ChessArena
Project Proposal

[By CodePlay]

Eddie Xu
Jeffery M Joseph
Samuel Park

Revision date:
14-02-2024

Table of Contents

1. Introduction

1.1. Purpose

Describe what kind of project you propose (desktop application, game, mobile application, etc.). Describe your idea. Justify why you would like to explore, design and implement a project of this kind. What would you learn from this project?

Creating a chess game in C# presents a comprehensive learning opportunity that spans programming logic, algorithmic thinking, and user interface design, all centered around the classic game. This project will challenge us to implement the nuanced rules of chess, develop an intuitive user interface, and perhaps explore AI for single-player modes, leveraging C#'s robust features and .NET framework. It's an exercise in applying object-oriented principles, understanding event-driven programming for responsive gameplay, and enhancing our collaborative skills within a team environment. The endeavor not only aims to bolster our technical proficiency in software development but also to deliver a compelling digital chess experience, demonstrating the power of C# in bringing traditional board games to life in the digital realm.

1.2. Team contact information

1. Name: Eddie Xu
   Email: wendixu2018@gmail.com
   Phone: 778-867-1630

2. Name: Jeffery M Joseph
   Email: jefferymjoseph@gmail.com
   Phone: 236-863-8061

3. Name: Samuel Park
   Email: sampark1080@gmail.com
   Phone: 604-618-5895

1.3. Scope of the project

1.3.1 Included in the Scope

- Basic Chess Functionality: Implementing the core mechanics of chess, including movement rules for all pieces, turn-based gameplay, and the conditions for check and checkmate.
- Graphical User Interface (GUI): Developing a user-friendly interface in C# to interact with the game, displaying the chessboard, pieces, and allowing for mouse or touch-based inputs for moving pieces.
- Basic Save/Load Functionality: Enabling players to save their game progress and load it later to resume play.

1.3.2 Not-Included in the Scope

- Comprehensive User Account System: While basic player identification may be included, a full-fledged user account system with profiles, rankings, and matchmaking
- Single-Player Mode: Creating a basic AI opponent for players to compete against, focusing on simple algorithms for piece movement and decision-making.
- Extensive Customization Options: Offering extensive options for customizing the game's appearance, such as multiple chessboard and chess piece themes

2. Project Proposal

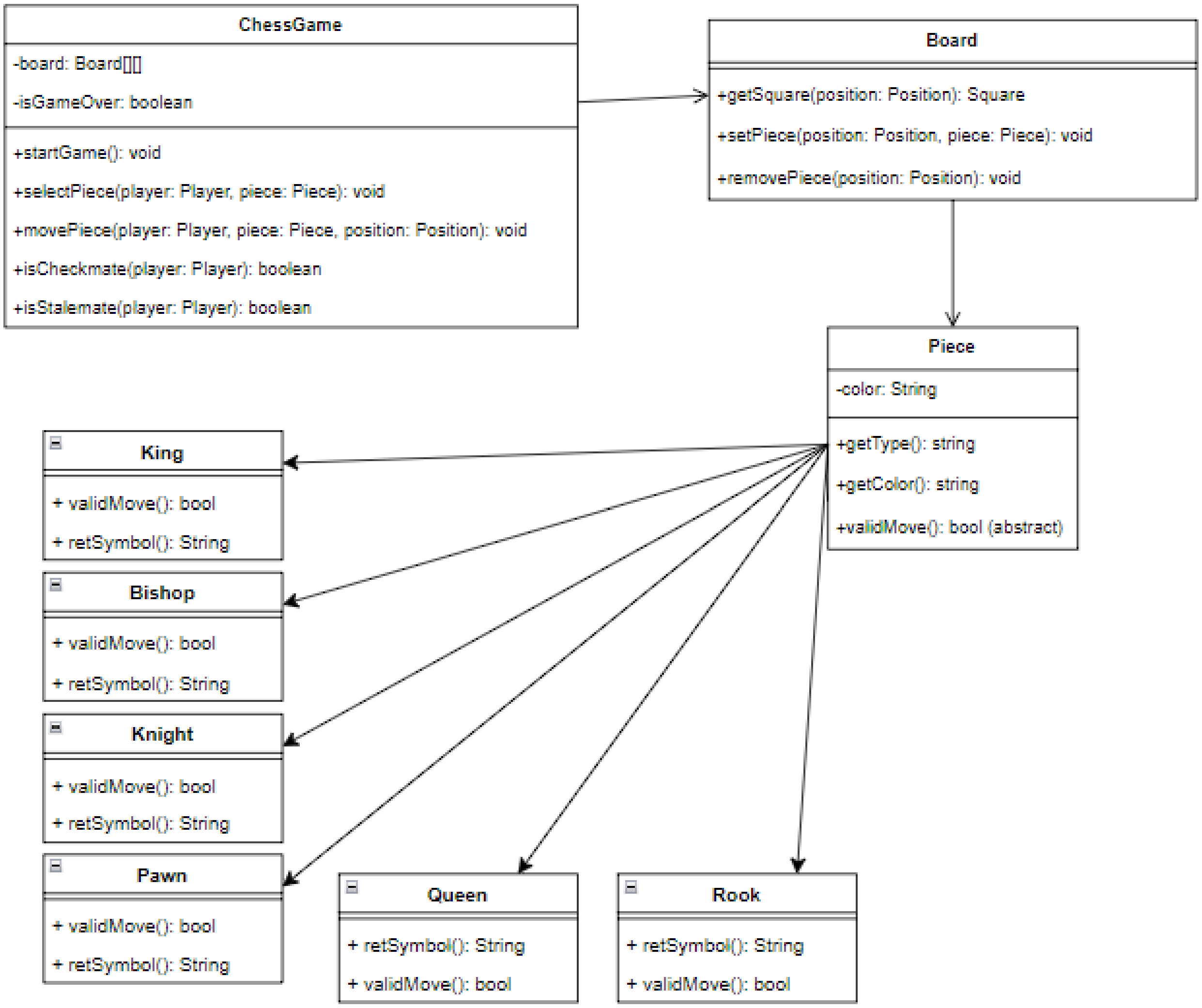
2.1. Existing Systems and Feasibility

Our chess game project enters a space with several existing digital chess applications, ranging from basic implementations accessible through web browsers to sophisticated platforms like Chess.com and Lichess that offer extensive features including AI opponents, online multiplayer, and tutorials. These systems demonstrate the feasibility of creating engaging and functional chess games. By focusing on core functionalities such as a solid chess engine, a user-friendly interface, and potentially a basic AI for single-player mode, our project is highly feasible within the course's timeframe. Leveraging C# and its robust .NET framework allows us to efficiently handle game logic and GUI development. It stands as a viable project with room for future expansion, providing a practical learning experience in software development and an opportunity to fill niches

2.2. Proposed Solution

Our proposed chess game project, tailored in C#, is focused on delivering an essential yet immersive chess playing experience. The project is scoped to include the foundational elements necessary for engaging gameplay: implementing the core mechanics of chess, such as piece movements, turn-based gameplay, and recognition of check and checkmate scenarios; developing a user-friendly graphical user interface (GUI) that displays the chessboard and pieces, supports mouse or touch inputs for piece movements, and enhances the overall player interaction; and integrating basic save/load functionality, allowing players to save their progress and resume play at their convenience. This concise approach ensures that our project remains practical and achievable within the course timeframe, laying the groundwork for a solid chess game that could be expanded in the future with additional features.

2.3. Barebone Diagram



2.4. Technology

Describe the technology that will be used. For example, what language, libraries, frameworks, etc. are used in this project, and how will they be used.

For our chess game project, we will primarily utilize C# as the programming language due to its suitability for desktop application development and its compatibility with the .NET framework. Additionally, we will leverage the following libraries and frameworks:

Unity Game Engine: Unity will serve as the primary development environment, providing powerful tools for building interactive 2D applications. We'll use Unity's GameObject system to manage game entities, including the chessboard and pieces, and to handle user inputs and interactions.

.NET Framework: As a comprehensive and widely-used framework for building Windows applications, .NET will provide essential functionality for our project, including data access, networking, and user interface development.

Unity UI Toolkit: Unity's built-in UI toolkit will enable us to create a user-friendly interface for our chess game, allowing players to interact with the game board, manage settings, and access additional features seamlessly.

Chess Engine Library: We'll incorporate a chess engine library written in C# to handle the core game mechanics, such as legal move generation, board evaluation, and game state management. This library will form the backbone of our game's logic, ensuring accurate and consistent gameplay.

Several resources are available to support our development efforts with the chosen technologies:

Unity Documentation: Unity provides comprehensive documentation, tutorials, and guides for learning the engine and its features. This includes resources specifically tailored to 2D game development, which will be valuable for our chess game project.

C# Programming Resources: Numerous online tutorials, books, and courses are available for learning C# programming. Websites like Microsoft Learn, Codecademy, and Coursera offer beginner to advanced-level courses covering C# fundamentals and advanced topics relevant to desktop application development.

Chess Engine Development Guides: Several resources are available online for learning about chess engine development in C#. Websites like GitHub repositories, Stack Overflow, and dedicated forums provide code examples, discussions, and tutorials on implementing chess engines and game logic.

2.5. Deliverables

At the end of the term, we will provide the following deliverables:

Source code: Complete source code of the chess game project, including all scripts, assets, and configuration files.

Installation instructions: Detailed instructions on how to set up and run the chess game on Windows platforms, including any prerequisites such as the .NET framework.

User documentation: A user manual explaining how to play the game, navigate the interface, and access various features.

Technical documents: Design documents outlining the architecture, functionality, and implementation details of the chess game project.

Bug report: A comprehensive report detailing any known issues, bugs, or limitations in the current version of the game, along with suggestions for future improvements.

Demo video: A demonstration video showcasing the gameplay and features of the chess game project.

3. Stage Plan

1. Proposal 11:30pm February 14, 2024

   -Research existing chess game applications and identify key features.
   -Define the scope and objectives of the chess game project.
   -Outline the proposed solution, including core features and technologies.
   -Identify potential user groups and their needs.
   -Create a high-level project plan, including timelines and deliverables.
   -Prepare and submit the project proposal document by 11:30pm on February 14, 2024.

2. Coding 1: Design, Prototype, Testing Plan *TBA*

   -Develop a detailed design document outlining the architecture and components of the chess game.
   -Create wireframes or mockups of the user interface for the chess game.
   -Prototype key functionalities, such as piece movement and game logic, to validate the design approach.
   -Define a testing plan, including unit tests and integration tests, to ensure the quality and reliability of the code.
   -Review and refine the design, prototype, and testing plan based on feedback from peers or instructors.
   -Complete coding tasks related to the initial implementation of the chess game.
   -Aim to have this stage completed by *TBA*.

3. Coding 2: Reviewed Design, Coding, Test Results *TBA*

   -Implement the reviewed design and incorporate any necessary changes or adjustments.
   -Continue coding tasks to build out additional features and functionalities of the chess game.
   -Conduct thorough testing of the implemented features, including functional testing, regression testing, and user acceptance testing.
   -Document test results and identify any bugs or issues encountered during testing.
   -Review the overall progress of the project and make adjustments to the timeline or scope as needed.
   -Prepare for the final presentation by organizing test results, bug reports, and other relevant documentation.
   -Aim to have this stage completed by *TBA*.

4. Final Presentation, Test Results, Bug Report *TBA*

   -Prepare a comprehensive final presentation showcasing the completed chess game project.
   -Present the project to peers, instructors, or stakeholders, highlighting key features, design decisions, and technical achievements.
   -Provide an overview of test results, including successful implementations and any identified bugs or issues.
   -Submit a detailed bug report documenting all known issues, along with suggestions for resolution or mitigation.
   -Reflect on the project's successes and challenges, discussing lessons learned and areas for future improvement or expansion.
   -Complete any remaining tasks, such as finalizing documentation or addressing last-minute issues.
   -Aim to have all deliverables submitted by *TBA*, to meet the project deadline.

4. Users

Casual Players: These users are primarily interested in playing chess for leisure and enjoyment. They may have varying levels of experience with the game, ranging from beginners to intermediate players. Casual players seek a user-friendly interface, intuitive controls, and the ability to play against AI opponents or friends in a relaxed environment. They may also appreciate features such as save/load functionality for interrupted games and customizable game settings to tailor the experience to their preferences.

Chess Enthusiasts: This group consists of users who are passionate about chess and may have a deeper understanding of the game's strategies and tactics. Chess enthusiasts often seek challenging opponents and opportunities to improve their skills through practice and analysis. They may value features such as adjustable difficulty levels for AI opponents, support for chess notation to review and analyze games, and access to educational resources such as tutorials or puzzles to enhance their gameplay experience.

Educators and Students: Teachers, chess coaches, and students constitute another user group with specific needs related to chess education and training. Educators may use the chess game project as a teaching tool in classrooms or chess clubs to demonstrate game principles, facilitate practice sessions, and engage students in interactive learning activities. Features such as customizable lesson plans, built-in tutorials, and support for multiplayer mode for group competitions can cater to the needs of educators and students alike.

Developers and Enthusiasts: This group includes individuals interested in game development, programming, or chess engine design. Developers may use the chess game project as a learning resource to study game development techniques, explore Unity and C# programming, or experiment with AI algorithms for chess. Features such as well-commented code, modular architecture, and integration with external libraries or APIs can facilitate learning and experimentation for developers and enthusiasts.

5. Risks

Realistic factors or developments that could prevent us from delivering a successful project:

Risk	Description of resolution	Result / Decision	Ref to file(s)	Status
Time Constraints	Prioritization of features; agile development to accommodate changes	Features prioritized; agile sprints planned	Proposal	Underway
Technical Complexity	Allocate additional time for complex features	Complex features simplified	Proposal	Underway
Team Coordination	Regular team meetings and use of project management tools.	Improved communication and task tracking	Proposal	Underway

6. Technical Issues

Describe the technical issues that you encountered at this stage. This table will be moved in a separate file updated during the project.

Description of Problem	Description of work	Result	Ref to file(s)	Status
Issues with Unit Testing	Was not able to implement detailed Unit Testing of various functions in the class due to errors	Incomplete unit tests	ChessGameProject Tests class	Underway