Name__Lance_Townsend_____ Mark _____/50

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

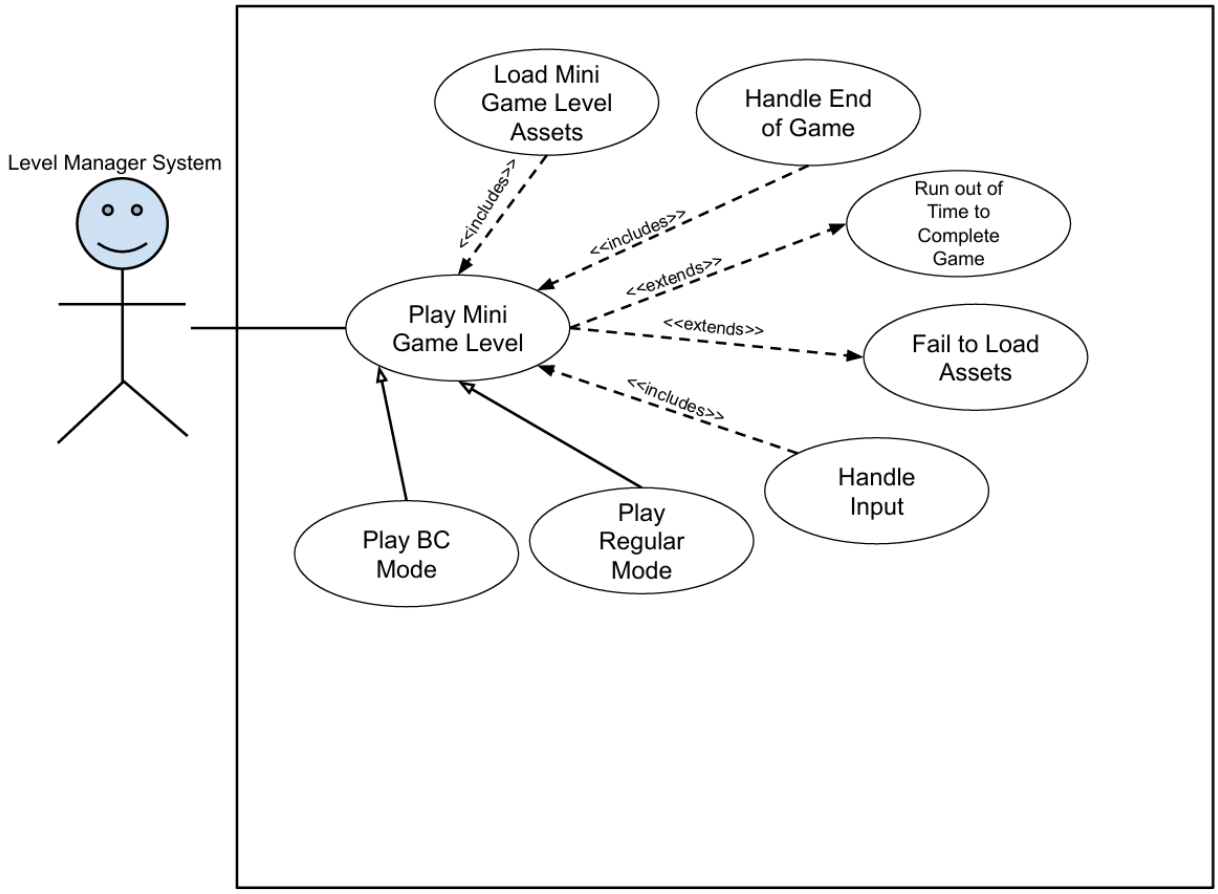
1. Brief introduction __/3

I oversee designing mini games that the player will play to win the dates they are on in our dating simulator. There will be a pool of mini games that can be randomly selected, and I am in charge of implementing those mini games.

The mini games will be games like pong, small platformer, mine sweeper, etc. When the player starts a date, they will be prompted to play a mini game. If they win, then they win the date, and if they lose then they lose the date. The harder the date, the harder the mini games. If a player wins the mini game, then they gain affection points for the specific character they went on a date with.

2. Use case diagram with scenario __14

Use Case Diagrams



Scenarios

[You will need a scenario for each use case]

Name: Load Mini Game Level

Summary: The Level Manger System will request to play a mini game level. To play the mini game level, we will need to know if it is a BC mode game or a regular game, and we will also need the assets for that specific game. We will also need to be able to handle the input and what to do after the game ends. Failing to load the assets and running out of time to complete the mini game are some exceptions that are also possible.

Preconditions: A date has been initialized for a mini game level to be loaded.

Basic sequence:

Step 1: Pick a random date from a list.

Step 2: Determine if it is a BC or Regular difficulty mode mini game.

Step 3: Load the level assets.

Step 4: Handle the input so the player can play the game.

Step 5: When the game is over, determine if the player won. If they did, add affection points to character.

Exceptions:

Step 3: The assets to the level failed to load: Notify the player and terminate the mini game.

Step 4: The player ran out of time to complete the mini game: They lose the mini game.

Post conditions: Assign the correct affection points to the character after the mini game is finished.

Priority: 2

ID: L01

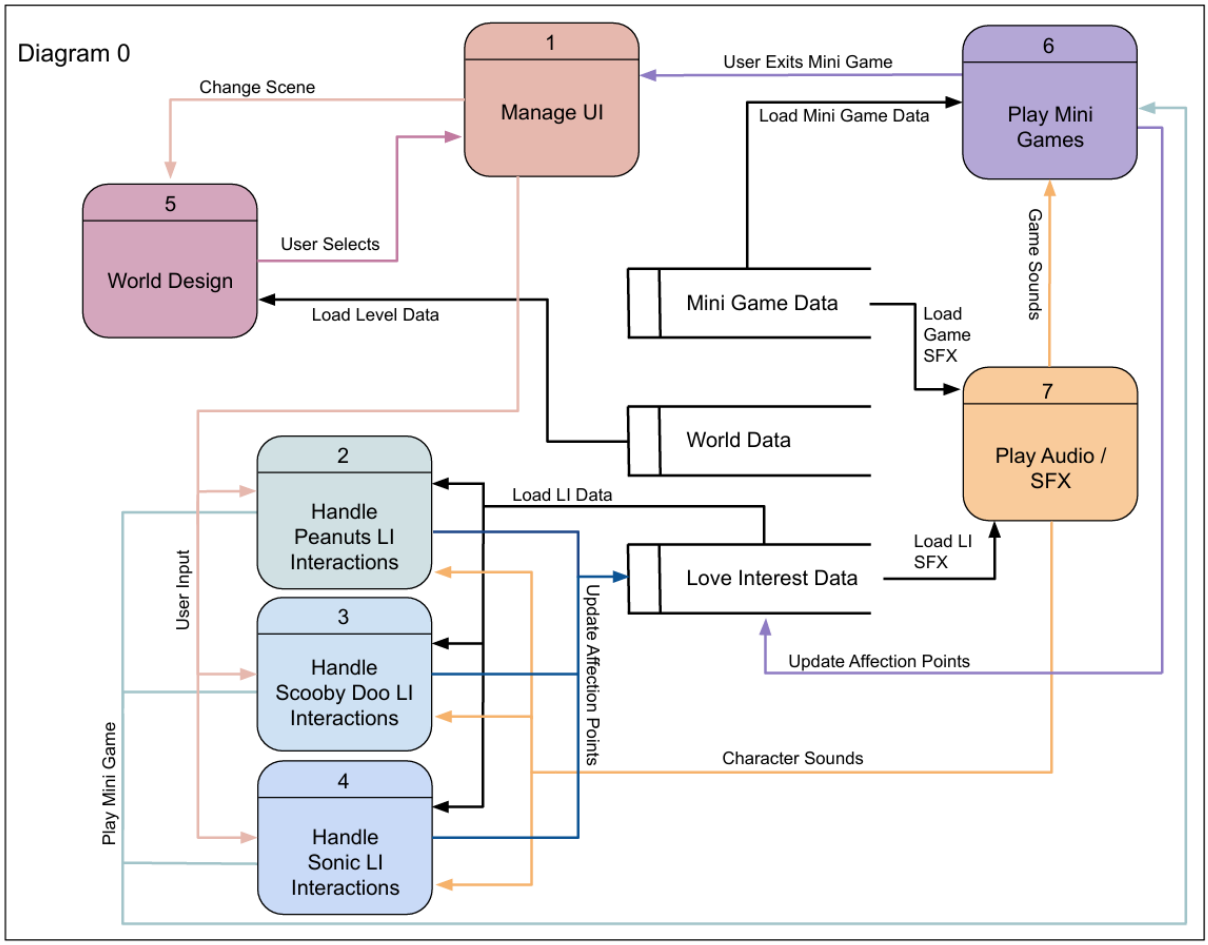
*The priorities are 1 = must have, 2 = essential, 3 = nice to have.

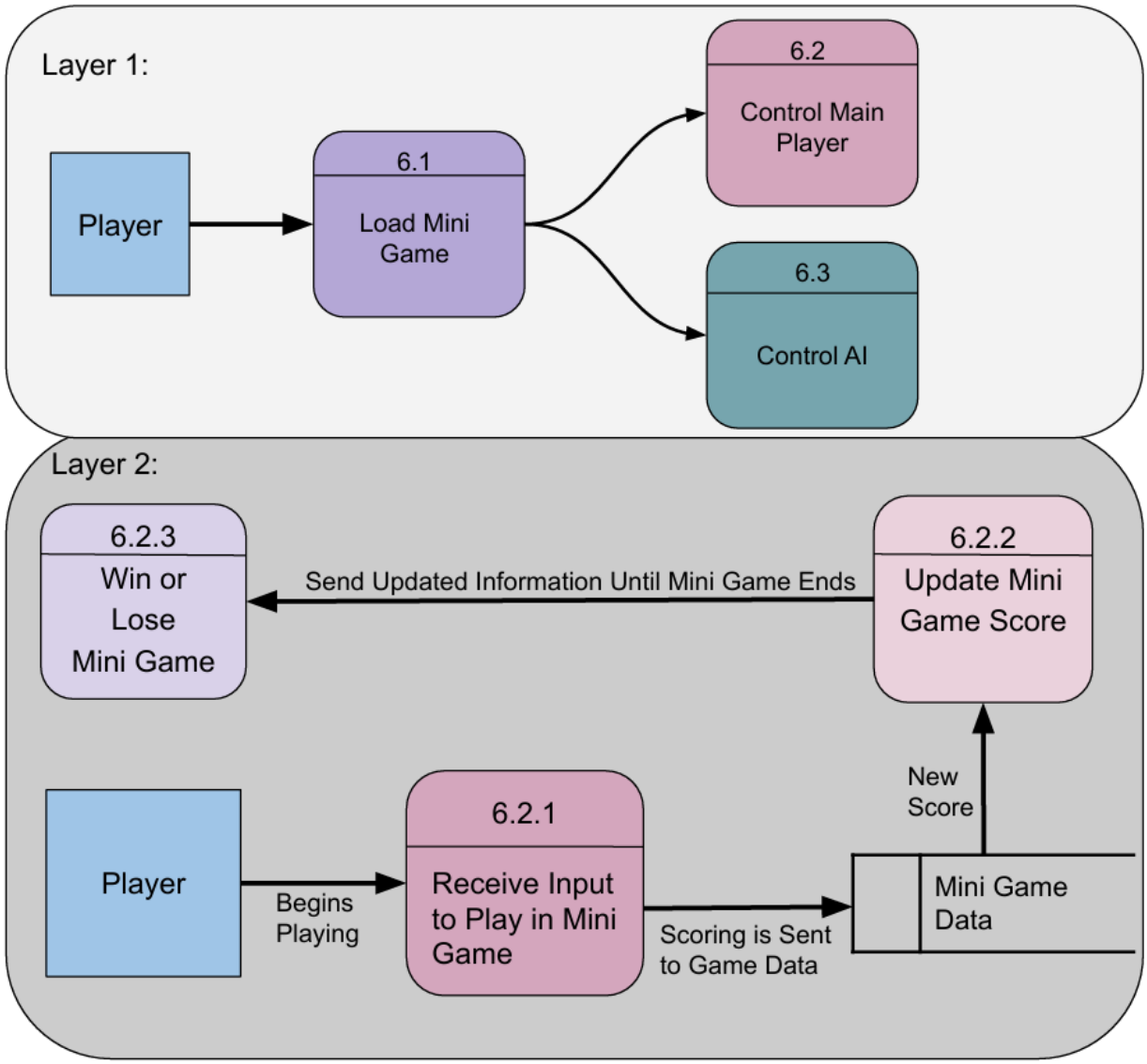
3. Data Flow diagram(s) from Level 0 to process description for your feature _____14

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

Example:

Data Flow Diagrams




Layer 1:
Player
6.1
Load Mini Game
6.2
Control Main Player
6.3
Control AI
Layer 2:
6.2.3
Win or Lose Mini Game
Send Updated Information Until Mini Game Ends
6.2.2
Update Mini Game Score
Player
Begins Playing
6.2.1
Receive Input to Play in Mini Game
Scoring is Sent to Game Data
Mini Game Data
New Score

Process Descriptions

```
BEGIN MiniGame

    WHILE MiniGameNotDone:

        PlayerInput = GetPlayerInput();

        score = CalculateScore(PlayerInput);

        SendScoreToMiniGameData(score);

        newScore = GetScoreFromDatabase();

        UpdateGameDisplay(newScore);

    END WHILE

    finalScore = GetScoreFromDatabase();

    result = CheckWinConditoin(finalScore);

    IF (result == 1) THEN

        DisplayWinMessage();

    ELSE

        DisplayLossMessage();

    END IF

END MiniGame
```

4. Acceptance Tests _____9

Since my feature has only one random aspect, which is which mini game to play, and the rest is not random, the testing is very straightforward. The testing will really be exercising the random aspect with every character and situation. Since the number of characters and mini games is known, we will be able to test every single possible case for my feature.

The input is a date with a character. For testing, we will test the mini games with every character in the game.

To test the mini games, we will run every mini game in BC and Regular mode for all the characters. We will also need to win and lose every single mini game as well as let the time run out for each game. This will have concluded testing every possible condition of my feature.

The output will be the correct affection points added to the character the play is on a date with if the player wins the game, and if they did not win, the points should not be added.

To automate this testing, we will run the mini games AI against itself and set up automated testing to trigger a date with every character. Also, we will set up the testing to ensure that the mini game will win, lose, and time out of every game. Then, it will be simple to ensure that the affection points added to the character is correct.

5. Timeline _____/10

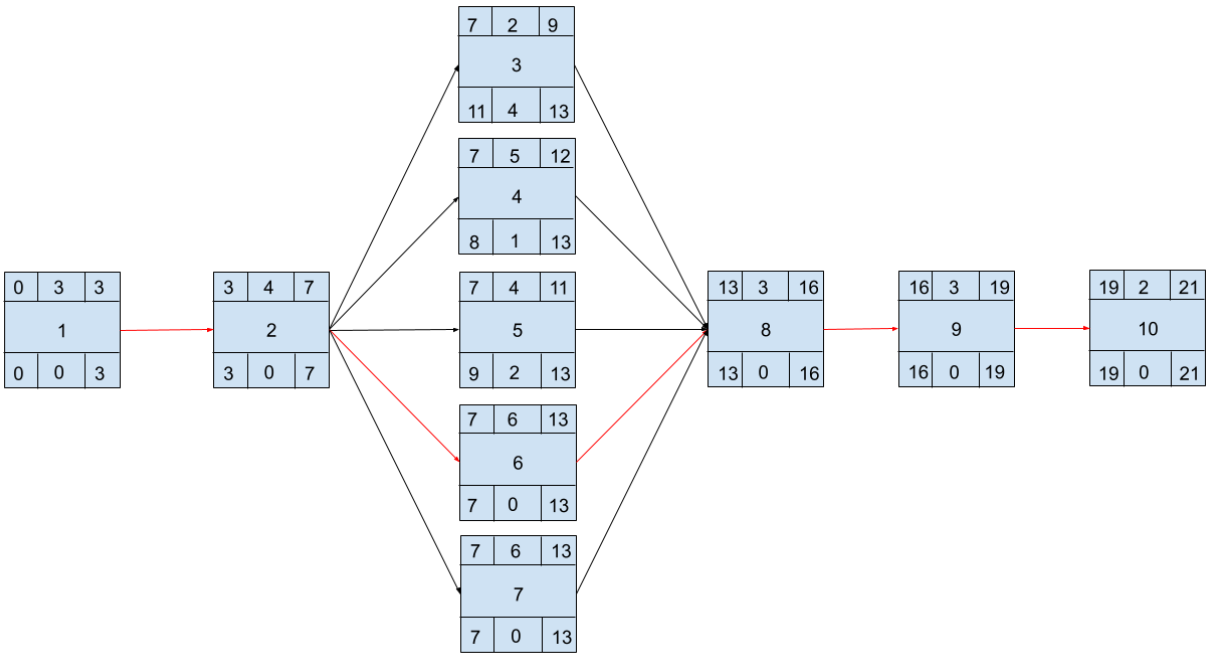
[Figure out the tasks required to complete your feature]

Example:

Work items

Task	Duration (Hours)	Predecessor Task(s)
1. Requirements Collection	3	-
2. Mini Game Level Designs	4	1
3. Pong Mini Game Programming	2	2
4. Math Mini Game Programming	5	2
5. Platformer Mini Game Programming	4	2
6. Memory Test Mini Game Programming	6	2
7. Minesweeper Mini Game Programming	6	2
8. User Documentation	3	3, 4, 5, 6, 7
9. Testing	3	8
10. Installation	2	9

Pert diagram



Gantt timeline