**ARTIFICIAL INTELLIGENCE**

****

**PROJECT REPORT**

**PROJECT TITLE:**CHESS GAM PYTHON

**SUBMITTED BY:**Rahul Sahu(5)

Ashish Sharma(37)

Aditya Shekhar(49)

Rajesh Kumar Gupt(51)

**SUBMITTED TO:**Miss.Shabnam Mam

GithubLink: <https://github.com/RahulSahu7957/chess-game-in-python1>

**CONTENTS**

**S. No TOPICS Page No.**

**1. INTRODUCTION ........................................................................ 6**

1.1. SCOPE ................................................................................. 6

1.2. OBJECTIVES ........................................................................... 7

1.3. SYSTEM ANALYSIS AND DESIGN ................................................... 8

1.3.1 DEVELOPING A SOLUTION STRATEGY ........................................ 8

1.3.1.1. FEASIBILITY STUDY ............................................... 8

1.3.1.2. ANALYSIS ........................................................... 9

1.3.1.3. SOLUTION STRATEGY ............................................. 9

1.3.2. DESIGN ......................................................................... 9

1.3.3. TESTING ........................................................................ 9

1.4. CHESS GAME: AN INTRODUCTION ............................................... 11

1.4.1. OBJECTIVE OF THE GAME .................................................... 12

1.4.2. THE PIECES AND THE RULES ............................................... 12

1.4.3. SOME SPECIAL MOVES AND CASES ......................................... 16

1.5. IRC AND WEB CHAT: AN INTRODUCTION ....................................... 18

**2. OVERVIEW ON PROGRAMMING LANGUAGE USED .................... 20**

2.1. JAVA: AN INTRODUCTION .............................................................. 20

2.1.1. FEATURES ..................................................................... 23

2.1.2. CLIENT-SERVER ARCHITECTURE ............................................ 31

2.1.3. DEFINITIONS AND ACRONYMS .............................................. 40

**3. HARDWARE AND SOFTWARE REQUIREMENTS ........................ 47**

**4. GUIDE ON USING THE APPLET ................................................ 48**

4.1. GUIDE TO USE THE CHESS APPLET ............................................... 48

4.1.1. GETTING FAMILIAR WITH PIECES ........................................... 48

4.1.2. HOW TO MAKE MOVE? ....................................................... 48

4.1.3. HOW TO CAPTURE OPPONENT PIECE? ...................................... 48

4.1.4. HOW TO RECONSIDER A PIECE TO MOVE? ................................. 49

4.1.5. HOW TO PERFORM A CASTLING? ............................................ 49

4.1.6. HOW TO PERFORM EN-PASSANT? ........................................... 49

4.2. GUIDE TO USE THE CHAT APPLET ................................................. 49

4.2.1. HOW TO SEND MESSAGE? ................................................... 49

**5. DATA FLOW DIAGRAMS ......................................................... 50**

**6. PROJECT DETAILS ................................................................... 53**

6.1. ENHANCEMENTS .................................................................... 53

6.2. PROBLEMS FACED .................................................................. 54

6.5. SCREENSHOTS ...................................................................... 79

**1. Introduction:**

The aim of the project is to provide both a chess as well as

chat facility in one application such that two users can play the

chess against each other as well as they can chat among

themselves on the network that is any number of computers

connected via a network. So the users can play the chess in

pair and in the meantime when any one of them wants to

communicate with the opponent player, he can also do that

simultaneously.

**1.1. Scope :**

Basically this project has two main purposes:

**Firstly**, to encapsulate the chess gaming and chatting in one

application so that a user can easily view and use both of them on a

single window.

**Secondly**, to make these two applications to run simultaneously

on the network. Generally, the chess game is normally played against

the human user and the computer on which the user is operating, so

the need is to make the chess game being played against two human

users on two different machines on the network, no matter their

physical location and as well as allowing chat among those two users.

**1.2. Objectives :**

The objectives of this project are:

a) To provide a user-friendly interactive environment to the

users of the application that helps them to play and

communicate with a lot of ease.

b) To provide help to the users in playing the chess that is the

different moves of the different pieces etc are being

explained to the users, if they require.

c) The care is taken that the user finds the same chatting

mechanism as he is normally used to.

d) Since there exists client and server as the project is based

on client server architecture, where server is serving as a

mediator in between the players and the client is making

request to server as well as doing all the part that is related

to playing logic.

e) The care has been taken that the application has less CPU

usage, so that other applications can also be performed, if

required.

**1.3. System Analysis and Design:**

**1.3.1. Developing a Solution Strategy:**

Several solution strategies were outlined with regular

constants. Conducting feasibility study for each strategy.

Recommending a solution strategy. The steps incorporated in

attempting the solution were divided into three phases:

**1.3.1.1. Feasibility Study:**

Once the program area or solution strategy was

identified or set of feasibility study was undertaken to

check the validity of available resources. To do the

study we need to consider the

a) Technical Feasibility,

b) Economical Feasibility and

c) Behavioural Feasibility.

**a) Technical Feasibility:**

It is combination of the Software and Hardware

requirements. The Hardware requirements comprise of

Pc's with minimal requirement and configuration

enough to support Graphical User Interface and

network establishment (LAN) or Internet available and

2 Button Mouse. The Software requirements are Java

Compatible Web Browser and network and Sockets

Port.

**Requirement Analysis and Specifications:**

The different requirements of the project are:

1. A Chess board and its layout & design.

2. The Chess pieces design & moves generation

technique and logics.

3. The Chat Applet layout & design.

4. Networking programming logic implementation

using Applet designed earlier.

5. Synchronization between the Chess Applet and

Chat Applet.

**b) Economical Feasibility:**

The cost requirement to establish the system

should be effective. Cost/Benefit Analysis is made to

find out projected Costs and Benefit for successful

development of the Application.

**SRS Documents:**

The different requirements are carefully

scrutinized, the moves planned, layout framed and

every changes move or message is communicated in

real time.

**1.3.1.2. Analysis:**

It is the detailed study of the various operations

or moves of the system. The key question is what

must be done to solve the problem.

**1.3.1.3. Solution Strategy:**

The solution strategy is to develop a layout that

comprises of both the chess gaming board and chat

window in one frame layout, so that the users can view

and use both the application simultaneously. The

layout that found most useful was the Java Applet.

The other solution strategy was to make the

application run over different network ad that is done

using Java Applets running over the networks using

sockets (Socket and Port Programming). The Chesscum-

Chat Application to be used simultaneously is

made by embedding Chat Applet with the Chess Applet

and using network programming.

**1.3.2. Design:**

The most creative and challenging phase of the system life

cycle is system design. The design is a solution to develop that

know-how of the new system under consideration.

It provides the understanding and procedural details necessary

for implementing the system recommended in feasibility study.

Design phase identifies the boundary and interface of the

systems, format I/O etc.

**1.3.3. Testing:**

Testing is vital to the success of the system. System

testing makes a logical assumption that if all parts of the system

are correct the resulis will be successfully achieved. Effective

testing early in the process translates directly into long term

cost saving, reduced number of errors. System testing is done

when all the modules of the system are in working order and

has been tested independently for proper working . All the

pieces are put into one system and test to determined, whether

it needs user's requirements. The best program is worthless if

doesn't needs. System testing is designed to uncover weakness

that were not found in earlier tests like program testing in which

only syntactical and logical are removed. The purpose of System

Testing is to consider all the likely variations to which it will be

subjected and then push the system to its limits.

**What do you test for?**

1) The first test for a system is to see whether it produces the

correct outputs.

2) A volume test is done in which we create as many as records

as would normally be produced to verify that the hardware

and software will function properly.

3) Stress Testing is done to prove that the system doesn't

malfunction under peak loads. We subject the system to a

high volume of data over a short time period.

4) A forced system failure is included to test the backup

recovery procedure. Unaccurate data are entered to see the

response of the system in terms of error detections and

protections and to test that the data and programs are

secured from unauthorised access.

5) The usability test verifies the user-friendly nature of the

system. This related to natural operating and error handling

procedures.

6) Boundary value analysis is done to find out that the system

does'nt produce incorrect result, when the values provided

are not within the prescribed domain or at the boundaries of

the respective domains.

**Types of Testing**:

1) **Program Testing:**

It checks types of errors that is Syntax and Logic.

2) **String Testing:**

It is used to test a set of related programs forming a

subsystem.

3) **System Testing:**

It test the system as a whole that is, it checks the interrelationship

among the various subsystems.

4) **Alpha Testing:**

It tests the system in simulated environment on simulated

data.

5) **Beta Testing:**

It test the system under realistic conditions with actual

data.

6) **User Acceptance Testing:**

An acceptance testing has the objective of telling the user

about the validity and reliability of the system that is the

developed system is functioning according to the needs

prescribed by the user. This testing is done with the actual

data in presence of the user at the users place.

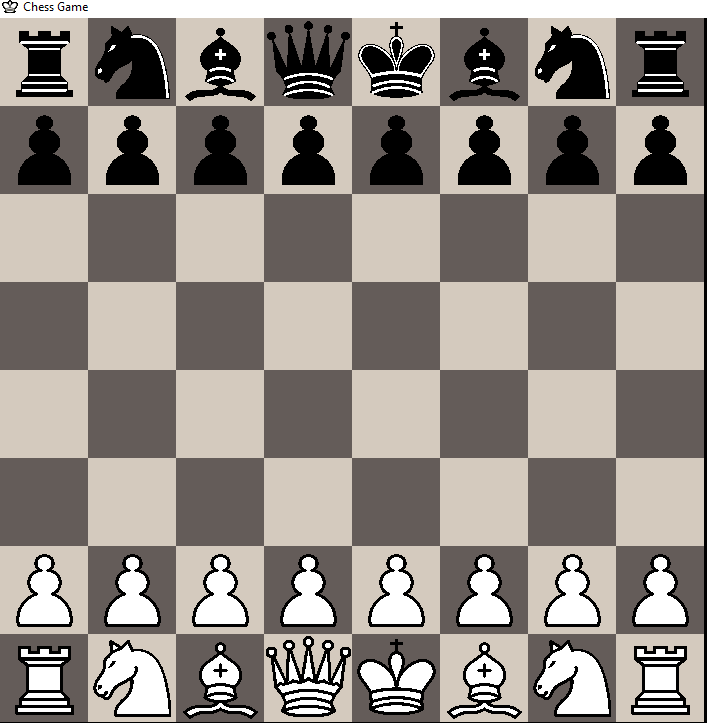
**1.4. Chess Game: An Introduction**

Chess is a game for two players using a chequered board of

sixty-four squares with eight pieces, viz. a king, queen, two

rooks, knights, and bishops, and eight pawns each, with the

object of placing the opponent's king in checkmate.



The game begins with the position shown above. The White

player (the player of the light colored pieces) moves first. Then

each player takes a single turn. In fact, a player must move in

turn. In other words a move cannot be skipped.

When setting up the pieces, keep in mind two things. The light

colored square goes on the player's right, and Queens go on

their color next to the Kings on the center files.

You may not move a piece to a square already occupied by one

of your own pieces. You may capture an opposing piece by

replacing that piece with one of your own pieces, if it can

legally move there.

**1.4.1. Objective of the Game:**

The primary objective in chess is to checkmate your

opponent's King. When a King cannot avoid capture then it is

checkmated and the game is immediately over. If a King is

threatened with capture, but has a means to escape, then it is

said to be in check. A King cannot move into check, and if in

check must move out of check immediately. There are three

ways you may move out of check:

1) Capture the checking piece;

2) Block the line of attack by placing one of your own pieces

between the checking piece and the King. (Of course, a

Knight cannot be blocked.);

3) Move the King away from check.

If a King is not in check, and no other legal move is possible,

then the position is said to be in stalemate. A stalemated game

is a draw, or a tie.

**3. Hardware & Software Requirements:**

The application must be capable of reflecting the changes

made to the chess board positions of the different pieces, when a

user makes a move or when some message is sent via a chat

application to other machine. All these changes must be reflected in

real time and without any delay that is according to the human

reaction time. It also needs to be multi-threaded to cater to the

needs of different users at the same time. Hence, the requirements

are a bit on the higher side.

1 It ideally requires a processor running at above 2 GHz

frequency.

2. It must be having 128 MB of RAM or higher.

3. The hard disk must be 4 GB .

4. The LAN or any other network should be established and LAN

card must be there on the machines where application will be

used

5.The pointing device must be an optical mouse.

6. The system runs effectively on Windows 2000 server but it

will also run equally well on compatible operating systems.

7. The web browser must be Microsoft Internet Explorer with a

resolution of at least 800 \* 600.

**4. GUIDE ON USING THE APPLET:**

4.1. Guide to use the Chess Applet:

The Chess Applet consists of 64 squares. With 32 squares

initially occupied by the pieces as discussed in the introductory

section. The Chess Apple is fully mouse oriented one. This guide has

been divided into various sections for user convenience. They are as

follows:

**4.1.1. Getting familiar with pieces**

****

**4.1.2. How to make move (It is a two-click process):**

The chess game is fully mouse oriented and key board is not

functional. The user can make a legal move by performing following

operations.

**First Click:** On the square containing the piece which the player

wants to move. (The player can move only his piece),

the square gets highlighted by RED colour.

**Second**

**Click:** The square where he want the piece to move.

**4.1.3. How to capture opponent piece:**

**First Click:**

On the square containing the piece which the player

wants to move. (The player can move only his piece),

the square gets highlighted by RED colour.

**Second**

**Click:**

The square containing the opponent piece which the

player wants to capture.

**4.1.4. How to reconsider a piece to move (Deactivating the**

**selected piece):**

Once the user as clicked on the piece to move and rethinks on

the piece to move, for this he can click on any of his other piece,

the highlighted square will be de-activated and he can start over

again.

Single Click on any of your own piece.

**4.1.5. How to perform a castling:**

To make any castling, whether o-o (short castling) or o-o-o

(long castling) the user has to perform following operation:

**First Click:** On the square containing the King.

**Second**

**Click:**

On the square where the king will be placed after

castling.

**4.1.6. How to perform an en-passant:**

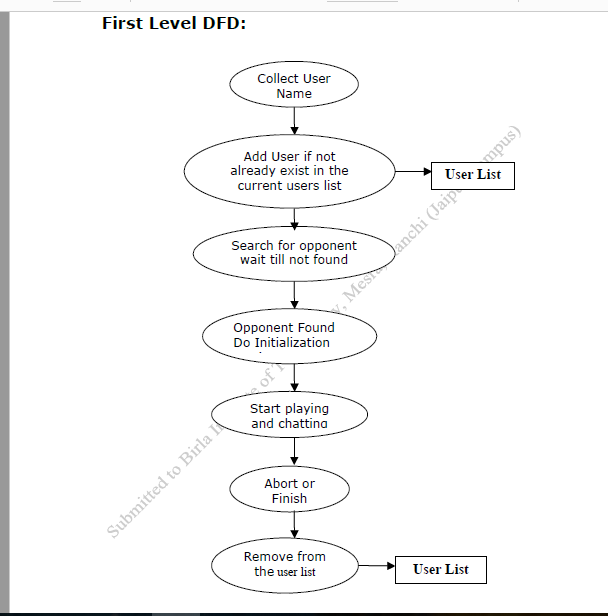
To achieve this, user has to perform following operation:

**First Click:** On the square containing the Pawn.

**Second**

**Click:** On the square skipped by the Opponent Pawn.

**5. DATA FLOW DIAGRAMS:**



**6.2. Problem Faced:**

1. Designing the board: whether to take a picture or drawing the

board using Graphics and java.awt classes

2. Move to be updated in opponents PC.

3. Alternatively changing turns of players.

4. A player to move only his pieces.

5. A white player has white at bottom and black having black.

6. Displaying nickname at appropriate place.

7. Pairing of players

8. Maintaining the status of the player at the server.

9. When updating the board after move we got stuck into a serious

problem in passing the graphics object to the related functions

which was finally solved by invoking all the related functions

from update function itself.

10. At the end of the project the updation became a tedious job

as the flow of control was hazy and the logic has to be

understood right from scratch.

11. How to identify the move is coming from which source,

whether it is server, chessboard, chat box or from the opponent

special cases.

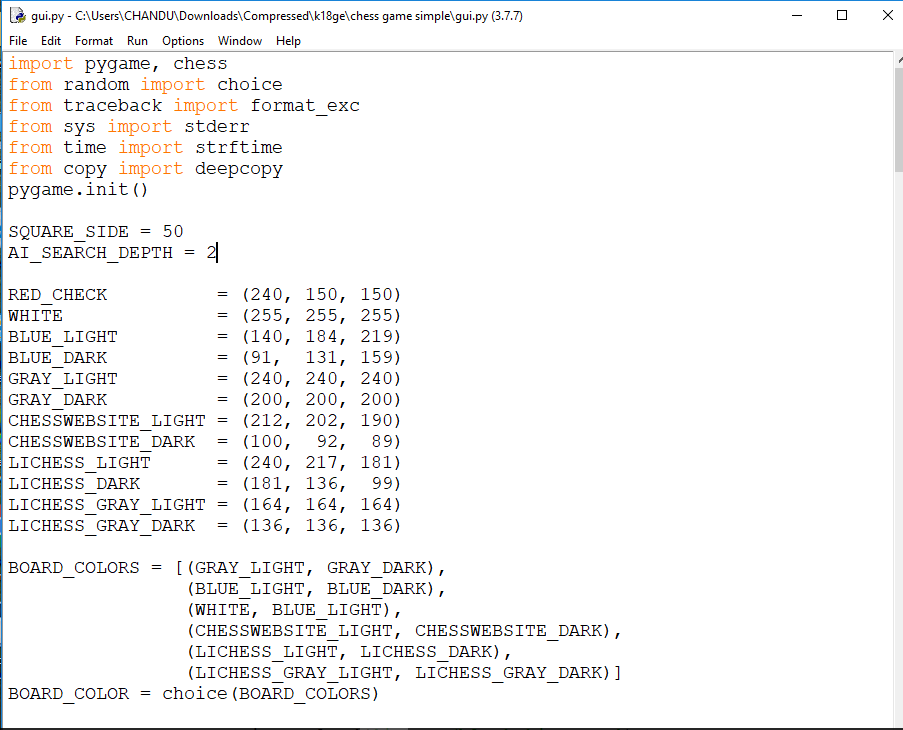
12. The difficulty was faced while implementing special chess

moves like castling, en-passent etc. which was solved by defining

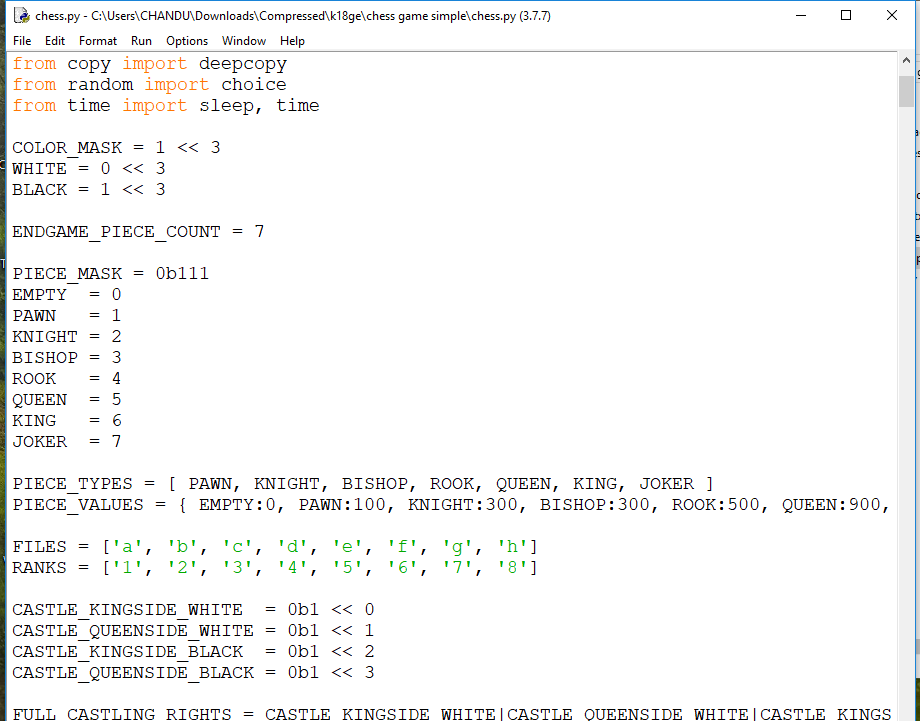
separate protocol in the portTalk class of the chessBoard.

**Code snippet:**

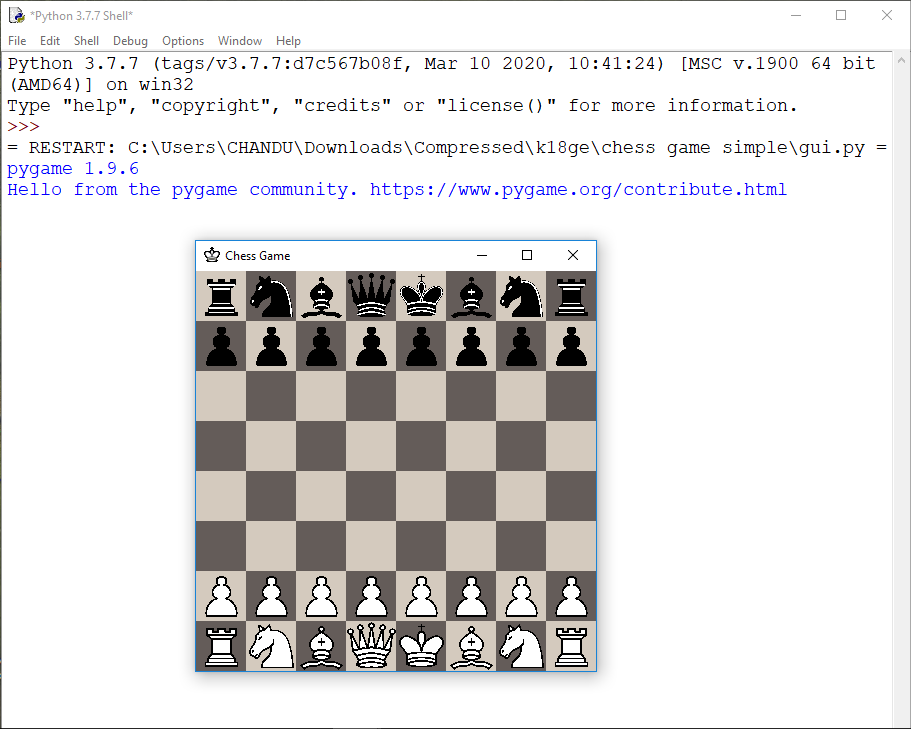
**Gui part-----**

****

**Chesspart:-**

****

**Output:**

****