**Gomoku AI Project Report**

This report describes the implementation of a Gomoku (Five in a Row) game with an AI opponent utilizing minimax algorithm with alpha-beta pruning, implemented as a web application with Python/FastAPI backend and JavaScript frontend.

# **AI Implementation**

**1. Game Tree Search & Minimax Algorithm:**

Implements recursive minimax algorithm to evaluate possible moves, maximize AI advantage while minimizing opponent advantage. Search depth adapts based on game stage (1-3 levels).

**2. Alpha-Beta Pruning:**

Enhances minimax with branch elimination to reduce evaluation nodes, allowing deeper searches within computational constraints.

**3. Evaluation Function:**

Scores board positions by detecting patterns (five-in-row, open/closed fours/threes/twos) across all directions with weighted values (100-100,000). Compares AI vs opponent patterns to determine optimal moves.

**4. Early Stopping Mechanism:**

Implements immediate termination for winning moves, depth-limited search, and adaptive depth based on game stage (early: depth=1, mid: depth=2, late: depth=3).

**5. Move Ordering Optimization:**

Sorts potential moves by preliminary evaluation to maximize alpha-beta pruning efficiency by examining promising moves first.

# **User Interface**

Responsive web interface with HTML/CSS/JS featuring wooden-textured board, realistic stone rendering, game mode selection, and visual feedback. Interactive elements enhance user experience.

# **Architecture**

Client-server architecture with Python/FastAPI REST backend and JavaScript frontend. Modular design separates AI logic, game mechanics, and API handling for maintainability.