Problem Statement and Goals
ProgName

Team #, Team Name
Student 1 name
Student 2 name
Student 3 name
Student 4 name

Table 1: Revision History

Date	Developer(s)	Change
Date1	Name(s)	Description of changes
Date2	Name(s)	Description of changes
...	...	...

1 Problem Statement

1.1 Problem

1.2 Inputs and Outputs

1.3 Stakeholders

1.4 Environment

2 Goals

3 Stretch Goals

4 Extras

Appendix — Reflection

The purpose of reflection questions is to give you a chance to assess your own learning and that of your group as a whole, and to find ways to improve in the future. Reflection is an important part of the learning process. Reflection is also an essential component of a successful software development process.

Reflections are most interesting and useful when they're honest, even if the stories they tell are imperfect. You will be marked based on your depth of thought and analysis, and not based on the content of the reflections themselves. Thus, for full marks we encourage you to answer openly and honestly and to avoid simply writing "what you think the evaluator wants to hear."

Please answer the following questions. Some questions can be answered on the team level, but where appropriate, each team member should write their own response:

1. What went well while writing this deliverable?
2. What pain points did you experience during this deliverable, and how did you resolve them?
3. How did you and your team adjust the scope of your goals to ensure they are suitable for a Capstone project?

Table 2: Minimum Viable Product (MVP) Goals

Goal	**Explanation**	**Reasoning**
Two-Player Core Loop	Support a one-versus-one match with turn-taking, drawing, discarding, and win condition checks. Includes starter card, discard pile, and reshuffling the stock pile.	Two players represent the smallest playable unit. Completing this ensures the core gameplay is functional and testable.
Classic Rules Engine	Implement the standard Crazy Eights rules: match by suit or rank, "8" acts as wild, and drawing occurs if no valid move exists.	Ensuring correctness of the classic game establishes a solid baseline before adding variations.
Dozenal (Base-12) Scoring/Display	Display scores, counters, or thresholds in base-12 notation while keeping classic rules unchanged.	Introduces dozenal in a simple, non-disruptive way that highlights novelty while retaining accessibility.
Move Validation and Feedback	Provide immediate feedback for invalid moves, suit selection UI after playing an "8," and clear state indicators.	Reduces errors, lowers learning curve, and improves user experience.
Testability and Determinism	Support seeded shuffling and provide basic logs or replays.	Facilitates unit/integration testing and reproducibility during evaluation.
Stability and Performance	Ensure responsive UI (¡200ms), no crashes, no deadlocks, and correct reshuffling.	Reliability is the baseline for acceptance and live demonstration.
Minimal UI	Provide a desktop or web interface showing hand, discard pile, current state, and dozenal score tracker.	Covers essential user interactions while limiting complexity at MVP stage.

Table 3: Stretch Goals

Goal	**Explanation**	**Reasoning**
3–4 Player Matches	Extend gameplay to support three or more players in a single match.	Crazy Eights is often played with more than two people, which increases replayability.
Online Multiplayer	Allow players to create/join rooms and synchronize state across network connections.	Brings the game closer to real-world usage and demonstrates system design capability.
Advanced Dozenal Variants	Add optional rule packs where cards related to 12 gain special effects or scoring thresholds use base-12 values.	Deepens the dozenal theme while keeping the classic mode intact.
Rule Configurator	Provide toggles for house rules such as stacking eights or alternate scoring methods.	Demonstrates variability management and supports experimentation.
Tutorial and Visual Guidance	Include first-game tutorial, invalid-move highlights, and play suggestions.	Lowers the learning curve and improves usability.
Save/Replay System	Allow saving and replaying completed games.	Supports debugging, user study, and richer documentation.
Cross-Platform Packaging	Deploy as a web app or desktop executable.	Lowers barrier for evaluators and external users to try the system.