**A project report**

**On**

**Chess**

Submitted in partial fulfillment of the requirement of Project – II (BCA179CO)

Of

Bachelor of Computer Application 2nd Semester

**Submitted to**



Purbanchal University

Biratnagar, Nepal

**Submitted By**

Milan Shrestha (325489)

Bishal Subedi (325483)

Benjin Lama (325480)

**KANTIPUR CITY COLLEGE**

Putalisadak, Kathmandu

May, 2021

**A project report**

**On**

**Chess**

Submitted in partial fulfillment of the requirement of Project – II (BCA179CO)

Of

Bachelor of Computer Application 2nd Semester

**Submitted to**

Purbanchal University

Biratnagar, Nepal

**Submitted By**

Milan Shrestha (325489)

Bishal Subedi (325483)

Benjin Lama (325480)

**Project Supervisor**

Mr. Bikash Neupane

Assistant Professor

**KANTIPUR CITY COLLEGE**

Putalisadak, Kathmandu

May, 2021

**CERTIFICATE OF APPROVAL**

The undersigned certify that they have read and recommended to the Department of computer in application for acceptance, a project report entitled “Chess” submitted by

Milan Shrestha (325489)

Bishal Subedi (325483)

Benjin Lama (325480)

In fulfillment for the degree of Bachelor in computer application

.............................

Bikash Neupane

Project Supervisor

Department of IT

.............................

Saroj Pandey

Head of Department

Department of IT

ACKNOWLEDGMENT

We would like to show our sincere gratitude to supervisor Mr. Bikash Neupane for providing continuous guidance and support throughout the project. Without his kind direction and help, this project would not have come together. In every phase of the project, he was of great help and thus has shaped this project to be completed marvelously.

Completion of any project requires a great deal of cooperation between a number of people. We have taken help from several people for the preparation of this project. So we would like to thank Mr. Saroj Pandey (HOD) and all the other supervisors for their help. To them, we would like to say- thank you for spending all those hours answering our queries and providing us with the necessary sources to carry out this project. Last but not least we would like to express our appreciation to all our friends for their valuable comments and suggestion which gave up inspiration to better our project. We thank all the people for their help directly and indirectly to complete our assignment.

ABSTRACT

Chess is a well know game that has been in existence since middle age and is one of the most entertaining games that challenge the player whilst providing entertainment. For this project chess game was chosen and was successfully built with the help of various features made available by object-oriented programming. Features such as inheritance, polymorphism, dynamic binding, class and object, message passing were hugely beneficial for the advancement of this project.

For programming SFML (Simple and Fast Multimedia Library) was used. It provides an interface to the various components of your PC, to ease the development of games and multimedia applications. It is composed of five modules: system, window, graphics, audio, and network.

With the help of this library, the team was able to create a GUI interface for the user and the chess engine so that we could have a chess game against A.I.

For ai, the team made use of one of the most popular chess engines, Stock fish. This engine provided us with the ability to have an A.I to play against the user and generate the moves it deemed appropriate.

For linking the chess engine and the GUI-based C++ interface the team wrote "Connector.CPP" to link these two units together. This acted as a universal chess interface between the two units. For writing, this UCI protocol was made use of as this is one of the most supported formats for creating an interface.

**Table of Content**

[ACKNOWLEDGMENT i](#_Toc61693835)

[ABSTRACT ii](#_Toc61693836)

[LIST OF FIGURES iii](#_Toc61693837)

[LIST OF TABLES iv](#_Toc61693838)

[ABBREVIATIONS v](#_Toc61693839)

[Chapter 1: INTRODUCTION 1](#_Toc61693840)

[1.1 Background 1](#_Toc61693841)

[1.2 Problem Definition 1](#_Toc61693842)

[1.3 Objectives 1](#_Toc61693843)

[1.4 Significance 2](#_Toc61693844)

[1.5 Features 2](#_Toc61693845)

[1.6 Team Structure 2](#_Toc61693846)

[1.7 Documentation Organization 3](#_Toc61693847)

[Chapter 2: System Analysis 4](#_Toc61693848)

[2.1 Requirement gathering process 4](#_Toc61693849)

[2.2 Feasibility Study 4](#_Toc61693850)

[Fig 2.3 A planned Gantt chart 6](#_Toc61693851)

[Chapter 3: System Design and Development 7](#_Toc61693852)

[3.1 Functional Analysis 7](#_Toc61693853)

[3.2 Algorithm 8](#_Toc61693854)

[3.3 Flowchart Fig 2.2 Flowchart of the program 10](#_Toc61693855)

[Chapter 4: System Development and Implementation 11](#_Toc61693856)

[4.1 Programming platform 11](#_Toc61693857)

[4.2 Test plan 11](#_Toc61693858)

[Chapter 5: Conclusion and Future Enhancement 15](#_Toc61693859)

[5.1 Conclusion 15](#_Toc61693860)

[5.2 Limitations 15](#_Toc61693861)

[5.3 Future Enhancement 15](#_Toc61693862)

[References 16](#_Toc61693863)

[Appendices 17](#_Toc61693864)

LIST OF FIGURES

|  |  |  |
| --- | --- | --- |
| **Figure Number** | **Figure** | **Page No.** |
| 2.2 | Example of a planned Gantt chart | 5 |
| 3.3 | Flowchart | 9 |

LIST OF TABLES

|  |  |  |
| --- | --- | --- |
| **Table Number** | **Table** | **Page No.** |
| 1.6 | Team Structure | 2 |
| 1.7 | Documentation Organization | 3 |
| 2.1 | Requirement gathering process | 4 |
| 2.12 | Schedule | 5 |
| 3.1 | Functional Analysis | 6 |

ABBREVIATIONS

|  |  |
| --- | --- |
| A.I | Artificial Intelligence |
| IT | Information Technology |
| IDE | Integrated Development Environment |
| SFML | Simplified Fast Multimedia Library |
| GUI | Graphical User Interface |

Chapter 1: INTRODUCTION

## 1.1 Background

## 

## Chess is a GUI-based application created using C++ language and makes use of SFML libraries to help achieve multimedia accessing. Through this library, various multimedia such as texture, sprites were made accessible. The aim of this application is to provide an entertaining way to play a game of chess against the 2nd player or one of the best A. I chess engine- stockfish. Various textures were used to create the board and the chess pieces, which were sourced from Wikipedia. For the pieces and moves, standard notations were made use of. Various chess concepts such as En passant, casting, promoting have been incorporated into this application. Strings were used to store the relative locations of the chess pieces and they were converted into their corresponding integer character by extracting string and using the conversion function to calculate space in world space relative to screen position. Lots optimizations were made to ensure that each frame had as few resources to load and hence making each frame render as fast as possible using as little computer memory as possible.

## 1.2 Problem Definition

## 

## Chess is an entertainment-based application. Hence for maximizing that aspect the application has to run smoothly and properly. This application had to be intuitive so symbols in the form of sprites were used to indicate chess pieces. The board is the standard chessboard. For the application to run smooth, i.e. high frame rate, each frame was loaded so that a minimum number of refresh would be required.

## 1.3 Objectives

The main objectives of this project is

* To implement GUI based application that provides entertainment to the end-user in smooth and efficient manner

## 1.4 Significance

* This project provides a gateway to integrating third-party libraries into the C++ programming language
* This project provides amusement for the parties interacting with the application
* This project allows for improvement in the player's skill by playing against the A. I and being able to analyze the game

## 1.5 Features

Some features of this program are:

* It offers a visually pleasing graphical interface
* It allows reverting the currently made move
* It allows for both single-player and multiplayer mode
* Allows analyzing the game against an A. I engine
* Advance concepts such as En passant and casting are allowed

## 1.6 Team Structure

|  |  |  |  |
| --- | --- | --- | --- |
| **S.N** | **Members Name** | **Symbol Number** | **Task Functions** |
| **1** | **Milan Shrestha** | 325489 | Designing, Coding and Documentation |
| **2** | **Bishal Subedi** | 325483 | Designing, Coding and Documentation |
| **3** | **Benjin Lama** | 325480 | Designing, Coding and Documentation |

## 1.7 Documentation Organization

|  |  |  |
| --- | --- | --- |
| Chapters | Heading | Contains(Topics) |
| Chapter 1 | Introduction | * 1. Background   2. Problem Definition   3. Objectives   4. Significance   5. Features   6. Team Structure   1.5 Documentation Organization |
| Chapter 2 | System Analysis | 2.1 Functional Requirements  2.2 Team structure and roles  2.3 Implementation Plan  2.4 User Defined Function  2.5 Data Structure  2.6 File Structure |
| Chapter 3 | Software Design and Development | 3.1 Function Analysis  3.2 Algorithm  3.3 Flowchart |
| Chapter 4 | System Development and Implementation | 4.1 Programming platform  4.2 Test plan  4.3 Implementation and result analysis |
| Chapter 5 | Conclusions | 5.1 Conclusion  5.2 Limitations  5.3 Future Enhancements |

Chapter 2: System Analysis

## 2.1 Requirement gathering process

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Requirement No. | Requirement Name | Requirement description | Process of gathering | Function number |
| 1 | loadPositon | Loads the position of the chess pieces | Discussion | 1 |
| 2 | VideoMode | Function that load the window in screen space | Team Discussion | 2 |
| 3 | loadFromFile | Functions that allows the loading of texture from file | Prototyping | 3 and 4 |
| 4 | window.pollEvent() | Polls whether any event has occurred or not in the screen space | Workshop | 5 |
| 5 | getNextMove | Gets the next move from the chess engine | Story board | 6 |

## 2.2 Feasibility Study

Firstly, for creating this project it was taken into account that a GUI element was required and hence external library was used. For the library, SFML was chosen. It was chosen because it offered better documentation and provided plenty of meaningful information need to get started with the project for relative newcomers.

Visual Studio was the IDE of choice, it was chosen because of its proper integration and accessibility is provided to make use of external libraries. Visual Studio also allowed for better coding, as it provided necessary shortcuts and an advanced compiler with debugging support.

This program required there to be multiple texture loading and unloading of each frame and hence require a strong computer system to run it.

Windows 10 was used as Operating System.

Microsoft Word was used to create the documentation.

##### 2.2.1 Resource feasibility

Resources that are required for this project include:

* Programming devices
* Programming tool
* Programming individuals

##### 2.2.2 Schedule

|  |  |  |
| --- | --- | --- |
| Activity | Start Week | End Week |
| Communication | 02/01/2021 | 09/01/2021 |
| Planning | 10/01/2021 | 20/01/2021 |
| Design | 01/02/2021 | 22/02/2021 |
| Development | 23/02/2021 | 08/04/2021 |
| Testing | 10/04/2021 | 18/04/2021 |
| Deployment | 18/04/2021 | 20/04/2021 |

Project was completed within the allocated time. Project deadline was set accordingly so that it matched our skill set. Functions were divided among team members and each of the members were able to come within a week of margin of error and submit their respective tasks on time.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Activities** | **No. Of Weeks** | **Plan/Actual** | **January** | | | | | | **February** | | | | **March** | | | | **April** | | | |
| 1 | 2 | 3 | 4 | 1 | 2 | | 3 | 4 | 1 | | 2 | 3 | 4 | 1 | 2 | 3 | 4 |
| Communication | 1 weeks | Plan |  |  |  |  |  |  | |  |  |  | |  |  |  |  |  |  |  |
| Actual |  |  |  |  |  |  | |  |  |  | |  |  |  |  |  |  |  |
| Planning | 2 weeks | Plan |  |  |  |  |  |  | |  |  |  | |  |  |  |  |  |  |  |
| Actual |  |  |  |  |  |  | |  |  |  | |  |  |  |  |  |  |  |
| Design | 3 weeks | Plan |  |  |  |  |  |  | |  |  |  | |  |  |  |  |  |  |  |
| Actual |  |  |  |  |  |  | |  |  |  | |  |  |  |  |  |  |  |
| Development | 6 weeks | Plan |  |  |  |  |  |  | |  |  |  | |  |  |  |  |  |  |  |
| Actual |  |  |  |  |  |  | |  |  |  | |  |  |  |  |  |  |  |
| Testing | 1 weeks | Plan |  |  |  |  |  |  | |  |  |  | |  |  |  |  |  |  |  |
| Actual |  |  |  |  |  |  | |  |  |  | |  |  |  |  |  |  |  |
| Deployment | 1 weeks | Plan |  |  |  |  |  |  | |  |  |  | |  |  |  |  |  |  |  |
| Actual |  |  |  |  |  |  | |  |  |  | |  |  |  |  |  |  |  |

## Fig 2.3 A planned Gantt chart

Chapter 3: System Design and Development

## 3.1 Functional Analysis

|  |  |  |  |
| --- | --- | --- | --- |
| Function No. | Function Name | Function description | Requirements covered |
| 1 | toChessNote | Converts 2 dimensional vector into string |  |
| 2 | toCoord | Converts string into 2 dimensional vector |  |
| 3 | move | Checks the condition whether or not the move is allowed, if allowed moves |  |
| 4 | ConnectToEngine | Connects the GUI with the chess engine |  |
| 5 | getNextMove | Gets the next move from the chess engine |  |
| 6 | window.isOpen() | Returns a true value if the window is opened |  |
| 7 | window.pollEvent(e) | Returns true is any event hass occured |  |
| 8 | getGlobalBounds().contains(pos.x,pos.y) | Return a x and y coordinate of the mouse |  |
| 9 | getPosition | Gets the position of the sprite |  |
| 10 | isKeyPressed | Checks whether or not the key has been pressed |  |
| 11 | setTextureRect | Sets the value of each texture to the sub set of original texture |  |

## 3.2 Algorithm

STEP 1: START

STEP 2: DISPLAY MAIN MENU

STEP 3: ASK FOR USER CHOICE

STEP 4: IF CHOICE IS 1

STEP 4.1: DISPLAYS STUDENT MENU AND ASK FOR USER CHOICE

STEP 4.2: IF CHOICE IS 1

STEP 4.2.1: LET USER ENTER ID AND DISPLAY THE CORRESPONDING STUDENT RECORD

STEP 4.3: IF CHOICE 2

STEP 4.3.1: LET USER ENTER FIRST NAME AND DISPLAY THE CORRESPONDING STUDENT RECORD

STEP 4.4: IF CHOICE 3

STEP 4.4.1: LET USER ENTER LAST NAME AND DISPLAY THE CORRESPONDING STUDENT RECORD

STEP 4.5: IF CHOICE 4

STEP 4.5.1: LET USER ENTER FIELD AND DISPLAY THE CORRESPONDING STUDENT RECORD

STEP 4.6: IF CHOICE 5

STEP 4.6.1: LET USER ENTER SEMESTER AND DISPLAY THE CORRESPONDING STUDENT/S RECORD

STEP 4.7: IF CHOICE 6

STEP 4.7.1: DISPLAY ALL EXISTING RECORD OF THE STUDENTS

STEP 5: IF CHOICE IS 2

STEP 5.1: DISPLAY ALL EXISTING RECORD OF THE T TEACHERS

STEP 6: IF CHOICE IS 3

STEP 6.1: DISPLAY THE ROUTINE OF CURRENT DATE

STEP 7: IF CHOICE IS 4

STEP 7.1: DISPLAY ALL EXISTING NOTICE

STEP 8: IF CHOICE IS 5

STEP 8.1: DISPLAY EDIT MENU AND ASK FOR USER CHOICE

STEP 8.2: IF CHOICE IS 1

STEP 8.2.1: LETS USER CREATE NEW STUDENT RECORD

STEP 8.3: IF CHOICE IS 2

STEP 8.3.1: LETS USER CREATE NEW TEACHER RECORD

STEP 8.4: IF CHOICE IS 3

STEP 8.4.1: DISPLAY THE ROUTINE AND ASK FOR USER TO SELECT PERIOD

STEP 8.4.2: ASK USER FOR NEW SUBJECT NAME

STEP 8.5: IF CHOICE IS 4

STEP 8.5.1: LET USER CREATE NEW NOTICE

STEP 8.6: IF CHOICE IS 5

STEP 8.6.1: ASK USER FOR STUDENT ID

STEP 8.6.2: ASK USER FOR CORRESPONDING STUDENT'S GPA

STEP 8.7: IF CHOICE IS 6

STEP 8.7.1: ASK USER FOR STUDENT'S ID AND DELETE THE CORRESPONDING RECORD

STEP 8.8: IF CHOICE IS 7

STEP 8.8.1: ASK USER FOR TEACHER'S ID AND DELETE THE CORRESPONDING RECORD

STEP 9: IF CHOICE IS 6

STEP 9.1: DISPLAY ALL THE EXAMINATION RESULT OF EXISTING STUDENTS

STEP 10: IF CHOICE IS 7

STEP 10.1: Stop

## C:\Users\milan\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Untitled Diagram.jpg3.3 Flowchart Fig 2.2 Flowchart of the program

Chapter 4: System Development and Implementation

## 4.1 Programming platform

Programming platform use for this project was C++ programming and ide Visual Studio.

## 4.2 Test plan

Since this program is really long there was a lot of trial and error required to make sure that bugs were removed and the program was made completely function

##### 

##### 4.2.1 Introduction

Waterfall methodology was used to create the program. And unit testing, integration testing, and system testing were conducted to ensure their proper risk management.

**Objective of Testing**

Some objectives of testing project were

* Ensure the Application Under Test conforms to functional and non-functional requirements
* Ensure the complete system performed as per the main objective of the project
* Bugs/issues are identified and fixed before final submition

**Roles and Responsibility**

Detail description of the Roles and responsibilities of different team members like

* Test Manager – Milan Shrestha
* Unit testing – Benjin Lama
* Integration testing – Bishal D Subedi
* System testing – Milan Shrestha

Amongst others

##### 4.2.2 Testing Methodology

Firstly the main data types were distinguished for the application. Then those were divided up for the team members to work on, and properly all the separate units were incorporated into the main projects. Each team member tested their own part of the code in the context of the whole application. And hence this approach allowed for the bugs to be solved relatively quickly. Then finally integration process was carried out and the whole unit was tested at once.

For testing we carried out tests such that defect clustering was avoided, risk assessments were done to make sure that exhaustive testing would not have been necessary.

Finally, system testing was conducted to ensure that the program was working as a well-rounded complete package. The functionality was tested from end-to-end before submitting the program for presentation.

**Bug Trials**

The goal of the trial is to

* To define the type of resolution for each bug
* To prioritize bugs and determine a cluster defects

##### 4.2.3 Test Completeness

A few criteria to check Test Completeness of this project were

* 100% test coverage
* All Manual test were executed
* All open bugs were fixed

##### 4.2.4 Resource and Environment Needs

**Testing tools**

Testing was done manually to ensure there was proper coordination between the team members and was done online by using GitHub. GitHub allowed for proper communication and provided with the progress of fellow team member to be noted, and also allowed for proper integration of the software.

**Testing Environment**

Following software’s were used for conducting testing

1. Windows 10
2. Visual Studio
3. Stockfish

##### 4.3 Implementation and result analysis

After multiple iterations with all the bugs eradicated finally the following windows was created.



Chapter 5: Conclusion and Future Enhancement

## 5.1 Conclusion

Chess is an entertaining and engaging game that challenges the chess player of any level. It makes use of various features of objected-oriented programming and libraries which would not have been possible by just using the C language. As it is the improved version of C it allowed for more versatility and allowed access to more complex arrangements to create a well-rounded application.

This project required a great deal of reading when it came to using various never-used libraries. SFML provided very easy to grasp documentation, other libraries were tried for this project but none offered the versatility and the simplicity that SFML did.

With access to libraries, various new implementations such as texture, sprites, decals were accessible. These allowed for a better GUI interface and easier implementation for difficult calculations.

The chess engine provided us with the A. I and was relatively simple to use as soon as it was figured out how to write the Universal Chess Interface. Which was learned by reading documentation provided by the GitHub post from the creator of the stockfish Chess engine.

This project introduced us to various aspect of deploying and cohesive teamwork which allowed us to access and communicate code online medium and collaborate to create a viable project. By keeping track of each other progress and communicating back and forth this project was made successful.

## 5.2 Limitations

The application loads new texture after each frame so for non-powerful devices running the operation might be very difficult. Some complicated logics such as castling and En passant require multiple interactions and logic checking hence taking a lot of time to process

## 

## 5.3 Future Enhancement

## In the future, we want to add an option to have two competing A.I to battle against each other or even perhaps have an option for the user to make use of A.I when playing against another A.I, but only when needed. Saving the current state of the game and being able to choose the difficulty level could also be implemented. A better library could be used to create more sprite animations and improve the visual quality of the overall application.

# **References**

Torres, JC (2006). UCI protocol for chess variants.

Ang, Bobby (1999). Business World Online, p. 2.

Young, R. R. (2006, 4 12). Description of the universal chess interface (UCI), p. 1.

Appendices



This images were took as a screenshots from the game’s window.

