Name - Divyansh - Mark _____/50

[**Instructions**: Remove everything that is not a heading below and fill in with your own diagrams, etc.]

1. Brief introduction __/3++

I'll be adding dynamic character animations to our top-down shooter game, ensuring that every player action, like walking and attacking, results in a smooth, context-appropriate animation. If an asset is missing, the system will safely fall back to the default animation.

2. Use case diagram with scenario __14

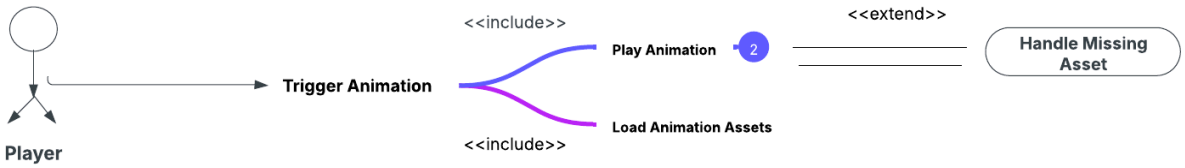
[Use the lecture notes in class.

Ensure you have at least one exception case, and that the <<extend>> matches up with the Exceptions in your scenario, and the Exception step matches your Basic Sequence step.

Also include an <<include>> that is a suitable candidate for dynamic binding]

Example:

Use Case Diagrams



Scenarios

[You will need a scenario for each use case]

Name: Trigger Animation

Summary: When a player's action is detected (for example, moving or attacking), the game must display the corresponding animation for the character.

Actors: Game Engine, Player Input

Preconditions:

All primary animation assets (idle, walking, attacking) have been preloaded into the asset repository.
The character is active in the game scene.

Basic Sequence:
Step 1: The player performs an action (e.g., presses a movement key or initiates an attack).

Step 2: The Game Engine captures this input and sends a trigger event to the Animation Controller.

Step 3: The Animation Controller determines the correct animation (e.g., walking for movement).

Step 4: The Animation Controller calls the “Load Animation Assets” use case (<<include>>) to retrieve the appropriate asset.

Step 5: The Asset Loader returns the animation asset to the Animation Controller.

Step 6: The Animation Controller instructs the Renderer to play the animation (Play Animation use case).

Step 7: The correct animation is displayed on the screen.

Exceptions:

Exception Case:

Condition: The Asset Loader fails to find the requested animation asset (like, the attacking animation is missing).

Exception Step: In this situation, the Animation Controller immediately triggers the “Handle Missing Asset” use case (<<extend>>), which loads a default idle animation instead.

Outcome: A default idle animation is displayed, and an error is logged.

Post Conditions:

The character’s animation accurately reflects the player’s input or game event.

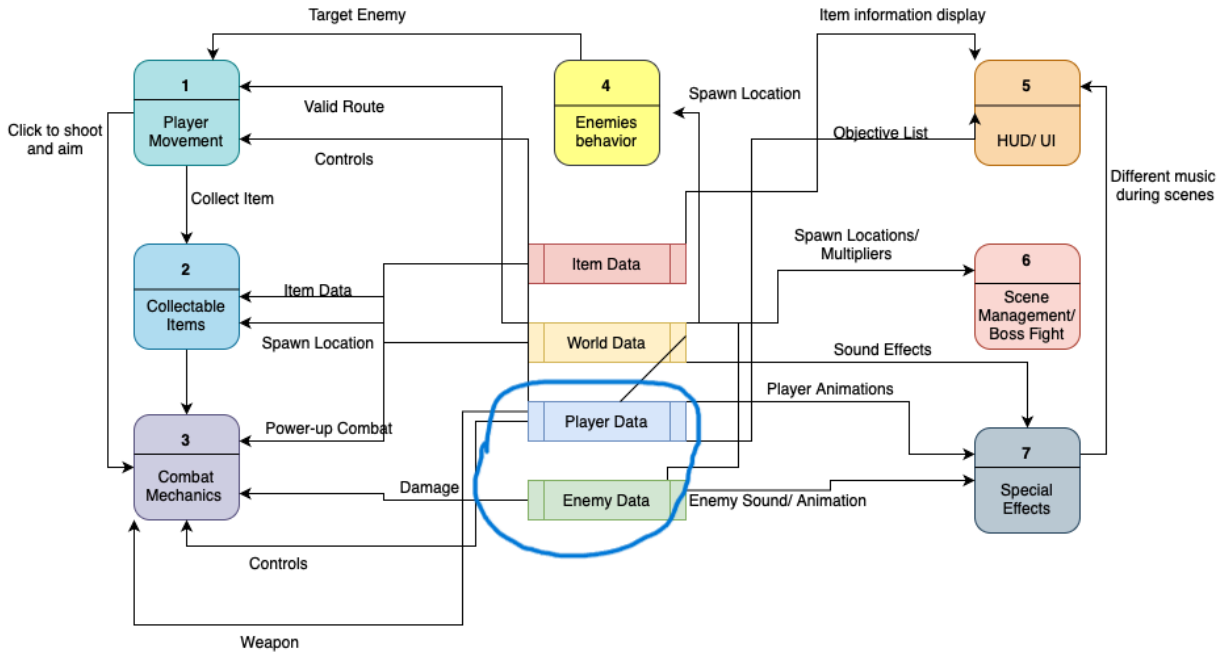
Priority: 2*
ID: C01
*The priorities are 1 = must have, 2 = essential, 3 = nice to have.

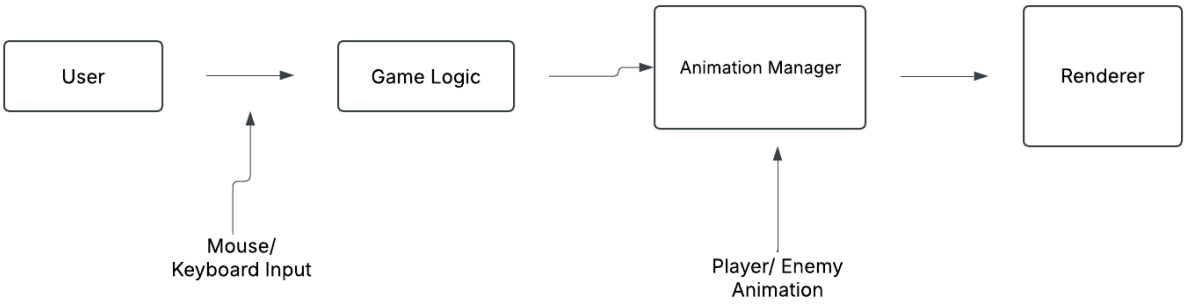
3. Data Flow diagram(s) from Level 0 to process description for your feature _____14

[Get the Level 0 from your team. Highlight the path to your feature]

Example:

Data Flow Diagrams





Process Descriptions

Process Name: Animation Manager

Purpose: Manage and display character animations based on game events.
Inputs:

Animation trigger events from Game Logic
Player input signals (e.g., movement, attack)

Steps:

Receive Event: The Animation Manager receives an animation trigger from the Game Logic.

Determine Animation: It evaluates the current game state to decide whether to use idle, walking, or attacking animation.

Load Asset:
Calls the Asset Loader to retrieve the required animation asset (this call is implemented as an <<include>>).

Asset Verification:
If the asset is successfully retrieved, proceed to Step 5.
If the asset is missing, trigger the “Handle Missing Asset” routine (<<extend>>) which loads a default animation.

Render Animation: The retrieved asset is sent to the Renderer, which displays the animation on screen.

Log Transition: Any errors or asset retrieval issues are logged for debugging.

Outputs:

Animation asset data sent to the Renderer
Logs indicating successful or failed asset retrieval

4. Acceptance Tests _____9

Test Case 1: Valid Idle Animation

Input: No player input (character remains stationary).

Expected Output: Idle animation is displayed continuously.

Test Case: Valid Walking Animation

Input: Player presses movement keys.

Expected Output: Walking animation is displayed and transitions smoothly as long as the input is active.

Test Case 2: Valid Attacking Animation

Input: Player initiates an attack command.

Expected Output: Attacking animation plays for the duration of the attack action.

Test Case 3: Missing Animation Asset

Input: Player initiates an attack while the attacking animation asset is missing/corrupted.

Expected Output: The system automatically loads and displays the default idle animation; an error is logged.

Test Case 4: Rapid State Change

Input: Player rapidly alternates between walking and attacking.

Expected Output: Animations transition smoothly without lag or display glitches.

Boundary Cases:

Provide tests with null or invalid inputs to ensure the system always falls back to a safe (idle) animation without crashing.

5. Timeline _____/10

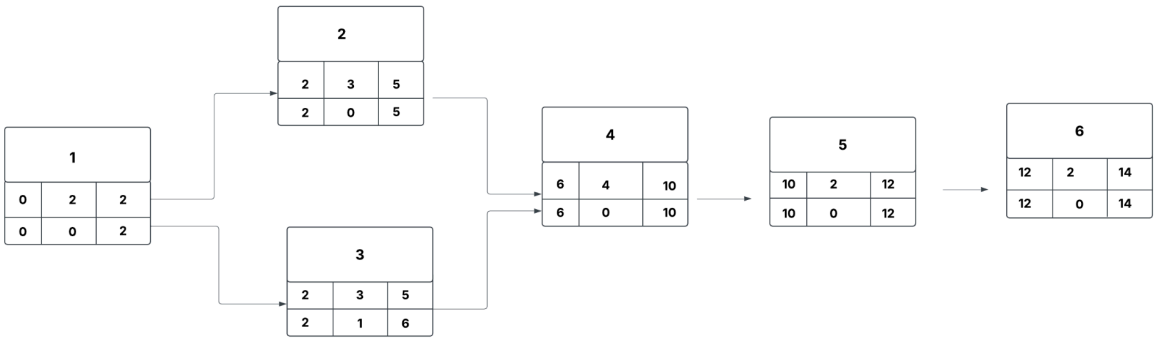
[Figure out the tasks required to complete your feature]

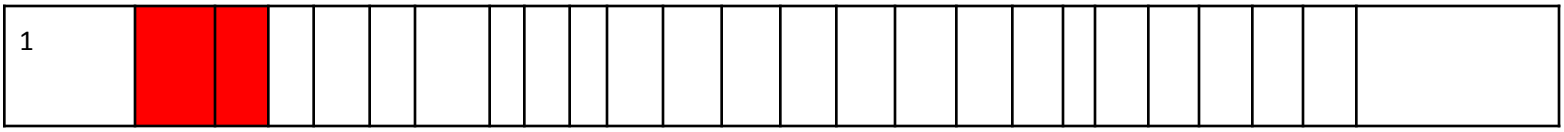
Example:

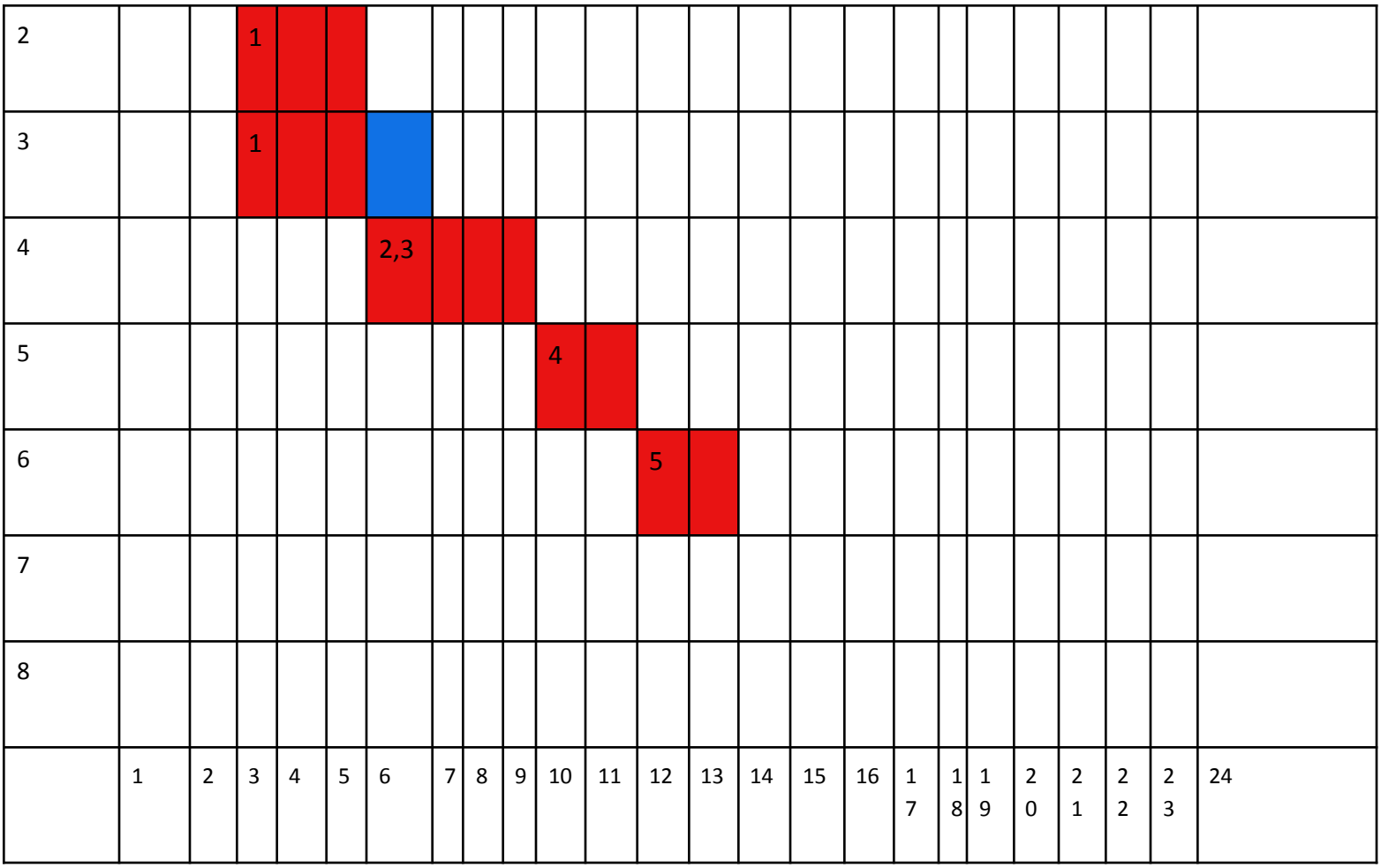
Work items

Task	Duration (PWks)	Predecessor Task(s)
1. Requirements Collection & Analysis	2	-
2. Animation Design & Storyboarding	3	1
3. Develop Animation Manager (Coding & Asset Loader)	3	1
4. Integration with game logic (integrating events)	4	2, 3
5. Testing & Debugging	2	4
6. Final Deployment & Documentation	2	5

Pert diagram







Red - Weeks

Blue - Slack