GuessIT!

1. Project Overview

This project is a level-based word-guessing game where players have six attempts to guess a randomly selected valid word in each round. Early levels provide hints by revealing some letters, but as the game progresses to increasing the difficulty fewer clues are given. Players earn points based on how quickly they guess correctly, the fewer attempts the higher scores they get. These points can be spent on power-ups at certain stages of the game.

2. Project Review

The similar existing project is called Wordle, a word-guessing game where player have to guess the correct word so the concept is pretty much the same but our project try to improves upon Wordle by introducing:

- **Level and Progression**: Instead of static difficulty, the game scales in challenge by revealing some letter revealing the random amount of letters based on the current level (the amount is based on the criteria explained in the Algorithm section)or change from guessing 5 letter word to 6 letter word as the level progresses.
- **A Scoring System**: Rewards for player when they can solved the word for each round
- **A Shop System and items**: Player can purchase items to help passing the level and enhance in decision making

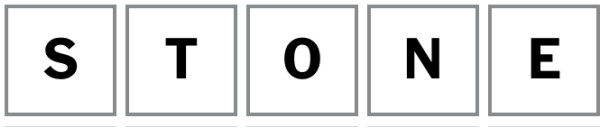
3. Programming Development

3.1 Game Concept

The game objective is to guess a hidden valid 5-letter word within six attempts based on the game rules

Game Rules:

Each letter will have it own box to store like the picture provided



- If a guessed letter **is not in the word**, the box turns **black**.
- If a guessed letter **is in the word but in the wrong position**, the box turns **yellow**.
- If a guessed letter **is in the correct position**, the box turns **green**.

Players will start with some letters revealed, which will be fewer as they level up. They can advance to the next level by scoring enough points for the requirement, The lower the attempt they guess the more points they get. There will be a shop at some checkpoints level that lets players buy items using earned points. The items is going to be like

- “Bomb” -> that can delete one attempt of guessed ex. You guessing for 5 time in that round after use this item the guessing time will reduce to 4
- “Bomb” -> Removes the latest failed attempt from the attempt count. (e.g. if a player has made 5 failed attempts, using this item will restore them to 4 failed attempts instead) in the short term it will delete the latest row of guesses if it is not the correct word yet.
- “Magnifying glasses” -> that can be used to reveal 1-2 letters based on your luck.

Additional items or level-based changes may be introduced later!

3.2 Object-Oriented Programming Implementation

1. Letter Class

- Represents each letter in the player's guess.
- Stores the letter's position, color (based on correctness), and display properties.
- **Methods:** draw(), delete(), update_color().

2. Word Class

- Manages the correct word for each round.
- Checks if a guessed word is correct and assigns colors accordingly.
- **Methods:** check_guess(), generate_new_word(), reveal_letters().

3. Player Class

- Tracks player stats like score, level, and attempts.
- Updates points based on performance.
- **Methods:** update_score(), level_up(), use_powerup().

4. Item Class

- Represents power-ups like **Bomb** (removes a guess attempt) and **Magnifying Glass** (reveals a letter).
- There will be a class for each item like Bomb(), Magnifying_Glass() or more in terms of subclass using the method that inherits from Item class, It will make them perform the effect uniquely to each other.
- **Methods:** apply_effect(), get_item_details().

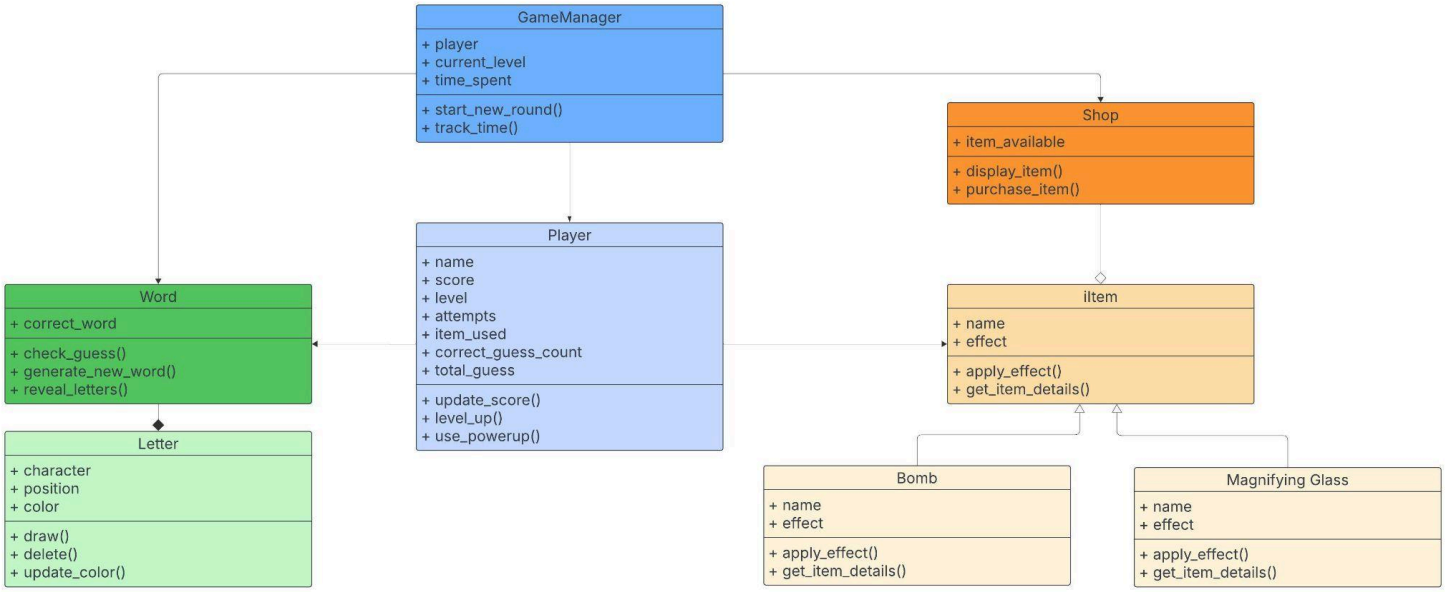
5. Shop Class

- Allows players to buy power-ups using points.
- Manages available items and their costs.
- **Methods:** display_items(), purchase_item().

6. GameManager Class

- Manages game flow, levels and interactions.

UML Class Diagram (some method and attribute still missing from now :))
Add missing attributes as the comment said



3.3 Algorithms Involved

- **Letter Reveal Logic:** Determines how many letters are initially shown.
- Letter Reval Logic: The reveal logic will ensure the number of letters revealed will change based on the level of the player
    1. Levels 1-5: It will random between 2-3 letters reveal and make sure that the revealed letters are spread across the word like it may not be consecutive to each other.
    2. Levels 6-10: it will random between 2-1 letters
    3. Levels 11-19:There will be only 1 letter to be revealed.
    4. Levels 20 and Highers: No letters will be reveal

- **Scoring System:** Awards points based on attempts used.
  (e.g. Players use all 6 guesses; the base score they get is only 10 points if 5 guesses will be 20 points , 4 guesses will be 30 points and so on.) The number can change depending on the price of the item.
  There might be some rules for bonus score as well
    - Letter Discovery Bonus: Players earn points for discovering correct letters in either the right or wrong position (awarded only the first time the letter is discovered)

- First-Time Yellow: (Correct letter but Wrong position) +5 points
- First-Time Green: (Correct letter and Correct position) + 10 points

4. Statistical Data (Prop Stats)

4.1 Data Features

- **Player Score** – Points earned per level.
- **Guess Accuracy** – How close each guess was to the correct word.
- **Attempts Used** – Number of tries before solving the word.
- **Power-Ups Used** – Tracks which items were used.
- **Level Reached** – Tracks which level player reached

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 via graph)?
Player score (Point earn per level)	It can help analyze scoring trends and how level difficulty affects points. It also can be used in balancing the score system and	**Player play manually**, It will record the score earn after each level	score from Player class	Graph (Histogram of score distribution) + Statistic (Average, min, max and more)

	difficulty adjustment later.			
Guess Accuracy	It helps track how well players are guessing letters	**Player play manually**, it will record accuracy (correct letter/guess) for each level	correct_guess_count and total_guesses from Player class	Graph (Line chart showing accuracy improvement from player over multiple levels
Attempts Used	It is used to measure how efficiently players guess the word. Helps in tuning difficulty progression like should i reveal more words or not?	**Player play manually**, It will record attempts used per level(round) throughout 1 gameplayed	attempts from Player class	Graph(Bar chart for attempts distribution) + Statistic (Average, min, max attempts)
Power-Ups Used	It use to track how often player use power-ups which can help in balancing the item and shop system	**Player play manually**, It will record power-up usage throughout 1 gameplayed	Item_used from Player class	Statistic(Usage rate like average or something) + Graph(Pie chart to show which power-up used the most)
Level	It help track	Collect level	level from	Graph(Bar

Reached	fame progression and whether difficulty scaling is effective	**Player play manually**,It will record the final level reached per game	Player class	chart of level reached by players

More specification on statistical

4.1.1 Table (showing statistical value):

There will be 2 tables

- Player scores table
- Attempts to use tables.

Both is displaying the same set of statistical measures (Mean, Median, SD, Min and Max)

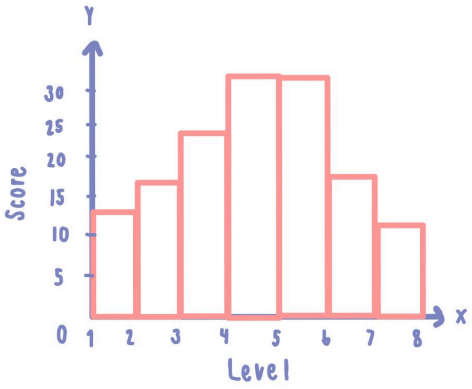
4.1.2 Graphs:

	Feature name	Graph objective	Graph type	X-axis	Y-axis
Graph1	Player score	This graph is used to analyze overall scoring trends.	Histogram	Level	Score
Graph2	Guess accuracy	This graph is used to observe player progress over time(level)	Line chart	Level	Accuracy
Graph3	Power-Ups used	This graph is used to observed player preference and the	Pie chart	Each item count (section)	

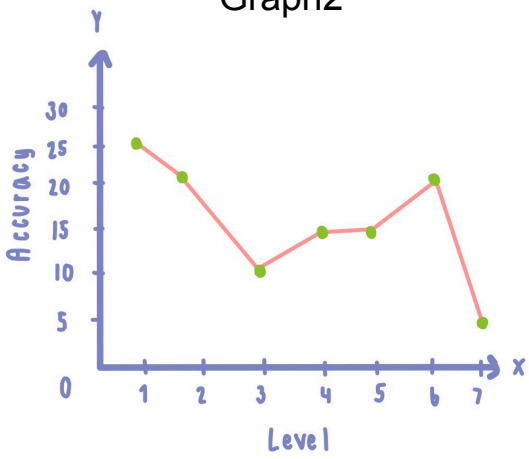
		usefulness of it			
Graph4	Level Reached	This graph is used to observe player progression is the game to hard or not	Bar chart	Level	Count

4.1.3 Hand Drawn Expected Graph:

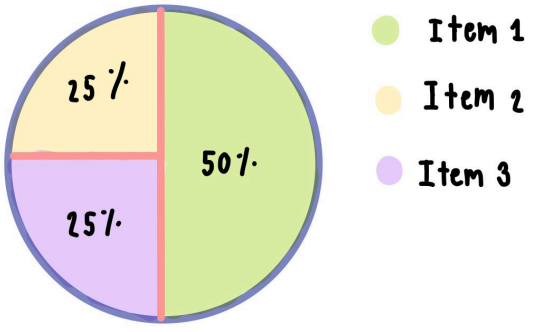
Graph1



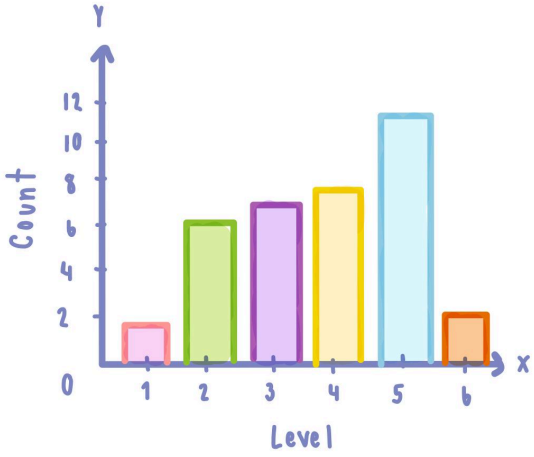
Graph2



Graph3



Graph4



4.2 Data Recording Method

The data will be stored in a CSV file recording player performance for analysis.

4.3 Data Analysis Report

Statistical Measures: Average score, accuracy, win rates, power-up usage rate, attempts per level .

Data Representation: Graphs and tables displaying player progress and trends. Ex. using Bar graph to show the distribution of levels reached by players, Table to summarize player key stats like average score, attempts per level, and most used power-ups and maybe use Line Chart to show accuracy improvement from player over multiple rounds.

4.4 Project Timeline

Week	Task
1 (10 March)	Proposal submission / Project initiation
2 (17 March)	Full proposal submission
3 (24 March)	Develop core game mechanics (input handling, score system, word validation/data)
4 (31 March)	Implement level progression and letter reveal logic
5 (7 April)	Implement power-ups and shop system
6 (14 April)	Submission week (Draft)

4.4.1 Weekly goals

- 26 March - 2 April
    - Develop the main structure of the game such as class Letter, Word, Player.
    - Implement some letter reveal logic and level progression
- 3 April - 9 April
    - Make sure level progression and letter reveal logic work well and scale base on the level
    - Begin working on power-ups (Bomb, Magnifying Glass) and Shop system
    - Make the game playable with basic gameplay functionality
- 10 April - 16 April
    - Complete the shop and power-ups system
    - Testing the game and adjust some balance if needed
    - Start working with csv file for the data analysis part
- 17 April - 23 April
    - Play the game to collect the data and keep tracking if it work well or not
    - Begin working on some data visualization when data is enough
    - Begin working on GUI for both game and the data visualization part
- 24 April - 11 May (Final exam week)
    - Finalize GUI
    - Debugging the game and overall

4.4.2 Milestone goals

- 50%: 16 April (~Song Kran)
    - Playable prototype with functional core mechanics(e.g. Letter revealing, Level progression)
    - Power-ups and shop system to be almost done
    - Adjust some balanced of the game
- 75%: 23 April
    - Able to start collecting the data throughout game played
    - The game can work smoothly with all the function and logic
    - Partial of GUI and data visualization is being done

- 100%: 11 May
    - Test and debugging the game before final submission
    - Finalize on GUI and data visualization

5. Document version

Version: *4*
Date: *31 March 2025*

Date	Name	Description of Revision, Feedback, Comments
14/3	Pattapon	The idea is good overall :), but some parts are a bit unclear as I mentioned in my comments.
16/3	Phiranath	Don't forget to remove the italic fonts in some section and add more detail in the algorithm section.
29/3	Pattapon	Good job. There are some suggestions in my comments.
30/3	Phiranath	Agree with Pattapon. Good job! 👍