

CheckM8

By

Dylan Salmo #885349860

Ashton Alva #8886545557

Brenen Meregilano #886479021

Problem Statement:

We will develop a chess game that enables a human player to compete against an AI opponent. The AI should make logical and legal moves and respond quickly so the games feel natural. The project shows how artificial intelligence can be used for decision-making, strategy, and reasoning in a classic two-player game.

AI Concepts:

- Minimax Search: looks ahead a few moves to pick the best option for the AI and the worst for the player
- Alpha-beta Pruning: makes the search faster by skipping bad branches
- Heuristic Evaluation: gives each board position a score when the AI can't search deeper

Game Board - represents the chessboard as an 8x8 grid and display it on the screen

Move generation: write code so that all chess pieces move legally (no invalid moves are possible)

AI Move Logic-

- Use minimax with alpha beta pruning to choose next best move
- Limit search depth to 3-5 moves ahead for speed

EvaluationsFunction - score positions based on

- Material balance(piece value)
- King safety
- Piece mobility (number of legal moves)
- Center control

Gameplay loop - alternate between human and AI response until checkmate, stalemate, or draw

Interface - graphical board showing moves and turns.

Evaluation Method:

Accuracy:

- Run reference move-count checks on a few simple positions (depth 2–3): count all legal positions reachable and compare totals to known values.

Time to solution:

- Measure average time per move at depth 3–4 on 10 random middlegame positions.
 - Report the mean and variance; target ≤ 1 second at depth 3 on a standard laptop.
-

Team Roles

Ashton:

- Backend development
- AI logic and behavior systems

Brennen:

- Frontend development (UI/UX design, game interface)
- Asset creation (graphics, visuals, and effects)
- Quality Assurance (QA) testing

Dylan:

- Backend development
 - Core game logic and mechanics implementation
-

Project Timeline

Week 1 – Research & Initial Requirements

- Conduct background research on tools, frameworks, and technologies.
- Draft initial project requirements and core features list.

Week 2 – Requirements Refinement & System Design

- Finalize functional and non-functional requirements.
- Design overall system architecture and logic flow.
- Create basic design mockups or game flow diagrams.

Week 3 – Logic Refinement & GUI Implementation

- Finalize core logic structure and data handling.
- Begin GUI/Frontend development.
- Integrate preliminary backend with frontend.

Week 4 – Quality Assurance & Testing

- Perform unit tests on core logic and backend components.
- Conduct performance/speed tests.
- Gather feedback and identify bugs or inconsistencies.

Week 5 – Bug Fixes & Final Polish

- Fix identified bugs and optimize code.
- Refine UI/UX and finalize assets.
- Prepare final build for presentation or deployment.

Programming Languages / Tools:

- Language: Python

- Core tools:
 - pytest for unit tests
 - cProfile / simple timing for speed checks
 - argparse for a small command-line runner
 - matplotlib (optional) for quick charts of results
 - Git + GitHub for version control black + ruff for formatting/linting
- Inputs for tests:
 - A small text/CSV file of sample chess positions and a few tactic positions (stored as simple strings) to drive accuracy and timing tests.

Special Requirements(if needed):

None.