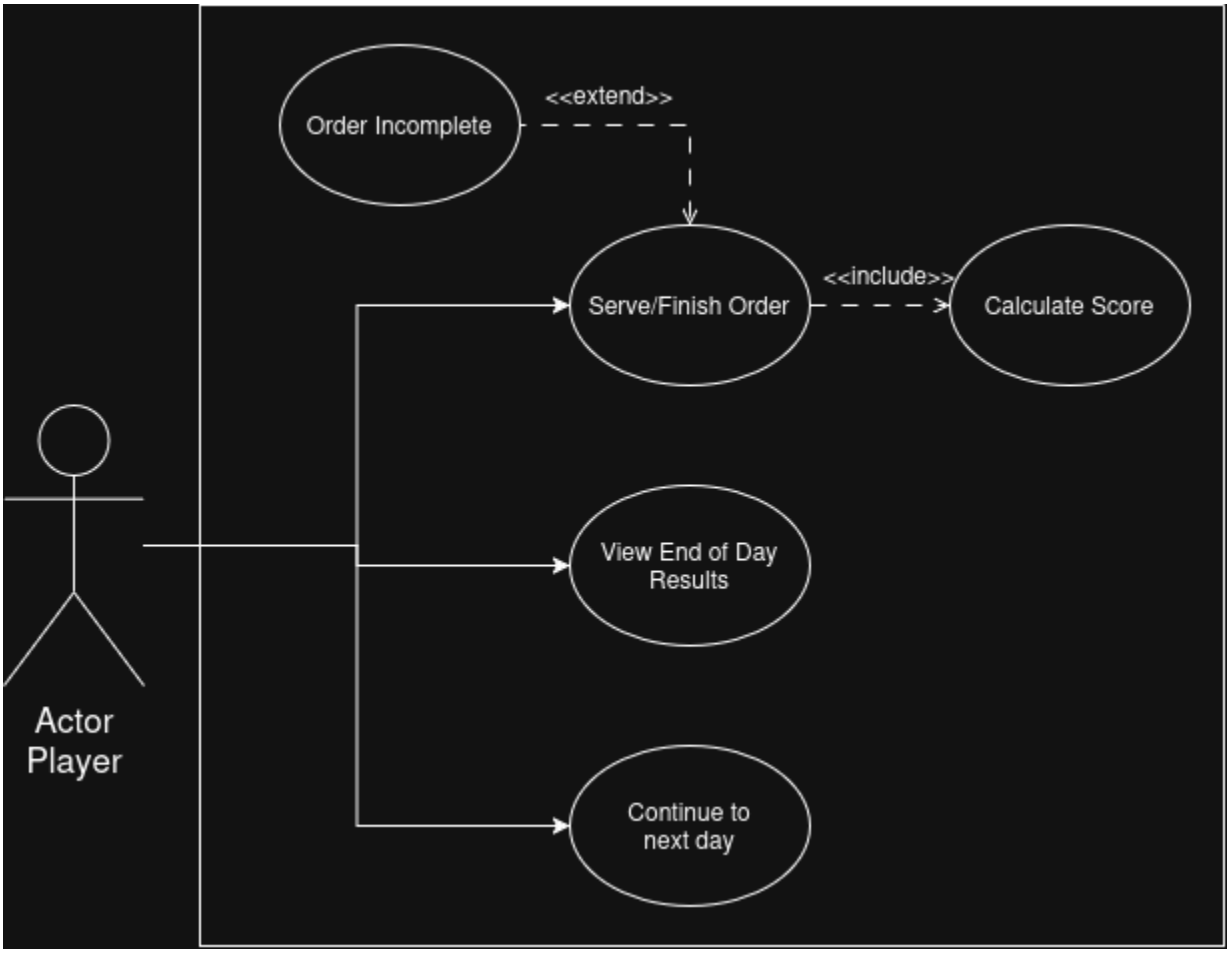
Name: Noah Turner Mark _____/50

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

1. Brief introduction __/3

ET-Pizzeria is a 2D mobile time-management game where the player completes pizza orders through multiple steps (ordering → toppings → baking → cutting → finish). This feature covers the final "delivery" portion of the gameplay loop: boxing/serving an order, validating that the pizza meets the customer request, calculating a score/tip, updating day totals, and displaying an end-of-day results screen. The feature also includes the scene transitions between the final gameplay step and the results screen.

2. Use case diagram with scenario __14



Scenarios

Use Case ID: F01

Name: Serve/Finish Order

Basic Sequence

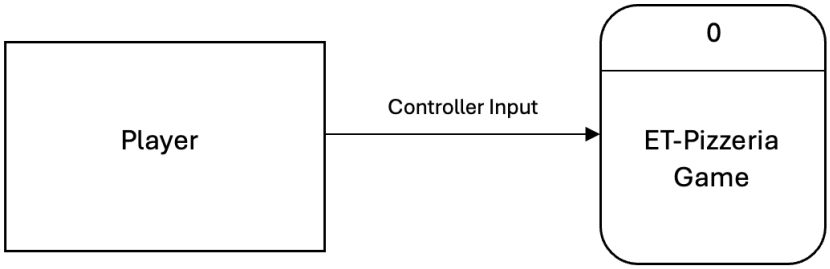
1. Player opens End-of-Day Results screen.
2. System retrieves day summary values and mistake totals.
3. System calculates total score from base value minus penalties.
4. System displays totals and mistake counts.
5. Player selects Main Menu or Restart.
6. System transitions to selected screen.

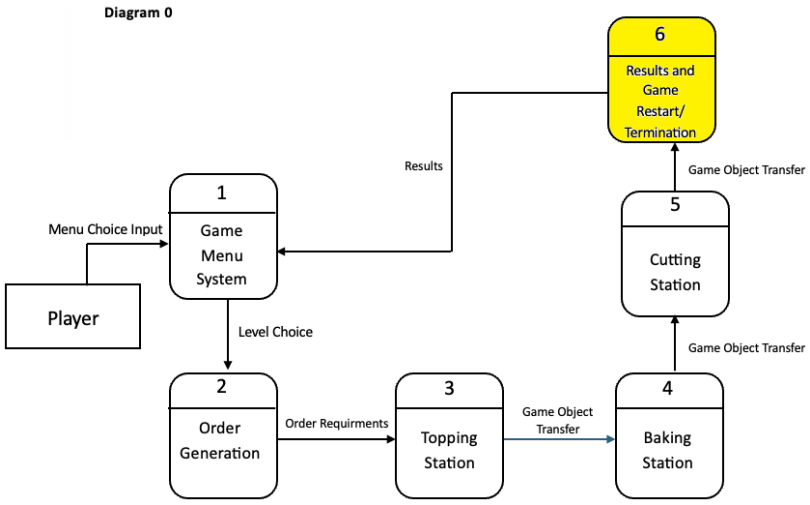
Exceptions / Alternative Flows

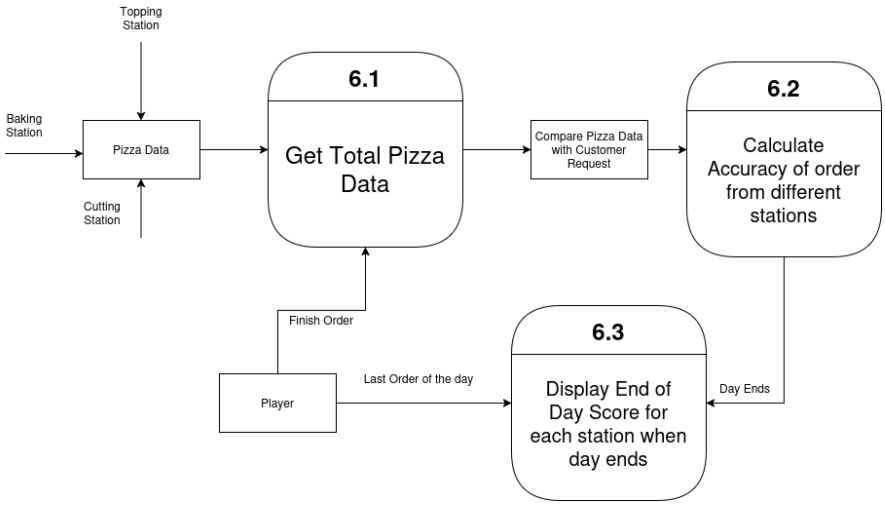
- **2a.** Day summary data not found → system uses default zero values.
- **3a.** Score calculation input missing/invalid → system sets score to 0 and displays warning/debug message.
- **6a.** Transition target fails to load → system returns to safe default (Main Menu).

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

Data Flow Diagrams







Process Descriptions

// ---------- 6.1 Get End of Day Customer values ---------- function GetEndOfDayValues(): // pull whatever your other scenes saved (order results / mistakes) baseTotal = sum(allOrders.basePay) mToppings = sum(allOrders.mistakesToppings) mBake = sum(allOrders.mistakesBake) mCut = sum(allOrders.mistakesCut) mFinish = sum(allOrders.mistakesFinish) return { baseTotal, mToppings, mBake, mCut, mFinish }

// ---------- 6.2 Calculate Accuracy of order from different stations ---------- function CalculateTotalScore(values): penalty = 10*values.mToppings + 15*values.mBake + 10*values.mCut + 10*values.mFinish totalScore = max(0, values.baseTotal - penalty) return { totalScore, penalty }

// ---------- 6.3 Display Total Scores and mistakes in End Screen ---------- function ShowEndScreen(): values = GetEndOfDayValues() // 6.1 calc = CalculateTotalScore(values) // 6.2

UI.SetText("Base Total", values.baseTotal)
UI.SetText("Topping Mistakes", values.mToppings)
UI.SetText("Baking Mistakes", values.mBake)
UI.SetText("Cutting Mistakes", values.mCut)
UI.SetText("Finish Mistakes", values.mFinish)
UI.SetText("Penalty", calc.penalty)
UI.SetText("Final Score", calc.totalScore)

// button outputs shown in your diagram
UI.OnClick("Main Menu", LoadMainMenu)
UI.OnClick("Restart From Beginning", RestartGame)

4. Acceptance Tests _____9

[Describe the inputs and outputs of the tests you will run. Ensure you cover all the boundary cases.]

Example for random number generator feature

Run feature 1000 times sending output to a file.

The output file will have the following characteristics:

- Max number: 9
- Min number: 0
- Each digit between 0 and 9 appears at least 50 times
- No digit between 0 and 9 appears more than 300 times
- Consider each set of 10 consecutive outputs as a substring of the entire output. No substring may appear more than 3 times.

Example for divide feature

Output	Numerator (int)	Denominator (int)	Notes
0.5	1	2	
0.5	2	3	We only have 1 bit precision for outputs. Round all values to the nearest .5
0.0	1	4	At the 0.25 mark always round to the nearest whole integer
1.0	3	4	At the 0.75 mark always round to the nearest whole integer
255.5	5	0	On divide by 0, do not flag an error. Simply return our MAX_VAL which is 255.5.

5. Timeline _____/10

[Figure out the tasks required to complete your feature]

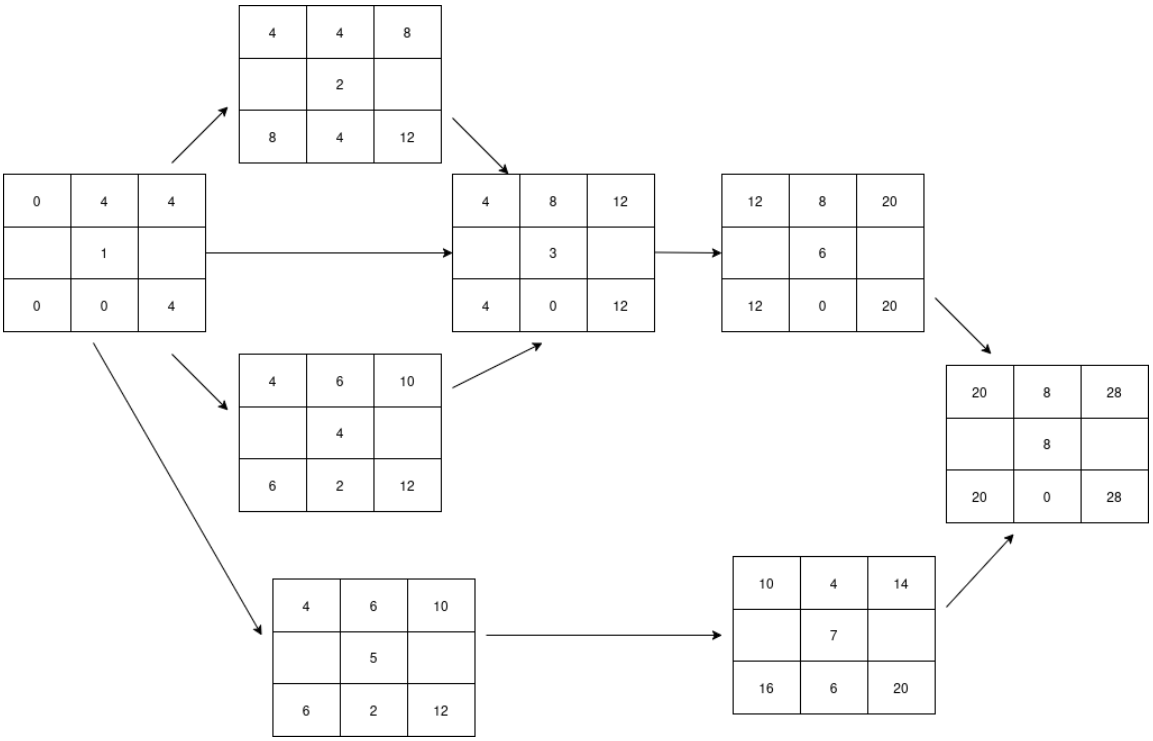
Example:

Work items

Task	Duration (Hours)	Predecessor Task(s)
1. Define result fields + scoring inputs	4	-
2. Implement OrderResult class	4	1
3. Implement DaySummary totals/mistake aggregation	6	1

4. Implement ScoreCalculator basic logic	8	1
5. End screen UI hookup (display values)	8	2,3,4
6. Main Menu / Restart button transitions	4	5
7. Integration test (full end screen flow)	6	5,6
8. Bug fixes / polish	8	7

Pert diagram



Gantt timeline

	2	4	6	8	10	12	14	16	18	20	22	24	26
1													
2													
3													
4													
5													
6													
7													

8													