**Number Mastermind  
Sidu Thallikar**

# PROJECT PROBLEM STATEMENT

# The project is a number-based logic game, inspired by the classic "Mastermind," aimed at strengthening a user’s deductive thinking, memory, and pattern recognition. The program generates a secret 4-digit number with no repeating digits, and the user will have to guess the number in as few tries as possible using feedback on correct digits and placements. This makes for an engaging and brain-stimulating project experience.

# CONCISE DESIGN OVERVIEW

# The Number Mastermind game will be coded in Java, using a console-based interface for interaction through the terminal. A random number with 4 unique digits will be generated by the program. For each guess the user inputs, the game will provide feedback such as how many digits are correct and in the correct place ("bulls") and how many are correct but in the wrong place ("cows"). The program will loop until the correct number is guessed or the maximum guess limit is reached. Input will be validated to ensure guesses are numeric and 4 digits long.

# SCOPE STATEMENT

# The goal is to make a simple yet mentally stimulating number game that helps users build logical reasoning and improve mental calculation. The program should provide real-time feedback and allow for repeated plays, tracking each round’s number of guesses. A clean and user-friendly interface in the console is also part of the project scope, along with reliable guess checking.

# TIMELINE

A graph with multiple squares

AI-generated content may be incorrect.

# TEST PLAN

# Testing will focus on ensuring the accuracy of the random number generation, correctness of the bull and cow feedback logic, and validation of user input. Unit testing will be conducted using Java’s built-in debugger, checking for edge cases such as invalid inputs and repeated digits. System testing will be done manually by simulating user guesses and verifying if the feedback matches the hidden number. The game will also be tested for smooth restarts and proper handling of win/loss scenarios.

# RISKS AND CONTINGENCY PLANS

One potential risk is the random number generator producing repeated digits, making the game unsolvable. This will be avoided by adding checks to prevent digit repetition. Another risk is incorrect feedback logic, leading to misleading hints to the user. This will be addressed through careful step-by-step testing of different guess scenarios. Lastly, input validation may fail if unexpected input types (like letters) are entered, so try-catch blocks and proper input filtering will be used to prevent crashes.