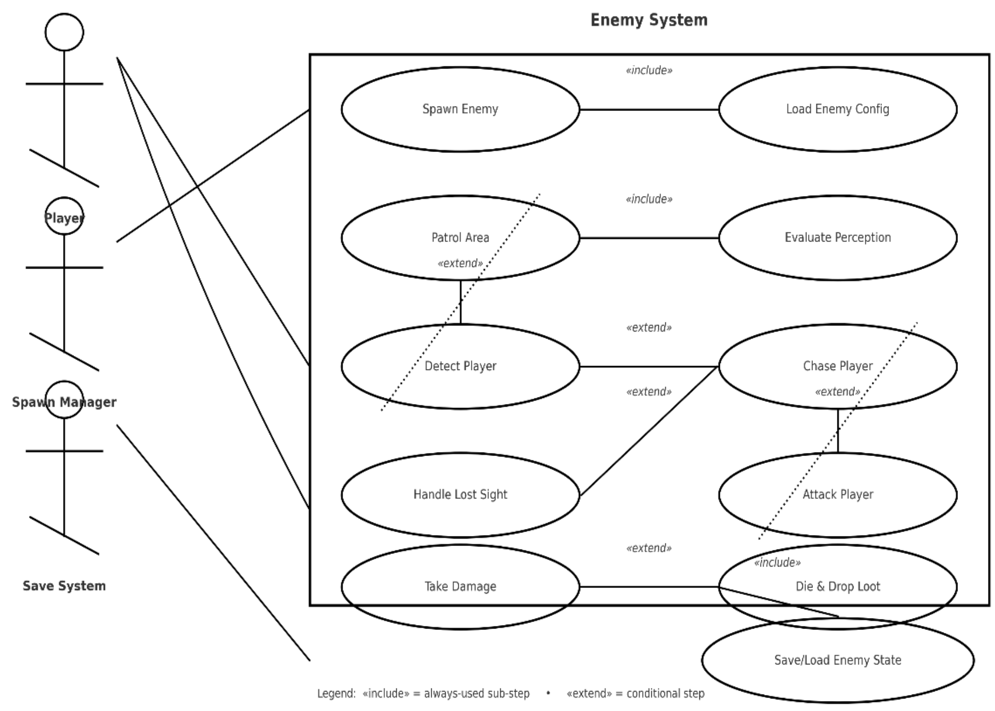
Name: Anjal Karki Mark ______________________/50

1. Brief introduction __/3

Where's My Spaceship? is a 2D platformer that jumps across **Prehistoric, Medieval, and Cyberpunk** eras.
My role is to build the enemies for each era and make them feel unique yet fair. I will create a reusable enemy system with clear behaviors—**patrol**, **detect**, **chase**, **attack**, **take damage**, and **drop items**—so encounters are readable and fun. The system is **data-driven** (tuned from simple config files), connects to level spawners and the HUD, and supports saving/loading of enemy state. The goal is evolving enemy variety that keeps players learning and adapting as they progress through each era.

2. Use case diagram with scenario __14

Use Case Diagrams



Scenarios

Name: Detect & Chase

Summary: Enemy notices the player and starts chasing until in attack range or the player is lost.
Actors: Player, Enemy System
Preconditions: Enemy is active and patrolling; game running.
Basic sequence:

Step 1: Enemy patrols its route.
Step 2: Perception check runs (vision/hearing/LOS). *(<include> Evaluate Perception)*
Step 3: If the player is detected, switch state to **Chase Player**. *(<extend> Detect Player)*
Step 4: Enemy moves toward player using pathfinding.
Step 5: If within attack distance, hand off to **Attack Player**.
Exceptions:
Step 3: Player breaks line-of-sight for N seconds → **Handle Lost Sight** and return to **Patrol Area**.
Step 4: Path blocked → choose alternate path; if none, return to **Patrol Area**.
Post conditions: Enemy is either attacking or back on patrol.
Priority: 1*
ID: UC-EN-01

Name: Attack Player

Summary: Enemy performs a clear, readable attack when close enough.
Actors: Player, Enemy System
Preconditions: Enemy in **Chase Player**; player within attack distance.
Basic sequence:
Step 1: Enemy telegraphs the attack (wind-up).
Step 2: Attack executes; hitbox is active briefly. *(<extend> from Chase Player)*
Step 3: If it connects, apply damage and show HUD feedback.
Step 4: Enemy enters short recovery, then re-evaluates distance (back to **Chase Player** or repeat).
Exceptions:
Step 2: Player dodges/blocks → enemy enters **Recover/Stagger** and returns to **Chase Player**.
Step 1–2: Player moves out of range mid-windup → cancel and return to **Chase Player**.
Post conditions: Attack finishes; enemy either chases again or resets.
Priority: 1*
ID: UC-EN-02

Name: Die & Drop Loot

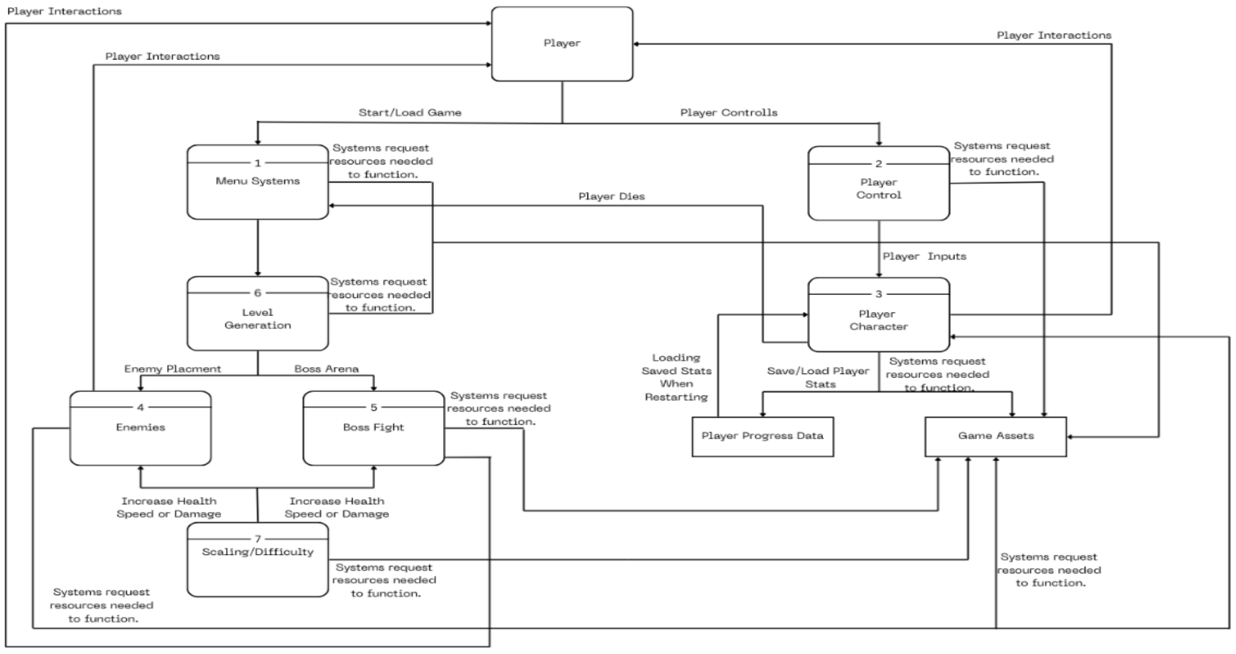
Summary: Enemy is defeated and the game updates world state.
Actors: Player, Enemy System, Save System
Preconditions: Enemy HP reaches zero.
Basic sequence:
Step 1: Play death animation; disable collisions.
Step 2: Roll and spawn loot (if any); notify HUD/inventory.
Step 3: **Save System** records that this enemy is defeated. *(Save/Load Enemy State)*
Step 4: Despawn after a short delay.
Exceptions:
Step 2: Inventory full → hold item on ground as a pickup.
Step 3: Save write fails → mark retry flag for next checkpoint.
Post conditions: Enemy removed; loot available; defeat state saved.

Priority: 2*
ID: UC-EN-03

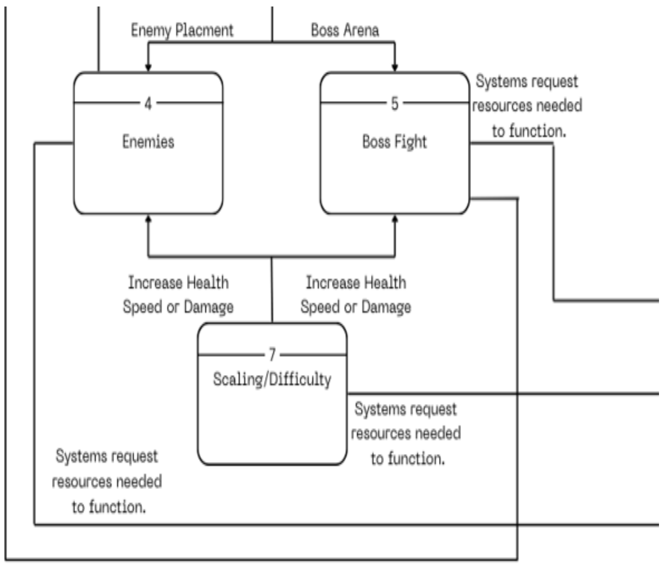
*Priorities: **1 = must have**, **2 = essential**, **3 = nice to have**.

3. Data Flow diagram(s) from Level 0 to process description for your feature ______14

Data Flow Diagrams



Player Interactions
Player
Player Interactions
Player Interactions
Start/Load Game
Player Controlls
1
Menu Systems
Systems request resources needed to function.
2
Player Control
Systems request resources needed to function.
Player Dies
Player Inputs
6
Level Generation
Systems request resources needed to function.
3
Player Character
Loading Saved Stats When Restarting
Save/Load Player Stats
Systems request resources needed to function.
Enemy Placment
Boss Arena
4
Enemies
5
Boss Fight
Systems request resources needed to function.
Player Progress Data
Game Assets
Increase Health Speed or Damage
Increase Health Speed or Damage
7
Scaling/Difficulty
Systems request resources needed to function.
Systems request resources needed to function.
Systems request resources needed to function.



Enemy Placment
Boss Arena
4
Enemies
5
Boss Fight
Systems request resources needed to function.
Increase Health Speed or Damage
Increase Health Speed or Damage
7
Scaling/Difficulty
Systems request resources needed to function.
Systems request resources needed to function.

Process Descriptions

P1. Enemy Placement

- Get spawn list (type, count, positions) from Level Generation.
- Load matching EnemyConfig; instantiate prefab at each point.
- Assign waypoints; set initial state = **Patrol**.

P2. Perception

- Check vision/hearing/line-of-sight each tick.
- Output: detected? and lastKnownPos to AI.

P3. Enemy AI Controller *(main path)*

- **Patrol** and call Perception every tick.
- If detected? → **Chase** toward lastKnownPos.
- If close enough and attack ready → **Attack**.
- If lost sight for T seconds → back to **Patrol**.

P4. Combat Resolver

- On **Attack**, enable hitbox briefly.
- If overlap with player → apply damage + HUD feedback.
- End attack; start cooldown.

P5. Despawn

- On HP ≤ 0: play death, disable collisions.
- Remove from AI update; deactivate/destroy after delay.

.

4. Acceptance Tests ______9

Table 1 — Patrol / Detect / Chase

Test ID	**Input (Player / Scene Setup)**	**Expected Output**	**Notes**
T01	**Player at 6m, inside FOV, clear LOS**	**Enemy Detects within ≤ 0.2s and switches to Chase**	**Boundary: baseline detection works**
T02	**Player at 6m, behind wall (LOS blocked)**	**Enemy does not Detect for full 3s window**	**Guards against false positives**

Test ID	Input (Player / Scene Setup)	Expected Output	Notes
T03	Start at 8m, flat ground; after Detect, time the approach	Enemy reaches attack range ≤ 3.0s	Normal chase responsiveness
T04	During Chase, player hides behind wall for T seconds	Enemy returns to Patrol at T ± 0.2s	Lost-sight timeout behavior
T05	Path blocked by obstacle; no alternate path	Enemy abandons chase and returns to Patrol	Exception handling for pathing

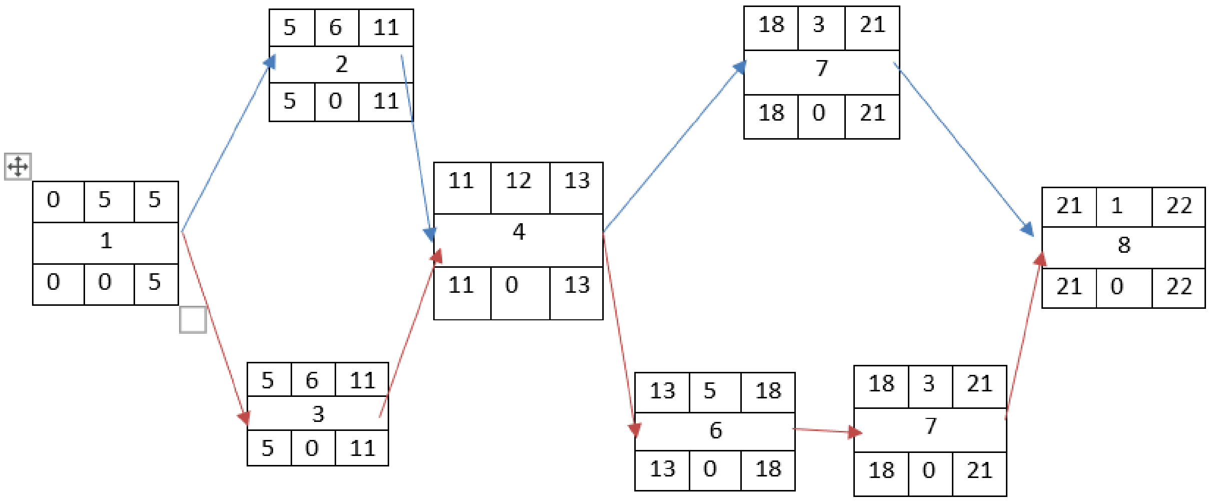
Table 2 — Attack / Damage / Death

Test ID	Input (Player / Scene Setup)	Expected Output	Notes
T06	In range; attack cooldown ready	Enemy telegraphs then attacks; ≤ 1 damage event produced	Prevents double-hit bug
T07	Player dodges during telegraph window	No damage applied; enemy enters Recovery then resumes Chase	Dodge window respected
T08	Apply 3 spaced hits to enemy (no i-frame overlap)	Enemy HP decreases once per hit	Damage timing correctness
T09	Reduce HP to exactly 0 with final hit	Death triggers once, collider off, despawn ≤ 2.0s after anim	Clean removal; no double death

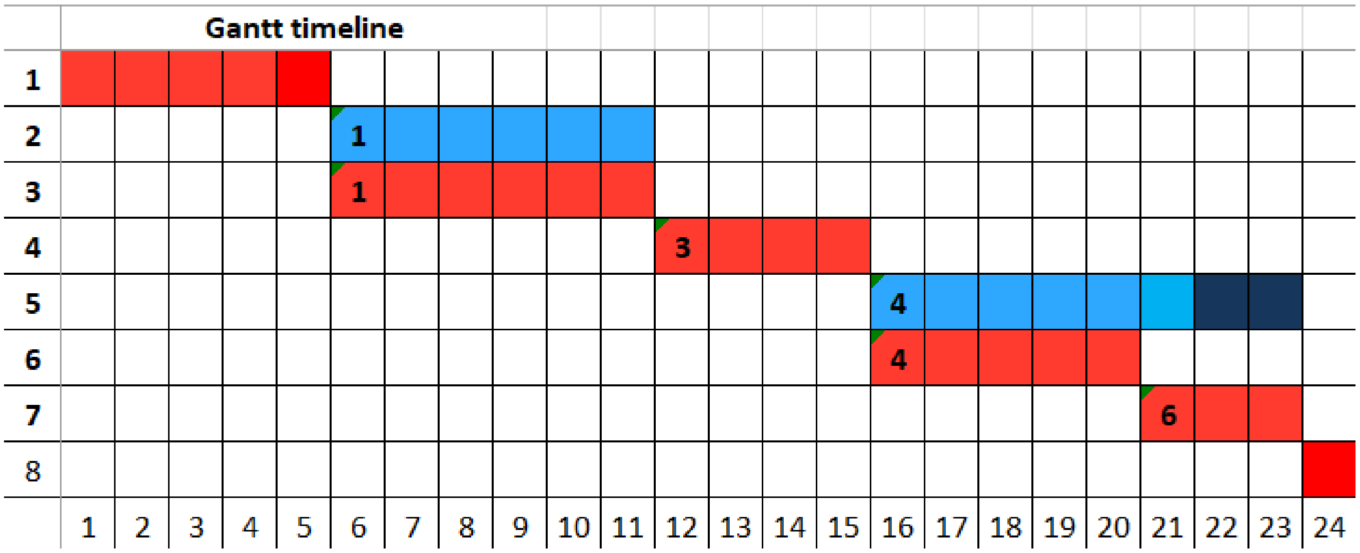
5. Timeline _______/10 Work items

Task	Duration (PWks)	Predecessor Task(s)
1. Requirements Collection	5	-
2. Scaling Algorithm Design	3	1
3. Level Balancing Rules	3	1
4. Database Construction	2	2, 3
5. UI Update	2	4
6. Programming	4	4
7. Testing	3	6
8. Integration with Game build	2	7

Pert diagram



Gantt timeline



	Gantt timeline																							
1	🟥	🟥	🟥	🟥	🟥																			
2						🟦 **1**	🟦	🟦	🟦	🟦	🟦													
3						🟥 **1**	🟥	🟥	🟥	🟥	🟥													
4												🟥 **3**	🟥	🟥	🟥									
5																🟦 **4**	🟦	🟦	🟦	🟦	🟦	⬛	⬛	
6																🟥 **4**	🟥	🟥	🟥	🟥				
7																					🟥 **6**	🟥	🟥	
8																								🟥
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24