Tower Leveling Clash

1. Project Overview

Tower Leveling Clash is a PVE (Player vs. Environment) turn-based card game where players ascend a multi-floor tower, battling enemies that grow stronger with each floor. The game follows a roguelike elimination rule: if a character dies, the game is over for that run.

Mechanics & Functionality:
Floor-based progression: Each floor contains 3 enemies with increasing strength.
Turn-based combat: The player and enemy take turns using cards to attack, defend, or use special effects.
Shop & Economy System: Players earn 15 coins per floor to buy items. Shop has 5 items if it was bought from a player it will generate a new one to complete 5 items.
Inventory System: Players maintain a lot of items and choose one to use with character for battle each enemy.
Victory Condition: The game ends when the player either clears all floors or loses their character.

2. Project Review
This project was inspired by Mizuno duel (Discord bot), a turn-based card battle game implemented as a Discord bot. In Mizuno Duel, players engage in PvP battles using text-based commands, and they can choose from three characters to play. Additionally, it features a shop system where players can purchase new characters if one is lost.

Key Enhancements in Tower Leveling Clash

1) Graphical User Interface (GUI) Instead of Text-Based Commands

- Mizuno Duel operates entirely through Discord text commands, which can make gameplay slower and harder to visualize.
- Tower Leveling Clash will use Pygame to provide a graphical UI, making characters more realistic, gameplay easier to understand, and interactions more intuitive.

2) Single Character Selection & Permanent Death Mechanic

- In Mizuno Duel, the player has access to three characters and can switch between them during battles.
- In Tower Leveling Clash, players select only one character and play until they clear all floors.
- If the character's health reaches 0, they are permanently lost, and the game is over for that run.
- This adds a higher level of strategy and risk management compared to Mizuno Duel.

3) Expanded Shop System with Items

- In Mizuno Duel, the shop only sells new characters, allowing players to replace lost characters.
- In Tower Leveling Clash, the shop introduces a more dynamic and strategic system, offering various items instead of characters, making economy management a key part of the gameplay. Including: Power-ups that enhance characters (e.g., health boosts, attack buffs).Consumable items that can be used in battle.This adds more depth to the economy system, giving players strategic choices between saving coins, or purchasing useful items.

3. Programming Development

3.1 Game Concept

Game flow:

Start Game → The player selects a character.
Battle Phase → The player battles one of three enemies on the current floor.
If the player wins, they move on to the next enemy or advance to the next floor.
Earn 15 coins per floor → Players can spend them in the shop for new items.
The cycle continues until the player clears all floors or their character dies.

Selling Point:
Roguelike Gameplay → Each run is unique due to randomized enemies and shop items.
Strategic Turn-Based Combat → Players must choose attacks, use items wisely, and adapt to enemy strategies.

3.2 Object-Oriented Programming Implementation

Key OOP improvements:
Separate Character, Inventory, Item, Shop, Save, GamePlay, and GamePlayUI classes.
Improved game state management.
Easier future expansions (e.g., adding new enemies or shop items without modifying core logic).

Class Shop:

- Attributes: item_in_shop, money
- Method: reroll(), check_money()

Store 5 item in shop and when user buy it, the shop should generate the new one

Class Character:

- Attributes: image, name, health, attack, defense, special_ability, level
- Method: attack_enemy(), use_ability(), take_damage(), level_up()

Store all detail for character in game and movement in character.

Class Item:

- Attributes: image, name, cost, ability

Store all details for items in game.

Class GamePlay:

- Attributes:clock,running,state,selected_character,enemies, current_turn, enemy_index, battle_log, inventory
- Methods:run_game_loop(),progress_to_next_floor(),check_game_over(), save_data()

Manages the main game loop, controls progression, and handles win/loss conditions.Save data per round into a GameStats class. Easy to add more save functionality later (e.g., inventory, achievements)

Class Inventory:

- Attributes: coin,items,max_slot
- Methods:add_coin(),add_item(), use_item(), discard_item()

Store all items and coins for user to use

Class GamePlayUI:

- Attributes: screen, characters, character_buttons
- Methods: draw_home_screen(),draw_selected_character_screen(), draw_battle_screen(),draw_game_over(),draw_game_victory(), draw_shop_screen(), draw_inventory_screen(), draw_floor()

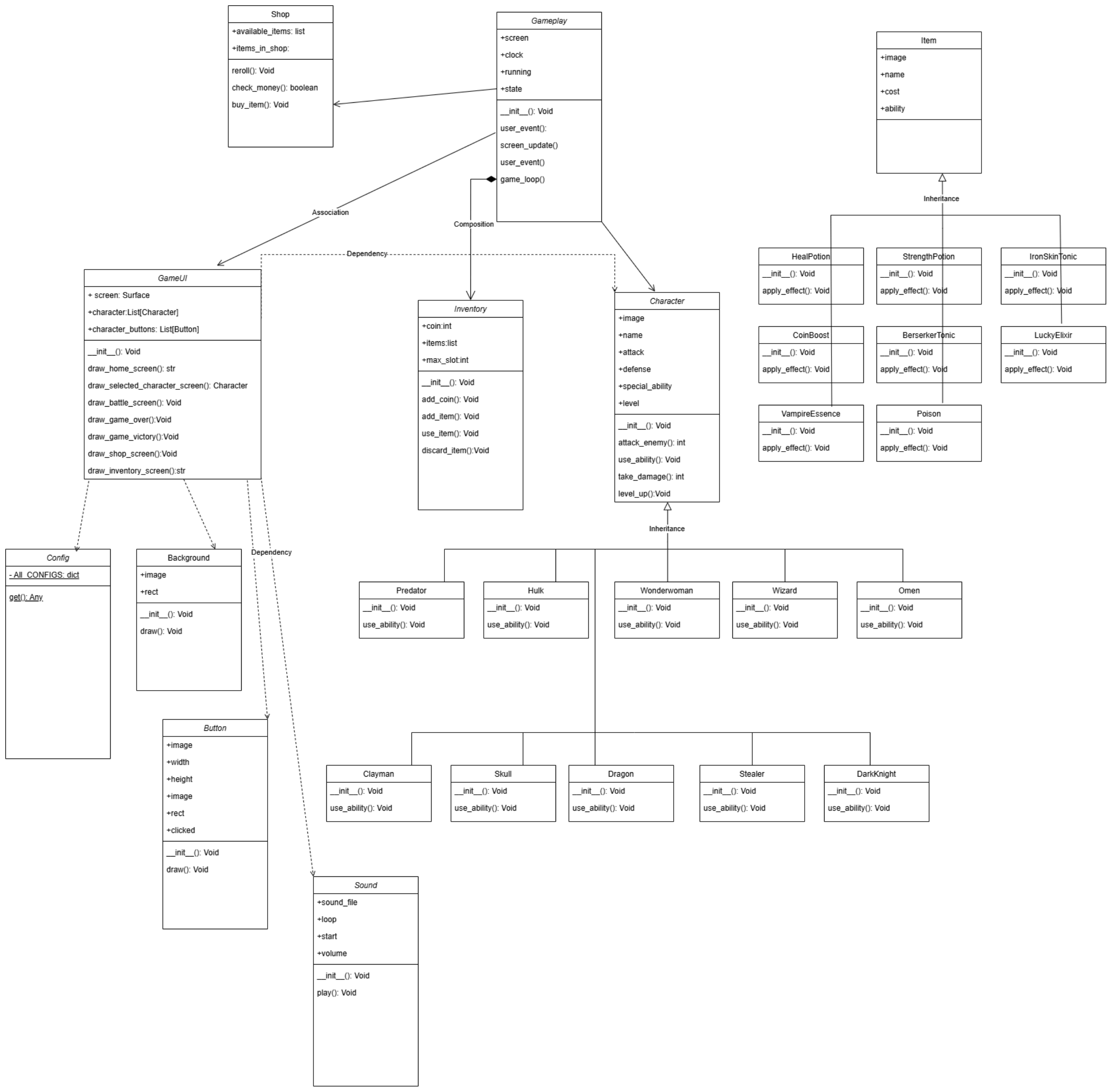
Display the window for user to see graphic and result

Class GameStats:

- Attributes: total_floor, total_damage_dealt, total_win, total_earned, total_spend, start_time, total_time
- Methods: start_time(), stop_time(), record_floor_clear(), record_damage(), record_win(), record_earn(), record_spent(), summary()

Collect gameplay data then write that data into a CSV file for analysis or visualization later

Current UML Diagram (Subject to Change)



This UML diagram will be updated after the final implementation of the code.

3.3 Algorithms Involved

- Turn-Based Strategy Algorithm → Controls when the player and enemy take turns, deciding which action to take.
- Enemy Scaling Algorithm → As the player progresses, enemies gain more health and attack power.
- Shop Reroll Algorithm → Uses randomization items to fill the empty slot shop items because the shop must always fix with 5 items.
- Enemy Selection Algorithm → Uses randomization to select 3 enemies for the next floor.

4. Statistical Data (Prop Stats)

4.1 Data Features

1) Total Floors Cleared : How far the player progressed before losing.

   Table Field: play_id, start_game_time, game_over_time, kill_number, floor_number, character

2) Damage Dealt Per Game : Tracks attack points each floor per game session.

   Table Fields: id, play_id, floor , attack_point

3) Total earn : Track all coins that earn per round and per floor.

   Table Field: play_id, floor, coin_earned

4) Total spend: Tracks coins spent and what items were bought.

   Table Field: play_id, item_bought, floor_num, coin_spent

5) Total time : Tracks time spent on each floor.

   Table Field: play_id, floor_num, start_time, end_time

4.2 Data Recording Method

It will save a record of each round in a csv file. Each feature will have its own CSV table, enabling clearer processing and visualization later.

4.3 Data Analysis Report

Statistical measures:

- Mean (Average): Average number of floors cleared.
- Median: Median damage dealt per game.
- Mean (Average): Average time per floor (total floor/ total time)

Using graphs to analyze the distribution of floors cleared, damage per round, average time per floor

Feature	Why it is good to have this data ? What can it be used for	How will you obtain 50 values of this feature data	Which variable (and which class will you collect this from?)	How will you display this feature data (via summarization statistics or vis graph)
Total Floors Cleared	Shows how far players get before failing, useful for game balancing	Run the game 50 rounds and record floors reached	total_floors in GameStats	Statistics for play summary and a bar graph
Damage Dealt Per Game	Measures how effective players are in combat, useful to analyze power balance	Record total_damage_dealt per session, repeat 50 times	total_damage_dealt in GameStats	Average line graph
Total Earn	Helps balance the coin economy	total_time in GameStats	total_earned in GameStats	Bar chart
Total Spend	Tracks how much players spend, helpful for pricing items	Sum total_spent per game over 50 runs	total_spent in GameStats	Bar chart or average coins spent
Total Time	Measures time spent per game	Use timer (start/stop), collect for 50 rounds	total_time in GameStats	Line graph

Feature to show in a table:

- Total Floors Cleared

Statistical values shown in table:
Example

Character	Average Kills	Max Floor Reached	Average Floor	Total game
Wizard	9.8	16	10.1	10

Graph:

	Feature Name	Graph objective	Graph type	X-axis	Y-axis
Graph1	Total Floors Cleared	Show avg kill count per character	Bar Graph	Character Name	Avg Kills

Graph2	Total Floors Cleared	Show % of characters reaching floor N	Pie Chart	Character Name	% Reach Floor N
Graph3	Character Usage	Show which character is picked most often	Pie Chart	Character Name	Usage Frequency
Graph4	Damage Dealt	Track avg attack per floor	Line Graph	Floor Number	Avg Attack Points
Graph5	Total Earn	Show money earned per floor per player	Bar Graph	Floor Number	Coins Earned
Graph6	Total Spend	Show item spending distributio n	Bar Graph	Item Name	Coins Spent
Graph7	Total Time	Tracks time spent on each floor.	Line Graph	Floor Number	Time spent

5. Project Timeline

Week	Planned Task
1 (10 March)	Submit Proposal #1 and initiate the project
2 (17 March)	Revise proposal format and begin implementing core gameplay (class structure, mechanics)
3 (24 March)	Continue gameplay implementation (combat system, turn mechanics, base classes)
4 (31 March)	Finalize core gameplay features; refine character mechanics and game logic
5 (7 April)	Start developing and integrating the graphical user interface (GUI)
6 (16 April)	Continue GUI development and interface design
7 (17-23 April)	Debug and test game functionality; begin storing gameplay data into CSV files
8 (24- April-11 May)	- Implement statistical graphs for Probability & Statistics analysis - Final debugging and testing - Prepare final files for submission
9 (11 May)	Final Submission

Tasks expected to be completed by 16 April (50%)

- Finalized game concept
- Implemented core class structures (Player, Enemy, GameStats, etc.)
- Developed main gameplay mechanics (turn-based combat, enemy generation, player actions)

Tasks expected to be completed by 23 April (75%)

- GUI fully implemented (inventory, shop, game over/victory screens)
- Game data (floors cleared, damage, coins, time) successfully recorded into CSV files

Remaining 25% of the tasks to be completed by 11 May

- Implement and visualize statistical graphs (histogram, pie chart, line graph)
- Refine and clean up all game code and user interface
- Final debugging
- Final report preparation and documentation

6. Document version

Version: 4.0
Date: 31 March 2025

Date	Name	Description of Revision, Feedback, Comments
15/3	Phiranath	Don’t forget to remove the italic format and change the file Version and Date after changes. The idea is interesting and the document is detailed. Good Job!
15/3	Rattapoom	Please also fill in the Project Timeline.
28/3	Phiranath	Some of the data features are not very useful and take too much time to collect them. The UML is not entirely correct. Overall good document. 👍
29/3	Pattapon	I recommend finding correlations, such as the relationship between damage received and damage dealt or creating a scatter plot to compare floors cleared versus time spent.