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

Name: Serve/Finish Order

Summary: The player serves the completed pizza order; the system validates completion, calculates score/tip, updates day totals, and transitions to results.

Actors: Player

Preconditions: An order exists and is currently active; player is on the Finish/Serve screen; pizza state from previous steps is available (toppings, bake state, cut state).

Basic sequence:

Player taps Serve/Finish.

System verifies required completion flags are present (order exists, pizza created, bake state recorded, cut state recorded, order is boxed/ready).

System calls [Calculate Score] (<<include>> UC-I01).

System updates day totals (money earned, accuracy %, tips, number of orders served).

System displays Serve Summary feedback (e.g., accuracy %, tip amount).

System transitions to End-of-Day Results when day/order count is complete (or transitions back to the next order if day is not complete).

Exceptions: (must match your <<extend>> and line up with a Basic Sequence step)

03_Champion_template

Step 2 Exception: Serve tapped but pizza is missing a required state (not baked / not cut / not boxed / no order loaded).

System blocks serve, displays message like "Order not ready—complete all steps," and remains on the Finish screen.

Step 3 Exception: Score calculation fails due to missing comparison data (order requirements not loaded).

System returns score = 0, logs error (debug), and displays minimal feedback.

Postconditions: Day totals reflect the served order; results/next-state screen is shown; current order is marked complete.

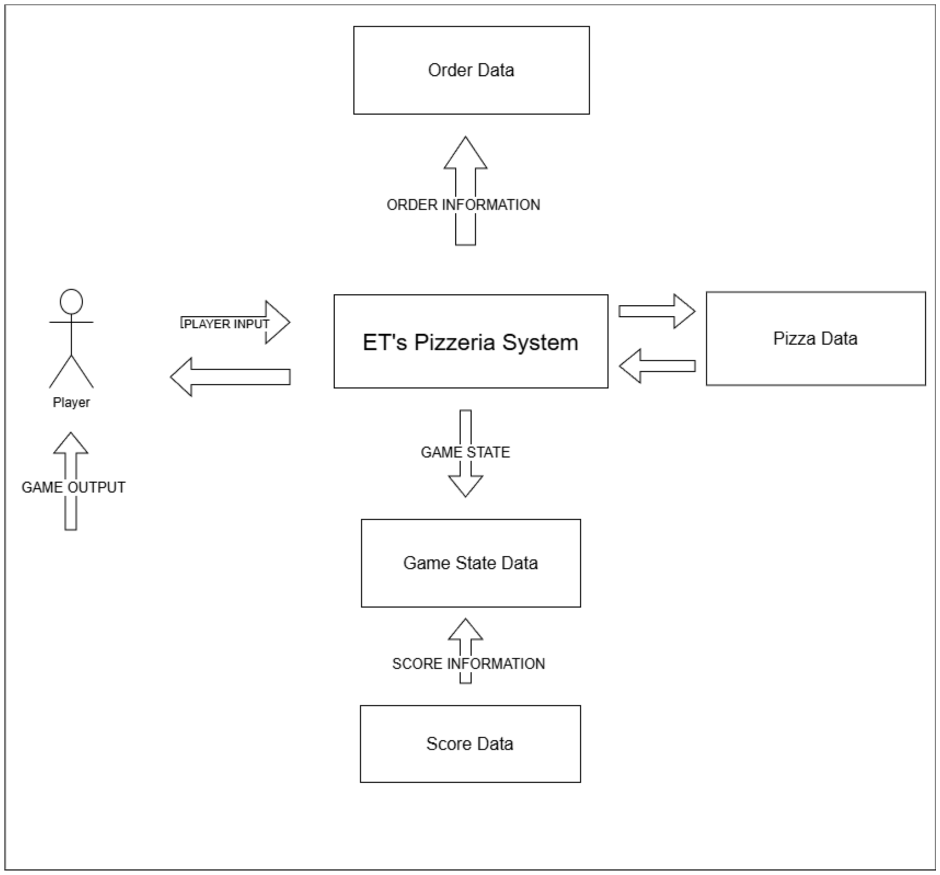
Priority: 1 (must have)
ID: F01

Name: View End-of-Day Results
Summary: The player views a results screen summarizing performance for the day.
Actors: Player
Preconditions: At least one order has been served OR the day has ended.
Basic sequence:
System loads day summary totals.
System displays results: orders served, average accuracy, total earned, total tips, time bonuses/penalties.
Player taps Continue.
Exceptions:
Step 1 Exception: Day summary not found (first run or data reset).
System displays zeros/defaults and continues.
Postconditions: Player can proceed to next day or main menu.
Priority: 1
ID: F02

Name: Continue (Next Day / Main Menu)
Summary: The player exits the results screen and proceeds.
Actors: Player
Preconditions: Results screen displayed.
Basic sequence:
Player selects Next Day or Main Menu.
System resets day counters if starting a new day; persists last day summary; transitions scenes.
Exceptions:
Step 2 Exception: Save/persist fails.
System continues but shows warning (“progress may not be saved”).
Postconditions: New game state is loaded.
Priority: 2 (essential)
ID: F03

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

Data Flow Diagrams



Process Descriptions

Process 5.1 Validate Completion:
Inputs: Order Data, Pizza Data
Output: Validation Status (ready/not ready)
Rules: block serving if baked/cut/boxed/order missing; return message to UI.

Process 5.2 Compute Score & Tip:
Inputs: Order Data requirements + Pizza Data results + optional timing info
Outputs: Accuracy %, Tip $, Final Score (stored to Score Data)

Process 5.3 Update Day Summary:
Inputs: Final Score + Tip + Accuracy
Outputs: updated totals (orders served, total earned, average accuracy) stored in Game State Data

Process 5.4 Display Results + Transition:

Inputs: Game State Data + latest Score Data
Outputs: results UI to Player; transition to next order or End-of-Day Results depending on “day complete” state

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]

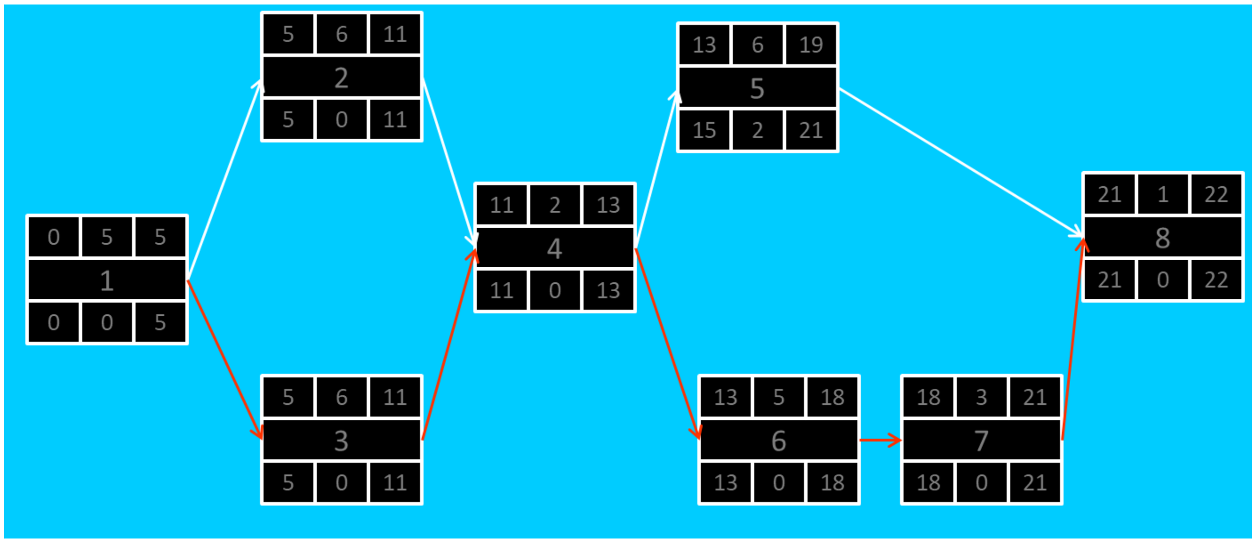
Example:

Work items

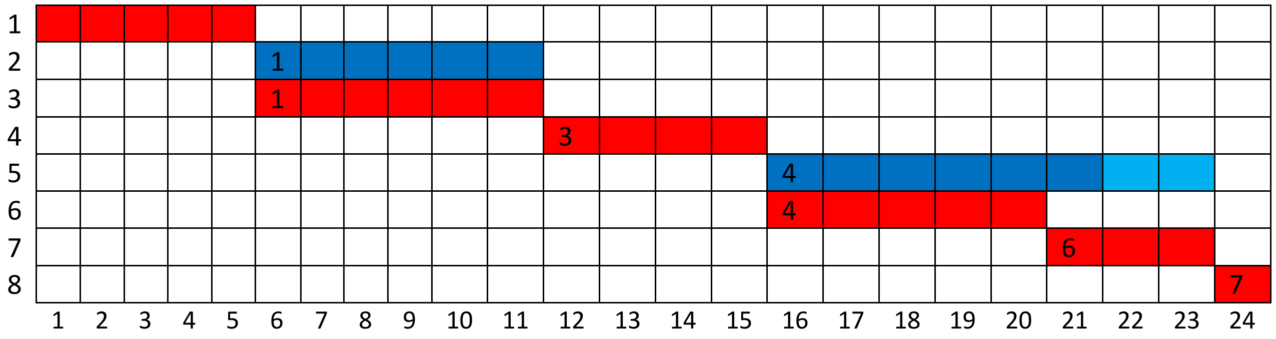
Task	Duration (PWks)	Predecessor Task(s)
1. Requirements Collection	5	-

2. Screen Design	6	1
3. Report Design	6	1
4. Database Construction	2	2, 3
5. User Documentation	6	4
6. Programming	5	4
7. Testing	3	6
8. Installation	1	5, 7

Pert diagram



Gantt timeline



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