

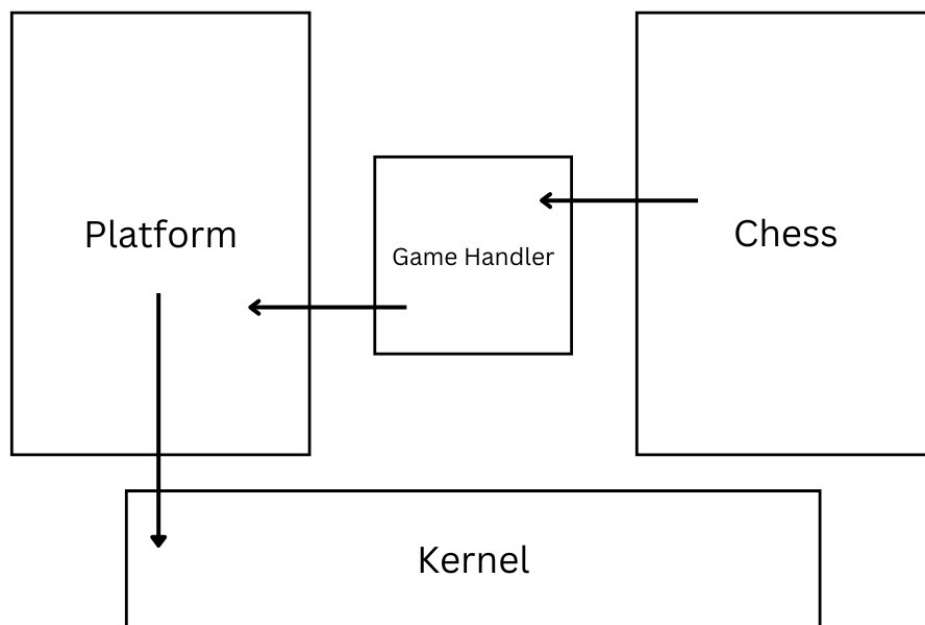
# Architectural Documentation

Project Name: Chess-Simulator  
Date: 26-February-2025  
Version: 0.0.1  
By: Tejas Kayande and Sammed Burse

## 1 Overview

This documentation dives into the architectural choices made to develop the application and an overview of the design choices and the architecture.

## 2 Architecture



*Fig. Architecture of the application*

The architecture of this application is divided into 3 components that are elaborated as follow:

### 2.1 Platform

This layer provides all the Operating System-specific utilities, such as:

- Creating a window to display the game.
- Handling keyboard and mouse inputs.
- Providing functions to draw pixels on the screen.

This is the only layer that interacts with the kernel of the Operating System and makes OS calls. This approach helps maintain a clean codebase and reduces the number of kernel calls by batching them efficiently.

## 2.2 Chess

This layer is responsible for handling the core chess logic, including:

- **Board Representation:** Manages the 8x8 chessboard state.
- **Move Validation:** Ensures that all moves adhere to standard chess rules.
- **Game Rules Enforcement:** Implements check, checkmate, stalemate, and special moves (castling, en passant, and pawn promotion).
- **Move History & Tracking:** Stores previous moves and allows for undo/redo functionality.

## 2.3 Game Handler

The Game Handler is the bridge between the Platform and Chess layers, managing game flow and interactions. Key responsibilities include:

- **Turn Management:** Alternates moves between players.
- **User Input Processing:** Captures and translates player actions into game commands.
- **Rendering Coordination:** Requests the Platform layer to draw the board and pieces based on game state updates.
- **Game State Management:** Determines whether the game is in progress, paused, or has ended.
- **UI Updates:** Displays status messages (e.g., check, checkmate) and maintains a smooth user experience.