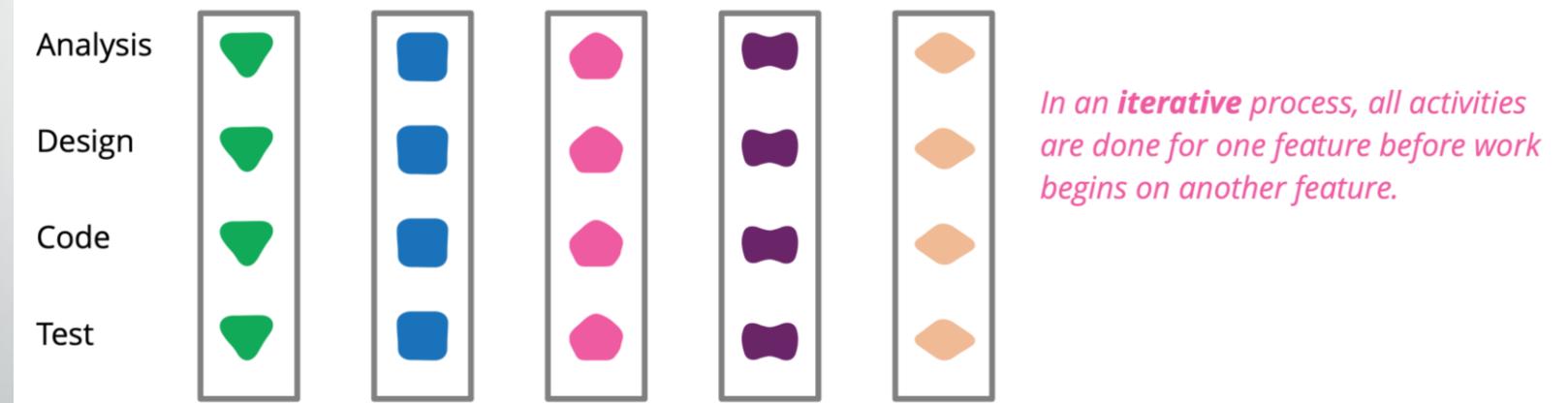


Iterative Model

Waterfall Model



Iterative Model



Scrum

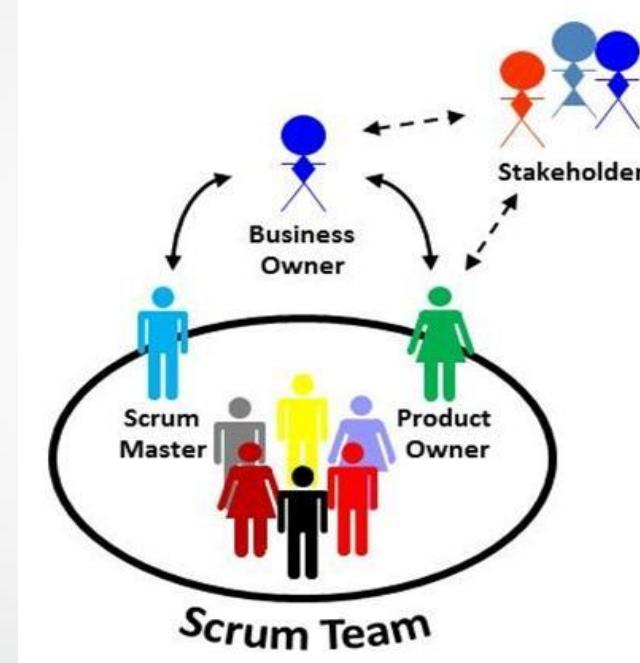


Scrummage or “scrum” is a term used in rugby, when the forwards from each side bind with their teammates and crash head-on into their opponents, to gain possession of the ball that’s rolled between them.

- **Scrum** is a pre-defined development lifecycle based on iterative development.
- **Scrum** is an implementation of the iterative philosophy pertaining specifically to project management.
- **Scrum** is an iterative framework that focuses on managing iterative development.

Components of Scrum

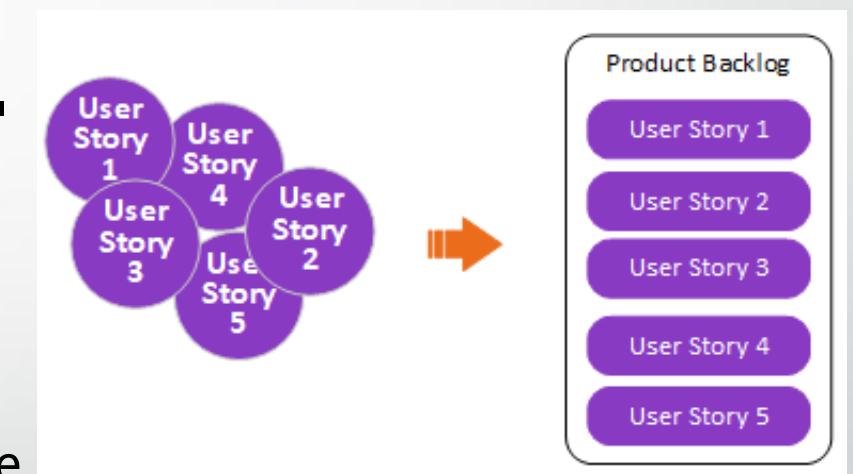
- Roles
 - Product owner
 - Scrum Master
 - Development team
- Activities
 - Sprint Planning
 - Sprint Review
 - Sprint Retrospective
 - Daily scrum
- Artifacts
 - Product Backlog
 - Release/Sprint Backlog
 - Burndown Chart



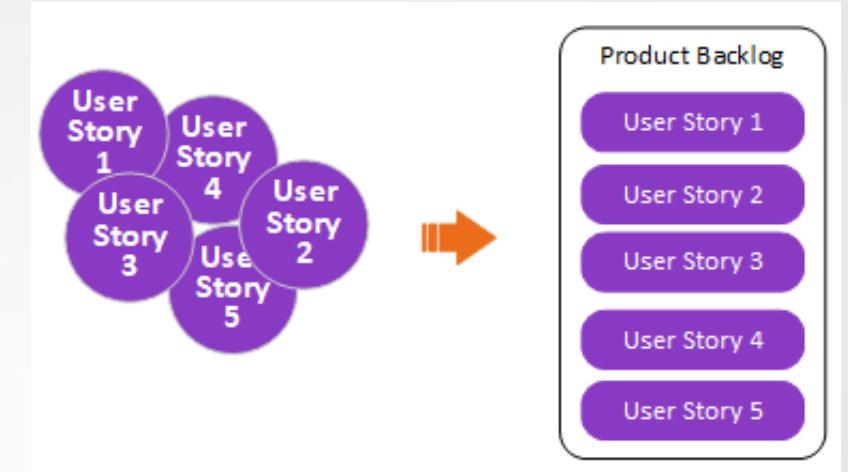
- **Stakeholders**
 - The most important role involved in Scrum
 - The reason a Team is developing a Product
- **Business Owner (BO)**
 - A special stakeholder, often the Team's sponsor or champion and controls the budget for the Team

The Product Backlog

- A Scrum Team's work is managed with a Product Backlog
 - A collection of Product Backlog Items
 - Items (User Stories) represent the stakeholders' needs and wants that add value to a product
- Team does not do anything for any Stakeholder unless it's in the backlog
- The Product Owner is responsible for prioritizing the Backlog
- Is an ordered list of everything that might be needed in the product and is the single source of requirements.

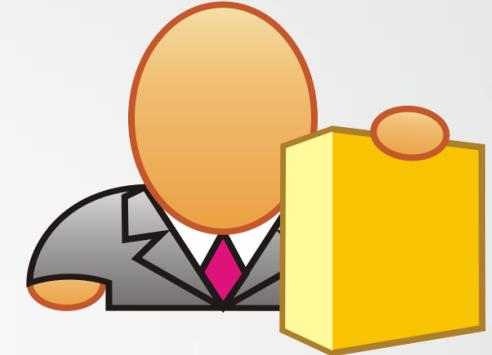


Product Backlog



- Evolves as the product and the environment in which it will be used evolves
- Is dynamic, it constantly changes to identify what the product needs, to be appropriate, competitive and useful.
- Exists as long as the product exists
- As the product is used and gains value, and the marketplace provides feedback, the Product Backlog becomes a larger and more exhaustive list

Scrum Roles: The Product Owner



- The product owner represents the user(s) and is from the business side of the organization.
- Responsible for keeping the customer and the management updated on the progress of the work.
- Works with stakeholders and team to make sure that
 - the Product Backlog items have enough details
 - The Product Backlog items align with the vision of the product
- Sole person responsible for managing the Product Backlog

Product Backlog Exercise

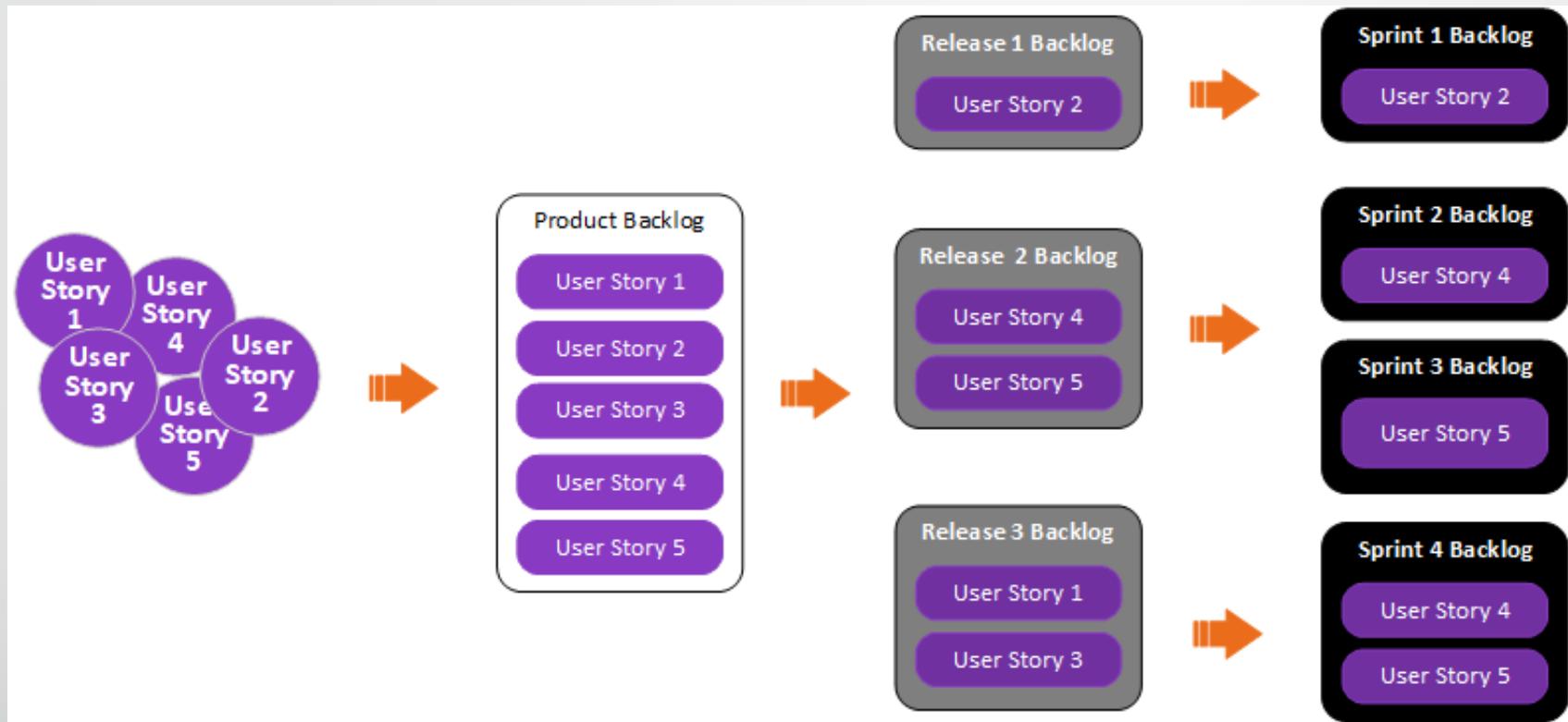
Imagine that you are tasked with creating an online software application that manages a Users To-do list of items. The following is a list of user-stories that the Product Owner and Stockholders have put together. Can you prioritize them based on what is most important?

Id	User Story	Priority
5	As an authorized User I want to edit the selected item	
9	As an authorized User I want to export a report of items and their status	
8	As an administrator I want to see the list of accounts on login	
2	As an authorized User I want to see the list of items so that I can select one	
3	As an authorized User I want to delete the selected item	
6	As an authorized User I want to set a reminder for a selected items so that I am reminded when item is due	
7	As an unauthorized User I want to create a new account	
1	As an unauthorized User I want to login	
4	As an authorized User I want to add a new item so that it appears in the list	
10	As an authorized user I want to logout	

Prioritized List of Product Backlog Items

Id	User Story	Priority
7	As an unauthorized User I want to create a new account	1
1	As an unauthorized User I want to login	2
10	As an authorized user I want to logout	3
9	As an authorized User I want to export a report of items and their status	4
2	As an authorized User I want to see the list of items so that I can select one	5
4	As an authorized User I want to add a new item so that it appears in the list	6
3	As an authorized User I want to delete the selected item	7
5	As an authorized User I want to edit the selected item	8
6	As an authorized User I want to set a reminder for a selected items so that I am reminded when item is due	9
8	As an administrator I want to see the list of accounts on login	10

Release Backlog



- With a prioritized set of user-stories, we are now ready to plan out several release-backlogs to describe sprints to get the work done.

Break the following list of items into one or two releases

Id	User Story	Priority
7	As an unauthorized User I want to create a new account	1
1	As an unauthorized User I want to login	2
10	As an authorized user I want to logout	3
9	As an authorized User I want to export a report of items and their status	4
2	As an authorized User I want to see the list of items so that I can select one	5
4	As an authorized User I want to add a new item so that it appears in the list	6
3	As an authorized User I want to delete the selected item	7
5	As an authorized User I want to edit the selected item	8
6	As an authorized User I want to set a reminder for a selected items so that I am reminded when item is due	9
8	As an administrator I want to see the list of accounts on login	10

Break the following list of items into one or two releases

Release 1	As an unauthorized User I want to create a new account	1
	As an unauthorized User I want to login	2
	As an authorized user I want to logout	3
	As an authorized User I want to export a report of items and their status	4
Release 2	As an authorized User I want to see the list of items so that I can select one	5
	As an authorized User I want to add a new item so that it appears in the list	6
	As an authorized User I want to delete the selected item	7
	As an authorized User I want to edit the selected item	8
	As an authorized User I want to set a reminder for a selected items so that I am reminded when item is due	9
Release 3	As an administrator I want to see the list of accounts on login	10

In Class Activity #12

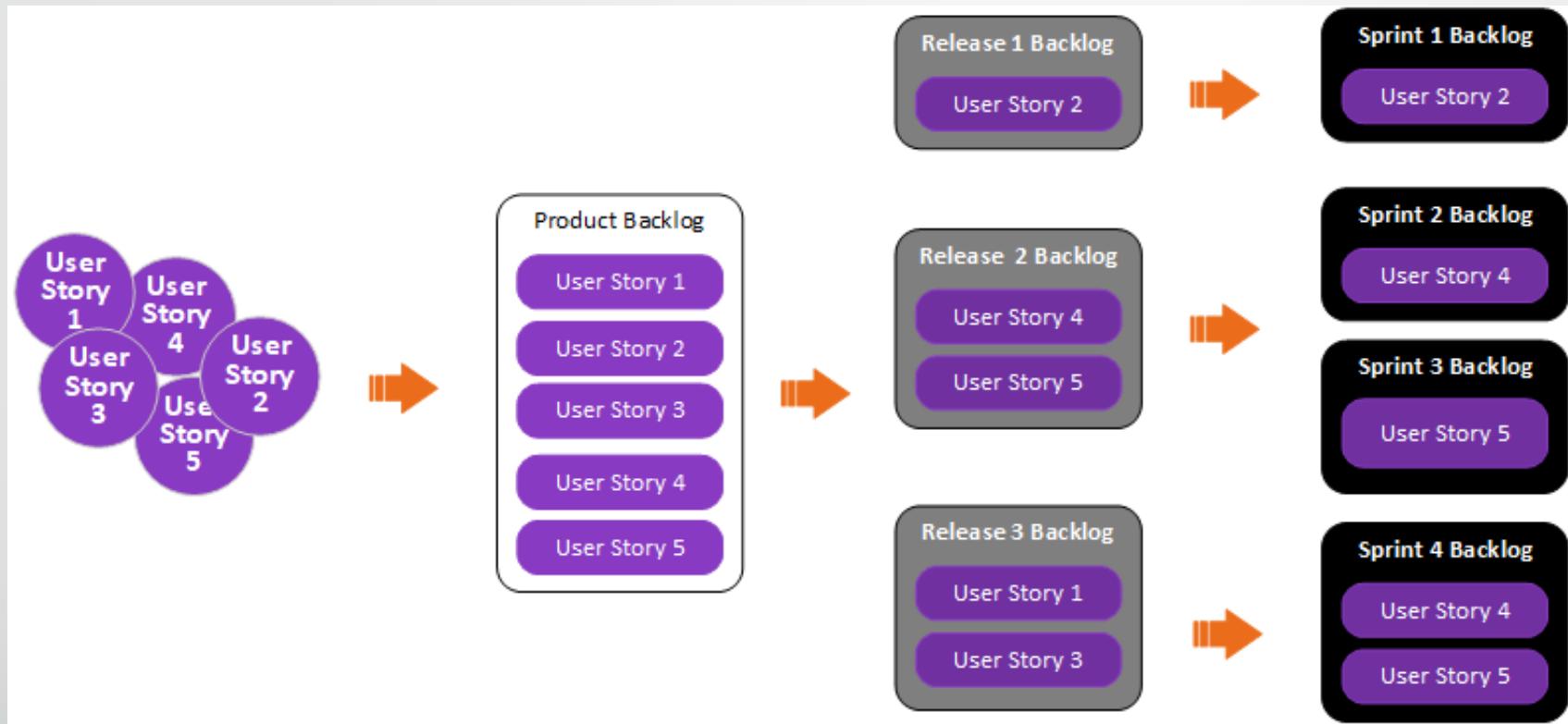
- Go back to your list of User-Stories for each of your products. Create a Product Backlog and prioritize items. You will then review this new prioritized items with your Product Owner and write down their feedback.
- Use your prioritized list of your Product Backlog Items and break them into one or more releases. For each release create a Release Backlog. You will then review each Release Backlogs with your Product Owner and write down their feedback.

BACS385: 20th Day of Class

Agenda

- Continue with Scrum
- Complete Group Activity #12

Release Backlog



- With a prioritized set of user-stories, we are now ready to plan out several release-backlogs to describe sprints to get the work done.

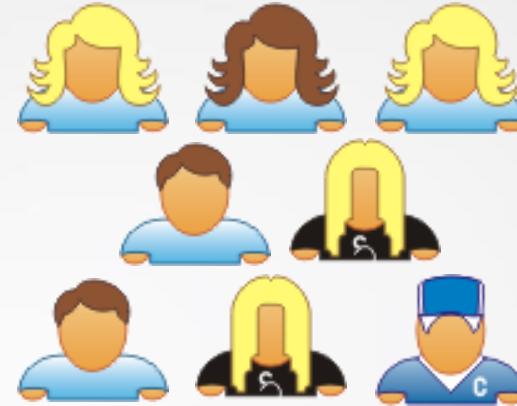
Scrum Roles: The ScrumMaster



- Represents management to the project
- Removes impediments
- Ensure that the team is fully functional and productive
- Shield the team from external interferences

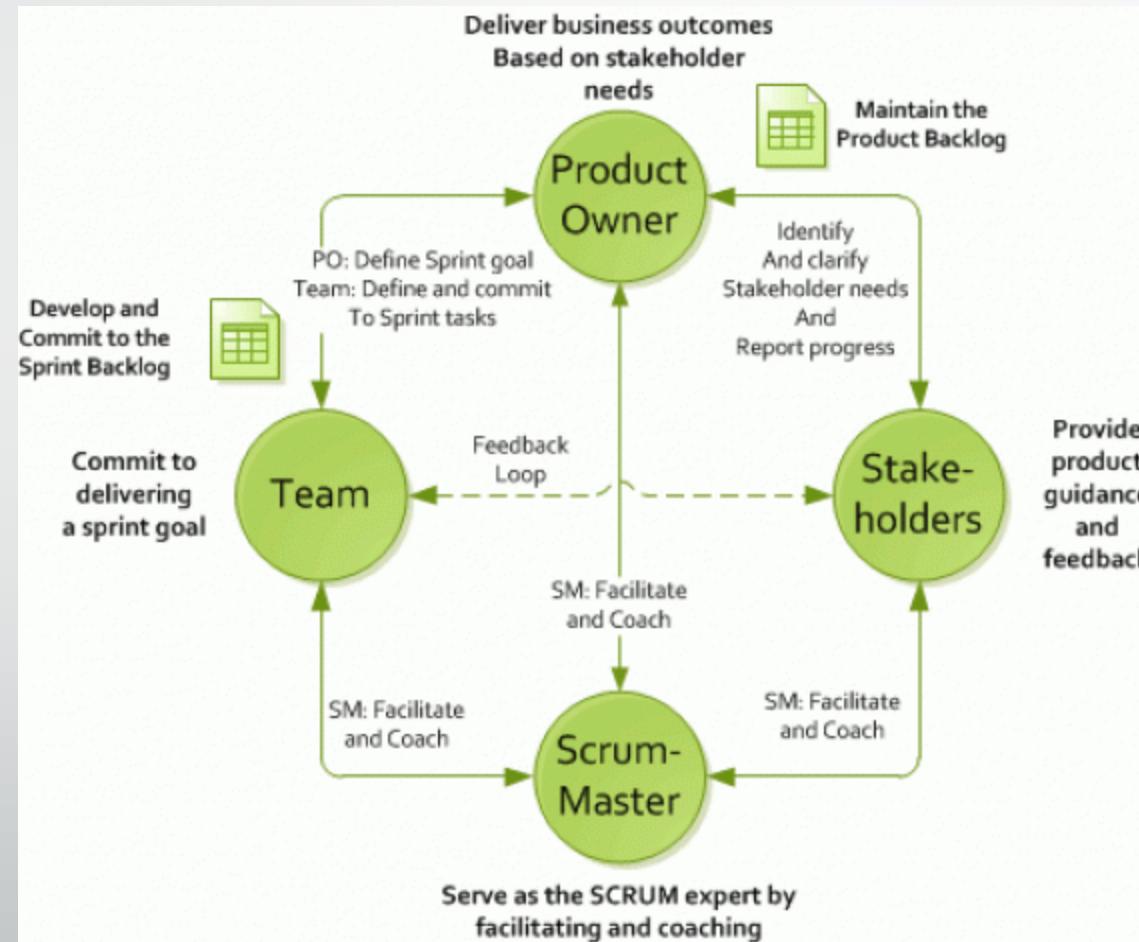
Scrum Roles:

The team

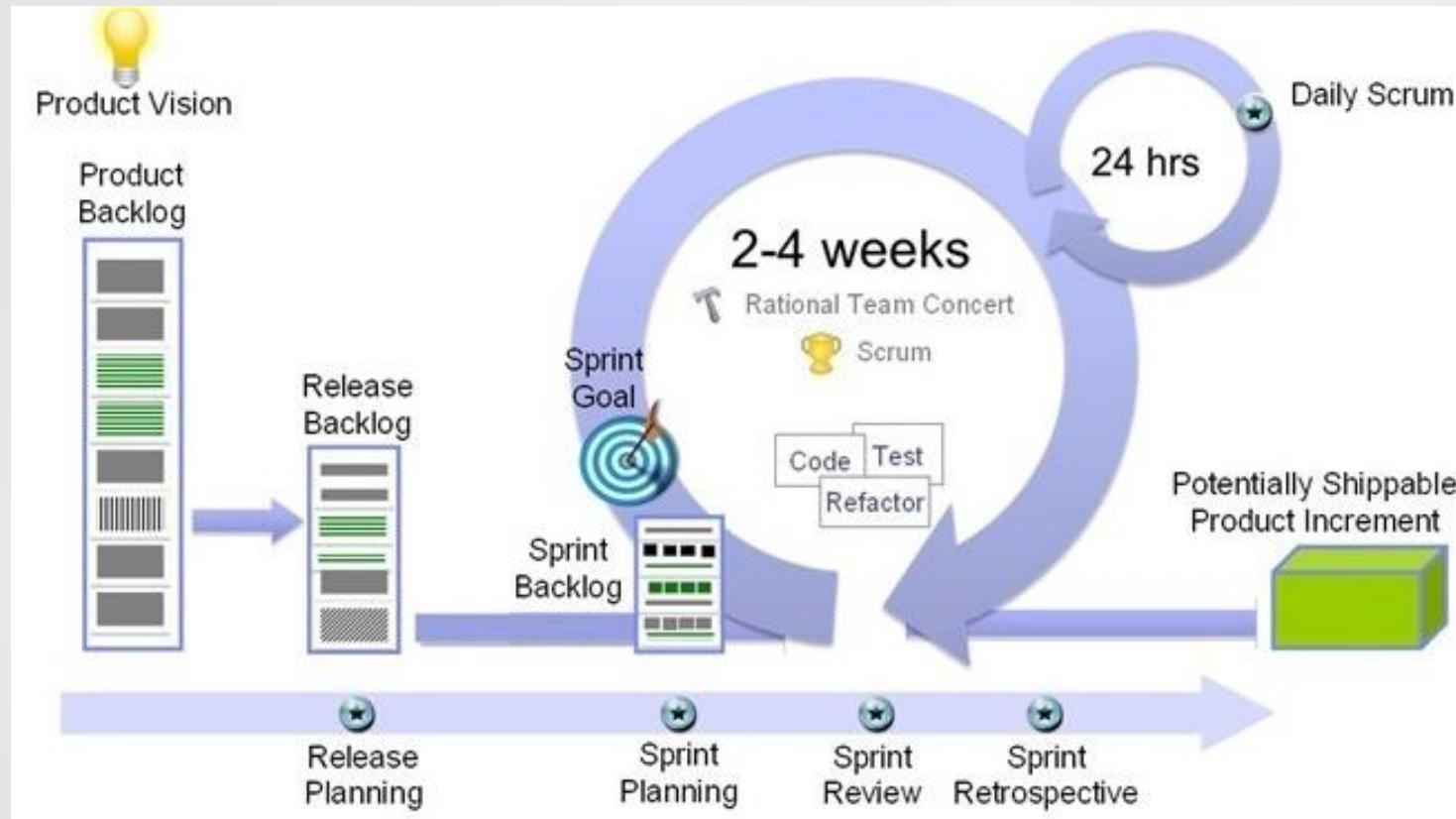


- Cross-functional:
 - Programmers, testers, user experience designers, etc.
- Members should be full-time
 - May be exceptions (DBA)
- Teams are self-organizing

The Scrum Roles Maintain a Balance of Power



Scrum Workflow



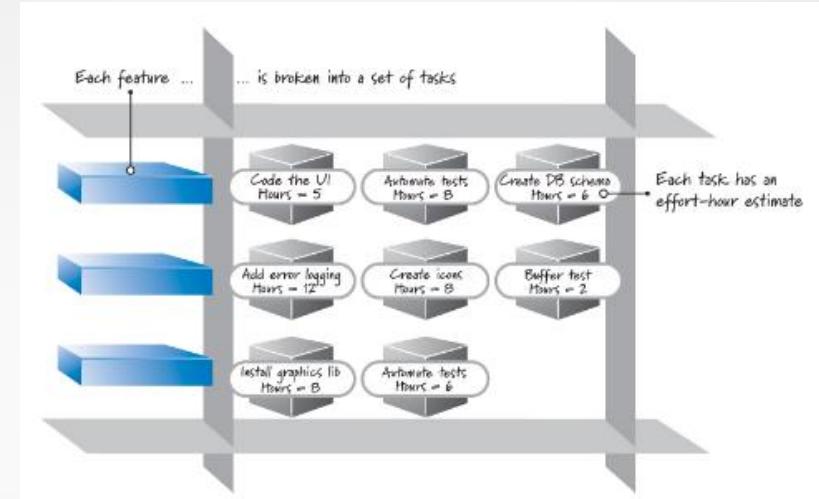
- You break down the product backlog into one or more Release Backlogs, and for a given release, you further break up the release backlog into a number of Sprint Backlogs which are essentially short duration milestones throughout your project.
- You then monitor the progress of each sprint using Daily Scrum meetings to ensure everything is on track.
- After each sprint, you have a Retrospective meeting to fine-tune everything.

Sprint Backlogs

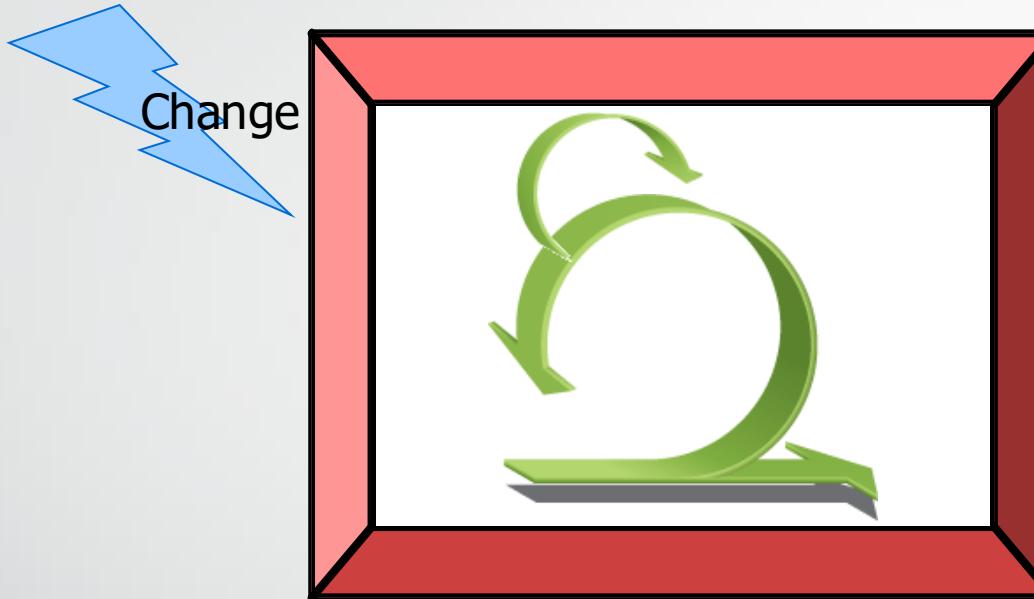
- Sprint Backlogs are a set of items selected from the Product Backlog to be completed during a Sprint, a time-boxed period in Scrum. They include the tasks, requirements, or features to be developed, along with their descriptions, and are owned and managed by the Development Team.

The Sprint

- Fundamental process flow of Scrum
- A short, fixed-length period of time
- Goal is to produce Backlog Items into renewable products that Stakeholders can provide feedback on
- Scrum Master ensures Team agrees to realistic goals



No changes during a sprint



- Plan sprint durations around how long you can commit to keeping change out of the sprint

Sprint Structure

- Sprint Planning
- Daily Scrum
 - Daily Standups
- Sprint Review
- Sprint Retrospective



Sprint Planning

- Product owner describes priorities
 - PO to work with the Team to negotiate what Backlog Items the Team will work on in order to meet Release Goals
- Team breaks product backlog items (features) into tasks
- Team commits to some amount of work

Example

Assume the following user-story was assigned to you. How would you go about completing this?

- As an unauthorized User I want to create a new account

As an unauthorized User I want to create a new account

- Create Wireframes to depict elements needed (e.g., input boxes, buttons).
 - Create account screen
 - Account creation success screen
 - Do we need to send an email? What should the email look like?
 - Think about an appropriate acceptance criteria for each wireframe
- Create Workflow diagram, showing the components required to account creation done.
 - Does account creation require approval stages?
- Create Entities that describe the data necessary to complete the user-story
 - What data is needed to describe an account?
 - Name: First/Last
 - Email address
 - Mailing address
 - Etc...

The Daily Scrum

- Essential tool to having communication flow freely between team members.
- The idea is to have fast paced stand-up meetings where team members quickly list the work they completed since the last meeting, and any obstacles in their way.
- By meeting daily, it ensures the team is always in-sync, and any major issues are dealt with as soon as they are known.

The Daily Scrum

- Parameters
 - Daily
 - 10-15 minutes
 - Stand-up
- Not for problem solving
- Helps avoid other unnecessary meetings
- Great way to manage remote teams
 - Prevents teams from wasting time

Everyone answers 3 Qs

What did you do yesterday? **1**

What will you do today? **2**

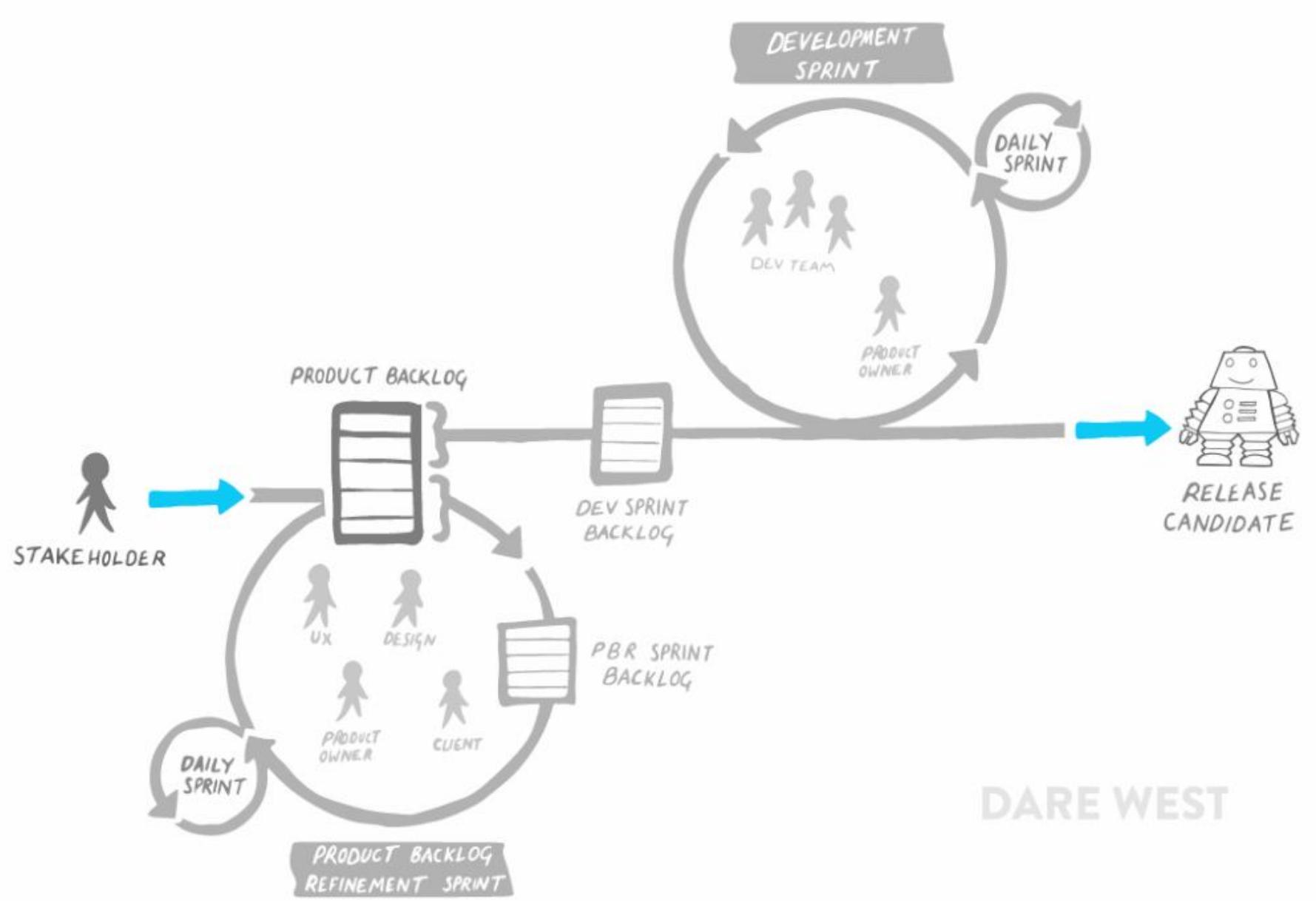
Is anything in your way? **3**

Sprint Review

- Team demonstrates what was accomplished. Demo the product to the team.
- 2-hour prep time rule
- No PowerPoints!
- Anyone can attend

Sprint Retrospective

- Focus on continuous improvement
- Look back at the last sprint and ask what would we like to?
 - Start doing
 - Stop doing
 - Continue doing





Complete Group Activity #12