Project Micro-Charter

Project Name:
American Checkers Game

Vision:
To host the American Checker's game on the public platform and make it accessible to everyone around the globe.

Project purpose:
To enable two players to play American checker's game such that at the end either a player can win or lose or make a draw with opponent player.

Elevator pitch:
This application contains a unique and user-friendly GUI design which will make you feel comfortable while playing. It also allows users to use either drag and drop option or touch and place for making a move. And a selected piece gets highlighted based on selection.

Business value:
We implemented the project based on Object-Oriented Programming principles using Java Latest Version and Java Swing which makes this application sustain for a long run. We are implementing the design to handle heavy traffic to reduce crashes.

Customers and users:
Users – Players who wants to play Online American Checkers Game irrespective of age.
Customers – Anyone who is interested in hosting our application online (e.g : a website owner to host the game for users, professor for students, etc)

Metrics:
The success of this application will be measured using average numbers of games played per month.

Milestones:
Human playing American checkers with another human
Human playing American checkers with computer

Risks:
Competitive Risks (Having many competitors outside for American Checkers Game)
Operational Risks (IT System Risk by having poor internet connection)

User Stories

Describe all user stories using the following template: As a <role>, I want <goal> [so that <benefit>]

ID	User Story Name	User Story Description	Priority	Estimated effort (hours)	Actual effort (if completed)	Status (completed, toDo, inProgress)	Developer names
1	User Starting Checkers Game	The user starts the Application of the game then the board with checker's pieces placed in correct positions is displayed.	High	10	11	completed	Eeshwara Sai Tota, Sai Kishore Reddy Surabhi, Venkata Subba Rao Inti
2	Player Moving Piece	The player makes a legal move.	High	10	12	completed	Sai Mukesh Parvathaneni, Hemanth Bandi
3	King promotion	When player's piece reaches last row, it promotes to king piece	Medium	10	9	completed	Eeshwara Sai Tota, Sai Kishore Reddy Surabhi, Venkata Subba Rao Inti
4	Capturing piece	When a player's piece hops over the opponent player's piece, he captures the opponent player's piece	High	10	13	completed	Sai Mukesh Parvathaneni, Hemanth Bandi
5	End of the game	When a player captures the opponent's last piece or when there is not valid move for the player or the players agree to a draw by mutual agreement, then the game should end	High	10	10	completed	Eeshwara Sai Tota, Sai Kishore Reddy Surabhi, Venkata Subba Rao Inti
6	Register New User	When a new user wants to play a game then he should register for game with valid user name, password, email, phone number.	Medium	10	12	completed	Sai Mukesh Parvathaneni, Hemanth Bandi
7	Login Existing User	If the user is already registered user and When a user wants to login, he should login with existing username and valid password.	Medium	10	9	completed	Eeshwara Sai Tota, Sai Kishore Reddy Surabhi, Venkata Subba Rao Inti

8	Human vs Bot	A User playing a game against the computer	High	10	10	Completed	Eeshwara Sai Tota

Acceptance Criteria (AC)

Describe all acceptance criteria using the Given-When-Then template.

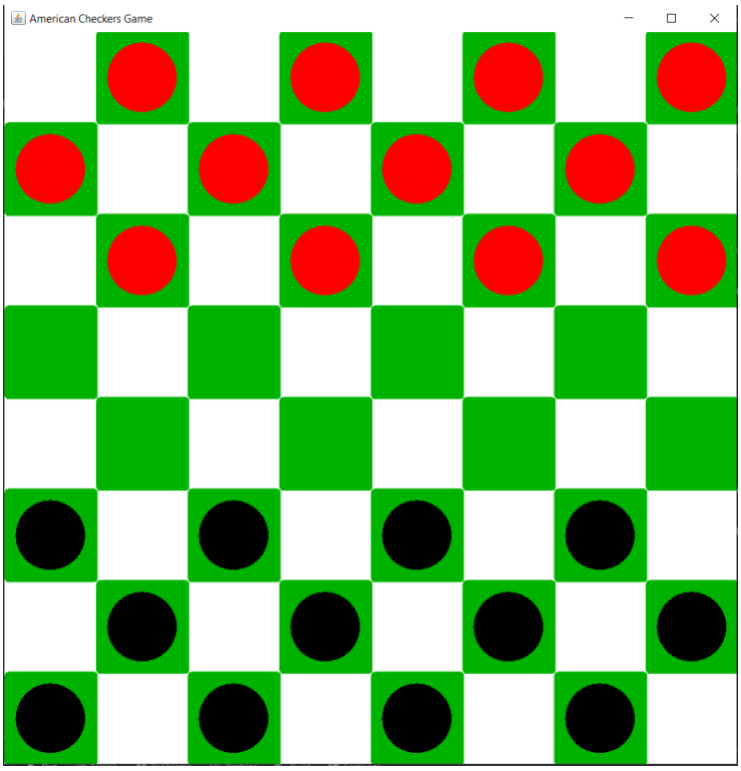
User Story ID and Name	**AC ID**	**Description of Acceptance Criterion**	**Status (completed, toDo, inProgress)**	**Developer Names**
1 User Starting Checkers Game	1.1	**AC 1.1** Start of the game **Given** the game application **When** the user runs the application **Then** a checkers board is displayed with alternate white and green squares	Completed	Eeshwara Sai Tota, Sai Kishore Reddy Surabhi, Venkata Subba Rao Inti
	1.2	**AC 1.2** Checkers piece placement on board **Given** the user runs the application **When** the checkers board is loaded **Then** the checkers pieces must be arranged in their correct position	Completed	Eeshwara Sai Tota, Venkata Subba Rao Inti
	1.3	**AC 1.3** Toss to determine who plays first **Given** the user runs the application **When** the checkers board is loaded **Then** the toss dialog box opens to determine who plays first	Completed	Eeshwara Sai Tota, Sai Kishore Reddy Surabhi
2 Players Moving Piece	2.1	**AC 2.1** First Move by Black **Given** a checkers board with checker pieces **When** the black player tries to make the first move of the game **Then** the player black should be able move his piece to desired position	Completed	Eeshwara Sai Tota, Sai Kishore Reddy Surabhi
	2.2	**AC 2.2** Invalid first move by red **Given** a checkerboard with checker pieces **When** player red tries to make the move **Then** player should not be able to move his piece to desired position	Completed	Eeshwara Sai Tota, Sai Kishore Reddy Surabhi, Venkata Subba Rao Inti
	2.3	**AC 2.3** Making a Valid Move **Given** a player's turn to make a move **When** a player tries to make a valid move **Then** the player should be able to make a valid move.	Completed	Sai Mukesh Parvathaneni, Hemanth Bandi
	2.4	**AC 2.4** Making Invalid Move **Given** a player's turn to make a move **When** a player tries to make an invalid move **Then** the player should not be able to make an invalid move.	Completed	Sai Mukesh Parvathaneni, Hemanth Bandi

	2.5	**AC 2.5** Making Valid Move by **king** piece **Given** a player's turn to make a move **When** a player tries to make a valid move using king piece **Then** the player should be able to make a valid move.	Completed	Eeshwara Sai Tota, Sai Kishore Reddy Surabhi
	2.6	**AC 2.6** Making an Invalid move by **king** piece **Given** a player's turn to make a move **When** a player tries to an invalid move by king piece **Then** the player should not be able to make an invalid move.	Completed	Sai Mukesh Parvathaneni, Inti Venkata Subba Rao
3. King Promotion	3.1	**AC 3.1** Promoting a normal piece to king piece **Given** a player's turn to move **When** the player's piece reaches last row **Then** the piece should be promoted to king piece	Completed	Sai Mukesh Parvathaneni, Inti Venkata Subba Rao
	3.2	**AC 3.2** Unsuccessful promotion of king **Given** a player's turn to move **When** the player's piece did not reach the last row **Then** the player's piece should not be promoted to king piece	Completed	Eeshwara Sai Tota, Sai Kishore Reddy Surabhi
4. Capturing a Piece	4.1	**AC 4.1** Making a Hop over opponents' piece **Given** a player's turn to move **When** a player makes a valid hop over opponent's piece **Then** the player should be able to make the valid hop and capture the opponent's piece.	Completed	Sai Mukesh Parvathaneni, Inti Venkata Subba Rao, Hemanth Bandi
	4.2	**AC 4.2** Making an unsuccessful Hop over same color piece **Given** a player's turn to move **When** a player makes an invalid hop over same colored piece **Then** the player should not be able to make the invalid hop and capture the piece	Completed	Eeshwara Sai Tota, Hemanth Bandi
5. End of the Game	5.1	**AC 5.1** Winning the Game **Given** a player's turn to move **When** the player captures opponent's last piece **Then** the player wins the game and win message appears on the screen.	Completed	Sai Kishore Reddy Surabhi, Eeshwara Sai Tota
	5.2	**AC 5.2** Drawing the Game **Given** a player's turn to move **When** neither player captures the opponent's piece in last 40 moves **Then** the game ends in draw and draw appears on the screen.	Completed	Sai Mukesh Parvathaneni, Inti Venkata Subba Rao,
	5.3	**AC 5.3** Winning the Game with no legal moves left **Given** a player's turn to move **When** a player has no legal move to make **Then** the opponent wins and win message appears on the screen.	Completed	Eeshwara Sai Tota, Hemanth Bandi
6. Register New User	6.1	**AC 6.1** Successful Registration of New User Given a non-existent valid username When I register a user with this username And a valid FirstName and Last Name And a valid password matching the confirmed password And a valid email address And a valid Phone Number Then the system should register a new user	Completed	Eeshwara Sai Tota, Hemanth Bandi

	6.2	**AC 6.2** Unsuccessful user registration with an existent username Given a username that already exists When I register a new user with this username And a valid FirstName and Last Name And a valid password matching the confirmed password And a valid email address And a valid Phone Number Then the system should not register the new user And display error message	Completed	Eeshwara Sai Tota, Hemanth Bandi
	6.3	**AC 6.3** Unsuccessful user registration with invalid password Given a non-existent valid username When I register a new user with this username And a valid FirstName and Last Name And an invalid password matching the confirmed password And a valid email address And a valid Phone Number Then the system should not register the new user And display error message	Completed	Eeshwara Sai Tota, Hemanth Bandi
	6.4	**AC 6.4** Unsuccessful user registration with valid password not matching with confirmed password Given a non-existent valid username When I register a new user with this username And a valid FirstName and Last Name And a valid password not matching the confirmed password And a valid email address And a valid Phone Number Then the system should not register the new user And display error message	Completed	Sai Mukesh Parvathaneni, Inti Venkata Subba Rao
	6.5	**AC 6.5** Unsuccessful user registration with invalid email address Given a non-existent valid username When I register a new user with this username And a valid FirstName and Last Name And a valid password matching the confirmed password And an invalid email address And a valid Phone Number Then the system should not register the new user And display error message	Completed	Sai Mukesh Parvathaneni, Inti Venkata Subba Rao
	6.6	**AC 6.6** Unsuccessful user registration with invalid Phone Number Given a non-existent valid username When I register a new user with this username And a valid FirstName and Last Name And a valid password matching the confirmed password And a valid email address And an invalid Phone Number Then the system should not register the new user And display error message	Completed	Sai Mukesh Parvathaneni, Inti Venkata Subba Rao
	6.7	**AC 6.7** Unsuccessful user registration with invalid last name Given a non-existent valid username When I register a new user with this username And a valid FirstName and an invalid Last Name And a valid password matching the confirmed password And a valid email address And a valid Phone Number Then the system should not register the new user And display error message	Completed	Sai Mukesh Parvathaneni, Inti Venkata Subba Rao
	6.8	**AC 6.8** Unsuccessful user registration with invalid first name Given a non-existent valid username When I register a new user with this username And an invalid FirstName and an valid Last Name And a valid password matching the confirmed password And a valid email address And a valid Phone Number Then the system should not register the new user And display	Completed	Sai Kishore Reddy Surabhi, Eeshwara Sai Tota

		error message		
7. Login User	7.1	**AC 7.1** Successful Login of User Given an existent valid username When I Login with this username And a valid Password Matching the password set by the user Then the system should login the user	Completed	Sai Kishore Reddy Surabhi, Eeshwara Sai Tota
	7.2	**AC 7.2** Unsuccessful Login of User with invalid password Given an existent valid username When I Login with this username And a Password not matching the password set by the user Then the system should not login the user And display error message	Completed	Sai Kishore Reddy Surabhi, Inti Venkata Subba Rao
	7.3	**AC 7.3** Unsuccessful Login of User with invalid username Given a non-existent valid username When I Login with this username And a Password matching the password set by the user Then the system should not login the user And display error message	Completed	Sai Kishore Reddy Surabhi, Inti Venkata Subba Rao
8. Bot Movement	8.1	**AC 8.1** Successful Move by Bot Given a user playing a game When a user plays a valid move Then the computer makes a valid move.	Completed	Eeshwara Sai Tota, Inti Venkata Subba Rao
	8.2	**AC 8.2** Change of Turn Given a user playing a game When the computer plays a valid move Then the turn must change to user	Completed	Sai Kishore Reddy Surabhi, Eeshwara Sai Tota

Initial User Interface Design



Description:

- The above American Checkers Game Board contains 8x8 square board which includes 64 small grids.
- It contains 32 white grids and 32 green grids.
- It includes 12 Black pieces on one side of the board and 12 Red pieces on the other side of the board.
- All pieces should be placed in the dark green grids initially before starting the game.

Implementation Tasks

1. Summary of production code

Summarize how each user story/acceptance criterion is implemented in your production code (class name and method name etc.)

User Story ID and Name	AC ID	Class Name(s)	Method Name(s)	Status (complete or not)	Developer Name(s)	Notes (optional)
1 User Starting Checkers Game	1.1	AmericanCheckersGUI	drawBoard(Graphics g)	Completed	Eeshwara Sai Tota, Sai Kishore Reddy Surabhi, Venkata Subba Rao Inti	Checks if board is visible or not
	1.2	AmericanCheckersGame, AmericanCheckersGUI	drawBoard(Graphics g), initGame()	Completed	Mukesh Parvathene ni, Hemanth Bandi	Checks if pieces are placed in correct position
	1.3	AmericanCheckersGUI	toss()	Completed	Eeshwara Sai Tota, Sai Kishore Reddy Surabhi, Venkata Subba Rao Inti	To choose which player will start first
2 Players Moving Piece	2.1	AmericanCheckersGame AmericanCheckersGUI	validMove() makeMove() changeTurn()	Completed	Eeshwara Sai Tota, Sai Kishore Reddy Surabhi, Venkata Subba Rao Inti	Make sure first move is played by black
	2.2	AmericanCheckersGame AmericanCheckersGUI	validMove()	Completed	Mukesh Parvathe neni, Hemanth Bandi	When red tries to play the first move, it should not be allowed

	2.3	AmericanCheckersGame AmericanCheckersGUI	validMove() makeMove() changeTurn()	Completed	Eeshwara Sai Tota, Sai Kishore Reddy Surabhi, Venkata Subba Rao Inti	Any valid move should be allowed and turn should be changed.
	2.4	AmericanCheckersGame AmericanCheckersGUI	validMove()	Completed	Mukesh Parvathe neni, Hemanth Bandi	Invalid move should not be allowed and turn should remain same
	2.5	AmericanCheckersGame AmericanCheckersGUI	validMove() makeMove() isKing() changeTurn()	Completed	Eeshwara Sai Tota, Sai Kishore Reddy Surabhi, Venkata Subba Rao Inti	King piece should be allowed to move in all four diagonal ways
	2.6	AmericanCheckersGame AmericanCheckersGUI	validMove() isKing()	Completed	Mukesh Parvathe neni, Hemanth Bandi	Making invalid move by King should be restricted
3. King Promotion	3.1	AmericanCheckersGame AmericanCheckersGUI	validMove() makeMove() setKing() changeTurn()	Completed	Eeshwara Sai Tota, Sai Kishore Reddy Surabhi, Venkata Subba Rao Inti	When normal piece reaches opponent first row, it should be promoted to king
	3.2	AmericanCheckersGame AmericanCheckersGUI	validMove() makeMove() changeTurn()	Completed	Eeshwara Sai Tota, Sai Kishore Reddy Surabhi, Venkata Subba Rao Inti	A normal piece not reaching opponent last row should remain a normal piece
4. Capturing a Piece	4.1	AmericanCheckersGame AmericanCheckersGUI	validMove() makeMove() changeTurn()	Completed	Mukesh Parvathe neni, Hemanth Bandi	Killing a opponent piece should be allowed

	4.2	AmericanCheckersGame AmericanCheckersGUI	validMove() changeTurn()	Completed	Eeshwara Sai Tota, Sai Kishore Reddy Surabhi, Venkata Subba Rao Inti	Killing self coloured piece should not be allowed
5. End of the Game	5.1	AmericanCheckersGame AmericanCheckersGUI	validMove() makeMove() changeTurn() setGameState()	Completed	Mukesh Parvathe neni, Hemanth Bandi	When no opponent piece remains, he should be declared winner
	5.2	AmericanCheckersGame AmericanCheckersGUI	validMove() makeMove() changeTurn() setGameState()	Completed	Eeshwara Sai Tota, Sai Kishore Reddy Surabhi, Venkata Subba Rao Inti	If no piece is captured in last 40 moves, then game should drawn
	5.3	AmericanCheckersGame AmericanCheckersGUI	validMove() makeMove() changeTurn() setGameState()	Completed	Mukesh Parvathe neni, Hemanth Bandi	When opponent has no legal moves, he should be declared win.
6. Register User	6.1	UserAccount UserAccountManager UserManagerGUI	registerNewUser() doesUserNameExist() executeUpdate()	Completed	Eeshwara Sai Tota, Sai Kishore Reddy Surabhi, Venkata Subba Rao Inti	New user registers successfully
	6.2	UserAccount UserAccountManager UserManagerGUI	registerNewUser() doesUserNameExist() executeUpdate() isUserNameValid()	Completed	Mukesh Parvathe neni, Hemanth Bandi	User registration fails if user enters invalid username
	6.3	UserAccount UserAccountManager UserManagerGUI	registerNewUser() doesUserNameExist() executeUpdate() isPasswordValid()	Completed	Eeshwara Sai Tota, Sai Kishore Reddy Surabhi, Venkata Subba Rao Inti	User registration fails if invalid password

	6.4	UserAccount UserAccountManager UserManagerGUI	registerNewUser() doesUserNameExist() executeUpdate() matchPassword()	Completed	Mukesh Parvatheneni, Hemanth Bandi	User registration fails if invalid password reenter
	6.5	UserAccount UserAccountManager UserManagerGUI	registerNewUser() doesUserNameExist() executeUpdate() isEmailValid()	Completed	Eeshwara Sai Tota, Sai Kishore Reddy Surabhi, Venkata Subba Rao Inti	User registration fails if invalid email
	6.6	UserAccount UserAccountManager UserManagerGUI	registerNewUser() doesUserNameExist() executeUpdate() isPhoneValid()	Completed	Eeshwara Sai Tota, Sai Kishore Reddy Surabhi, Venkata Subba Rao Inti	User registration fails if invalid phone number
	6.7	UserAccount UserAccountManager UserManagerGUI	registerNewUser() doesUserNameExist() executeUpdate() isLastNameValid()	Completed	Mukesh Parvatheneni, Hemanth Bandi	User registration fails if invalid last name
	6.8	UserAccount UserAccountManager UserManagerGUI	registerNewUser() doesUserNameExist() executeUpdate() isFirstNameValid()	Completed	Eeshwara Sai Tota, Sai Kishore Reddy Surabhi, Venkata Subba Rao Inti	User registration fails if invalid first name
7. login user	7.1	UserAccount UserAccountManager UserManagerGUI	login() executeQuery()	Completed	Mukesh Parvatheneni, Hemanth Bandi	User login successgul
	7.2	UserAccount UserAccountManager UserManagerGUI	login() executeQuery()	Completed	Eeshwara Sai Tota, Sai Kishore Reddy Surabhi, Venkata Subba Rao Inti	User login fails if wrong password

	7.3	UserAccount UserAccountManager UserManagerGUI	login() executeQuery()	Completed	Eeshwara Sai Tota, Sai Kishore Reddy Surabhi, Venkata Subba Rao Inti	User login fails if wrong username
8. Bot Movement	8.1	AmericanCheckersGame AmericanCheckersGUI	findValidBotMove()	Completed	Eeshwara Sai Tota, Sai Kishore Reddy Surabhi, Venkata Subba Rao Inti	Bot makes a valid move
	8.2	AmericanCheckersGame AmericanCheckersGUI	findValidBotMove()	Completed	Sai Mukesh Parvatha neni, Eeshwara Sai Tota	Turn changes to user

Summary of automated test code directly corresponding to some acceptance criteria

Summarize how user stories/acceptance criterion are tested by your test code (class name and method name).

User Story ID and Name	**Acceptance Criterion ID**	**Class Name (s) of the Test Code**	**Method Name(s) of the Test Code**	**Description of the Test Case (input & expected output)**	**Status**	**Developer Name(s)**
1 User Starting Checkers Game	1.1	CheckersBoardTest	testCheckersBoard()	Checks if board is visible or not	PASS	Eeshwara Sai Tota
	1.2	CheckersBoardTest	testCheckersBoardPiecePlacement()	Checks if pieces are placed in correct position	PASS	Sai Kishore Reddy Surabhi
2. Players Moving Piece	2.1	CheckersMoveTest	testFirstMoveByBlack()	Make sure first move is played by black	PASS	Eeshwara Sai Tota
	2.2	CheckersMoveTest	testFirstByRed()	When red tries to play the first move, it should not be allowed	PASS	Eeshwara Sai Tota

	2.3	CheckersMoveTest	testValidMove()	Any valid move should be allowed and turn should be changed.	PASS	Eeshwa ra Sai Tota
	2.4	CheckersMoveTest	testInvalidMove()	Invalid move should not be allowed and turn should remain same	PASS	Eeshwa ra Sai Tota
	2.5	CheckersMoveTest	testValidMoveByKing()	King piece should be allowed to move in all four diagonal ways	PASS	Eeshwa ra Sai Tota
	2.6	CheckersMoveTest	testInvalidMoveByKing()	Making invalid move by King should be restricted	PASS	Eeshwa ra Sai Tota
3. King Promotion	3.1	KingPromotionTest	testSuccessfulKingPromotion()	When normal piece reaches opponent first row, it should be promoted to king	PASS	Sai Kishore Reddy Surabhi
	3.2	KingPromotionTest	testUnSuccessfulKingPromotion()	A normal piece not reaching opponent last row should remain a normal piece	PASS	Sai Kishore Reddy Surabhi
4. Capturing a Piece	4.1	HopTest	TestValidHop()	Killing a opponent piece should be allowed	PASS	Inti Venkata Subba Rao
	4.2	HopTest	TestValidHop()	Killing self coloured piece should not be allowed	PASS	Inti Venkata Subba Rao
5. End of Game	5.1	EndGameTest	winPositionTest()	When no opponent piece remains, he should be declared winner	PASS	Sai Mukesh Parvath aneni
	5.2	EndGameTest	drawGamesTest()	If no piece is captured in last 40 moves, then game should drawn	PASS	Sai Mukesh Parvath aneni
6. Register New User	6.1	Test registration	testSuccessfulRegistration()	User registers successfully	PASS	Sai Kishore Reddy Surabhi
	6.2	Invalid user name	testUnsuccessfulRegistrationWithInvalidUserName()	User registration fails	PASS	Venkat Subba Rao Inti

	6.3	Invalid password	testUnsuccessfulRegistrationWithInvalidPassword()	User registration fails	PASS	Sai Mukesh Parvath aneni
	6.4	InvalidEmail	testUnsuccessfulRegistrationWithInvalidEmail()	User registration fails	PASS	Sai Kishore Reddy Surabhi
	6.5	InvalidPhone	testUnsuccessfulRegistrationWithInvalidPhone()	User registration fails	PASS	Eeshwara Sai Tota
	6.6	InvalidFirstName	testUnsuccessfulRegistrationWithInvalidFirstName()	User registration fails	PASS	Eeshwara Sai Tota
7.login user	7.1	Successful login	testSuccessfulLogin()	User login Successful Login	PASS	Sai Kishore Reddy Surabhi
	7.2	Invalid password	testUnsuccessfulLoginWrongPassword()	User login fails	PASS	Venkata Subba Rao Inti
	7.3	Invalid username	testUnsuccessfulLoginWrongUserName()	User login fails	PASS	Venkata Subba Rao Inti
8. Bot Movement	8.1	Bot makes move	testMoveByBot()	Bot moves successfully	PASS	Sai Kishore Reddy Surabhi
	8.2	Bot Hops	testHopByBot()	Hops Successfully	PASS	Venkata Subba Rao Inti
	8.3	Hop by human on bot	testHopMoveByHumanOnBot()	Hops Successfully	PASS	Venkata Subba Rao Inti
	8.4	Change turn after bot move	testTurnAfterBotMove()	Turn changes successfully	PASS	Sai Kishore Reddy Surabhi

6. Summary of manual test cases directly corresponding to some acceptance criteria

Summarize how user stories/acceptance criterion are tested manually

User Story ID and Name	Acceptance Criterion ID	Test Case Input	Test Oracle (Expected Output)	Status	Notes	Developer Name(s)
1	1.1	-	Checkers board	PASS	Checkers board should be display	Venkata Subba Rao Inti
	1.2	-	Piece arrangement	PASS	Checkers piece should be arranged correctly	Sai Mukesh Parvathaneni, Hemanth Bandi
	1.3	-	Player 1 or Player 2	PASS	Toss Dialog Box with toss winner	Eeshwara Sai Tota
6	6.1	-	Register page opens	PASS	Register page to register new user opens	Sai Kishore Reddy Surabhi
7	7.1	-	Login Page	PASS	Login page opens	Venkata Subba Rao Inti, Sai Kishore Reddy Surabhi
8	8.1	-	Human vs computer game starts	PASS	Vs computer game starts	Venkata Subba Rao Inti, Sai Kishore Reddy Surabhi

7. Summary of other automated or manual tests not corresponding to the acceptance criteria

Number	Test Input	Expected Result	Class Name of the Test Code	Method Name of the Test Code	Status	Developer Name(s)
1	RED, BLACK	RED, BLACK	CheckersPiecePositionTest	testCheckersPieceColor()	PASS	Eeshwara Sai Tota, Venkata Subba Rao Inti
2	-	false, true	CheckersPiecePositionTest	testCheckersKingPiece()	PASS	Sai Kishore Reddy Surabhi, Hemanth Bandi
3	-	Move by black	CheckersMoveTest	testFirstMoveByBlack()	PASS	Sai Kishore Reddy Surabhi, Hemanth Bandi
4	-	No move by red	CheckersMoveTest	testFirstMoveByRed()	PASS	Sai Kishore Reddy Surabhi, Hemanth Bandi

5	-	Red piece killed	HopTest	BlackTestValidHop()	PASS	Sai Kishore Reddy Surabhi, Hemanth Bandi
6	-	Black piece killed	HopTest	RedTestValidHop()	PASS	Eeshwara Sai Tota, Venkata Subba Rao Inti
7	-	Red piece not killed	HopTest	BlackTestInvalidHop()	PASS	Eeshwara Sai Tota, Venkata Subba Rao Inti
8	-	Black piece not killed	HopTest	BlackTestInvalidHop()	PASS	Eeshwara Sai Tota, Venkata Subba Rao Inti
9	-	Black king piece	KingPromotionTest	testBlackSuccessfulKingPromotion()	PASS	Eeshwara Sai Tota, Venkata Subba Rao Inti
10	-	Red king piece	KingPromotionTest	testRedSuccessfulKingPromotion()	PASS	Sai Mukesh Parvathaneni, Sai Kishore Reddy Surabhi
11	-	Black normal piece	KingPromotionTest	testBlackUnSuccessfulKingPromotion()	PASS	Sai Mukesh Parvathaneni, Sai Kishore Reddy Surabhi
12	-	Red normal piece	KingPromotionTest	testRedUnSuccessfulKingPromotion()	PASS	Sai Mukesh Parvathaneni, Sai Kishore Reddy Surabhi
13	-	Red wins	EndGameTest	redWinPosition()	PASS	Eeshwara Sai Tota, Venkata Subba Rao Inti
14	-	Black Wins	EndGameTest	blackWinPosition()	PASS	Eeshwara Sai Tota, Venkata Subba Rao Inti
15	-	Game Draws	EndGameTest	drawGameTest()	PASS	Eeshwara Sai Tota, Venkata Subba Rao Inti

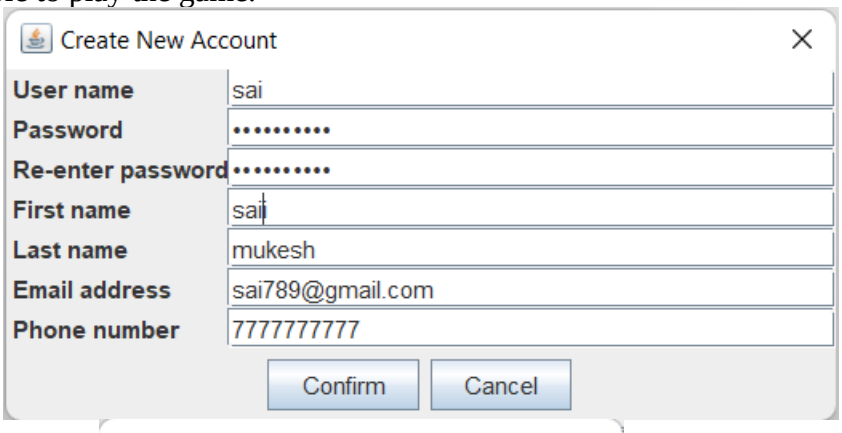
Design Documentation

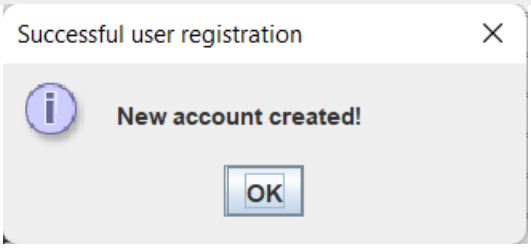
Contributors: Venkata Subba Rao Inti, Sai Kishore Reddy Surabhi

USER INTERFACE DESIGN:

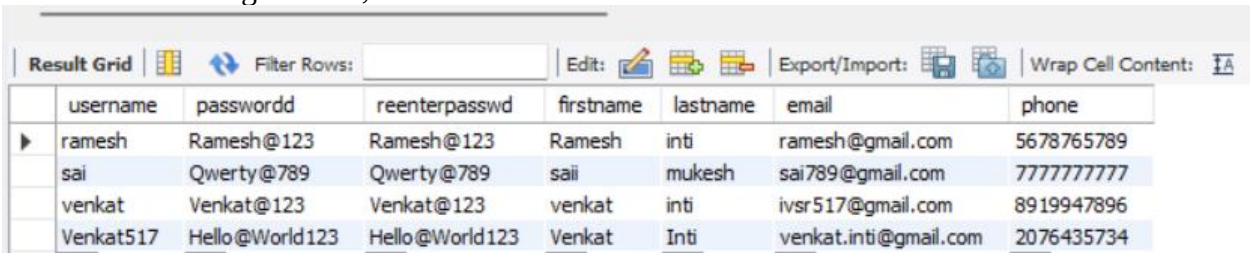
1.Register New User:

Whenever a user wants to play a game, the user should register for a game. For that, user should enter a valid username, first name, last name, email, password, reentered password, and phone number. Once all these fields are validated and verified then user should be able to register. Entered details will also get stored in the *registration* table in the *"American checkers"* MySQL database and once user successfully registered, he or she can be able to log in to the game and similarly whoever wants to play the game needs to register for the game. Only registered users can be able to play the game.





After successful registration, database will have the entered values as shown below.

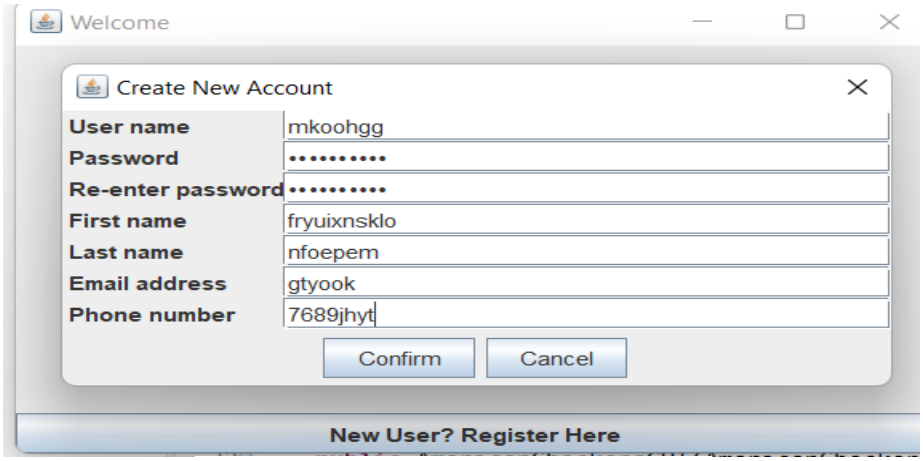


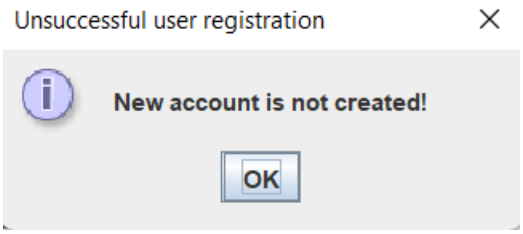
Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

username	passwordd	reenterpasswd	firstname	lastname	email	phone
ramesh	Ramesh@123	Ramesh@123	Ramesh	inti	ramesh@gmail.com	5678765789
sai	Qwerty@789	Qwerty@789	saii	mukesh	sai789@gmail.com	7777777777
venkat	Venkat@123	Venkat@123	venkat	inti	ivsr517@gmail.com	8919947896
Venkat517	Hello@World123	Hello@World123	Venkat	Inti	venkat.inti@gmail.com	2076435734

2. Register New User with Invalid Data:

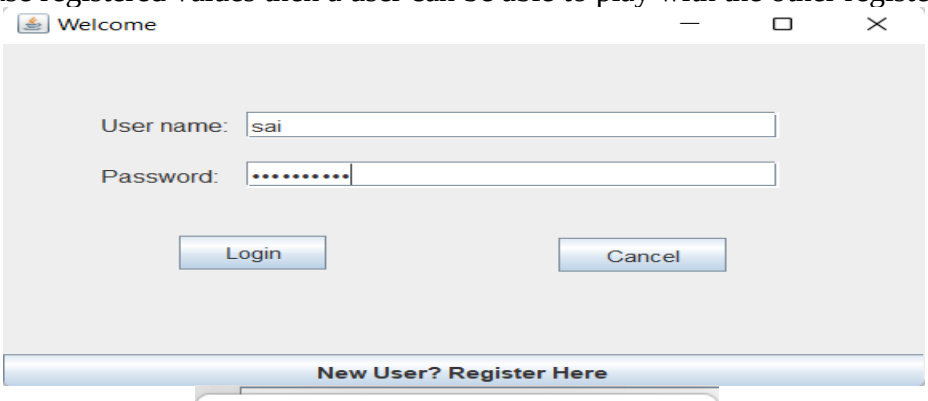
If user entered values mismatches with any of the validation scenarios, then you will get a pop up saying that invalid data entered. And also, data won't be stored in the MySQL American checkers Database created by our team.

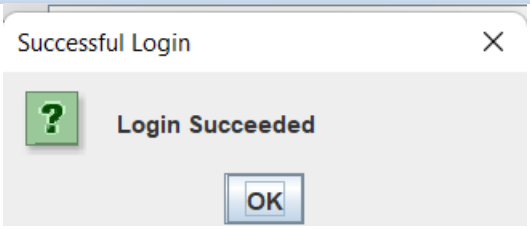




3. Login User:

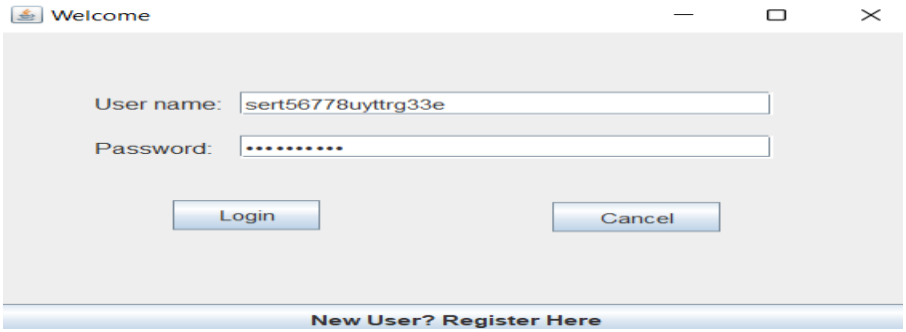
While doing login, a user should enter username as well as password, once this username and password matches with the database registered values then a user can be able to play with the other registered login user.

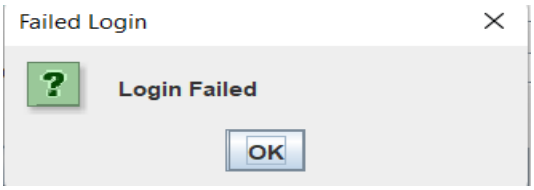




4. Invalid Login User:

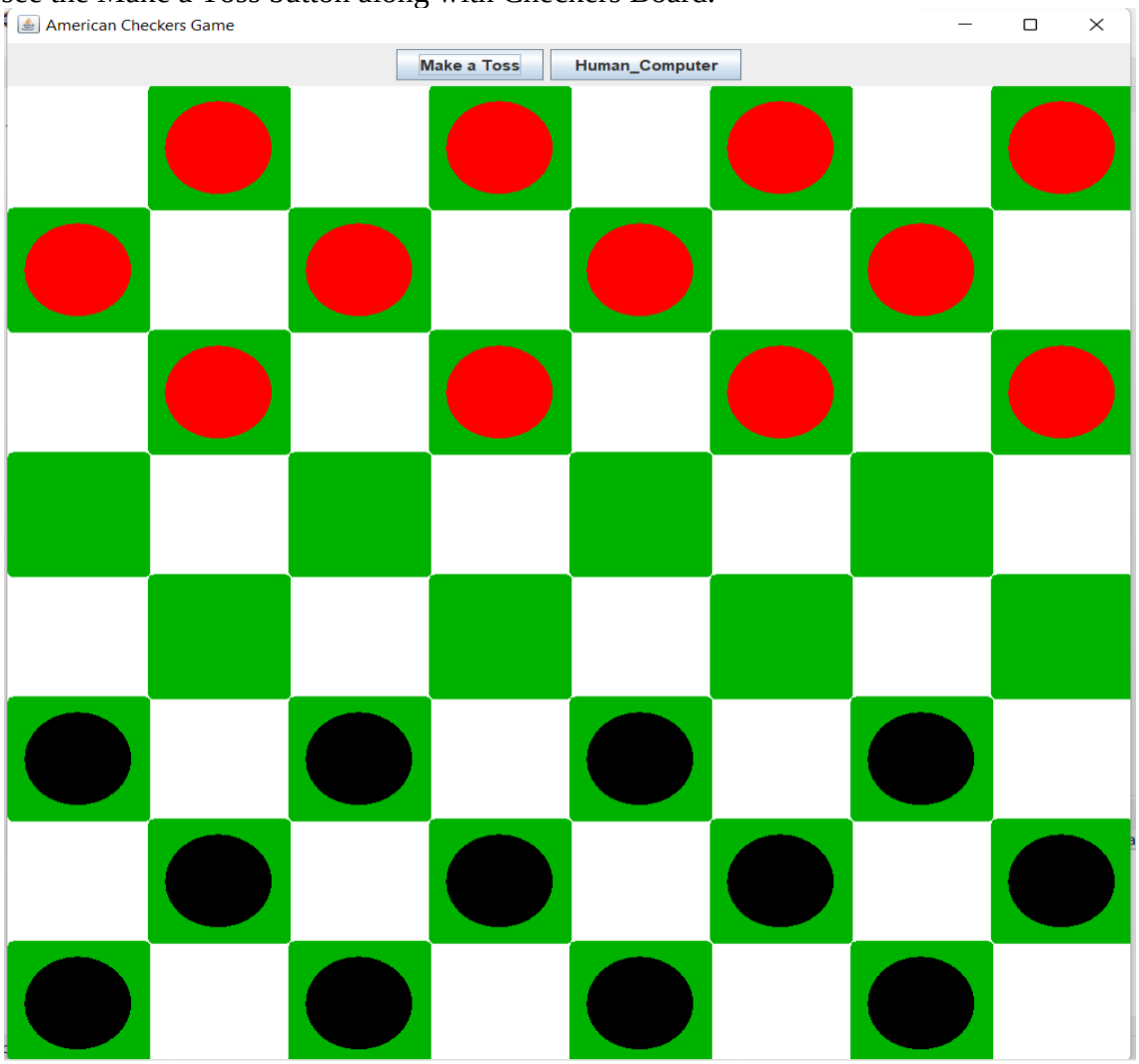
If you enter username or password mismatches with the database record, then you will get a pop-up with message no registered user.



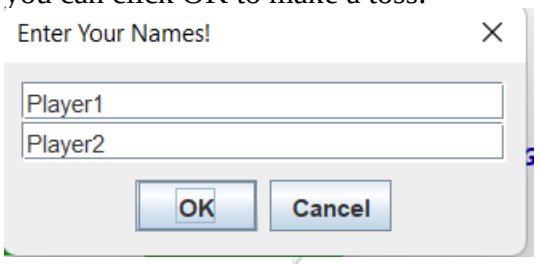


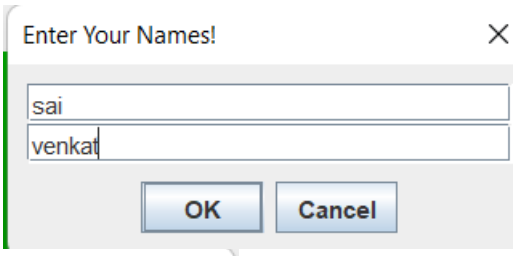
5. Make a Toss:

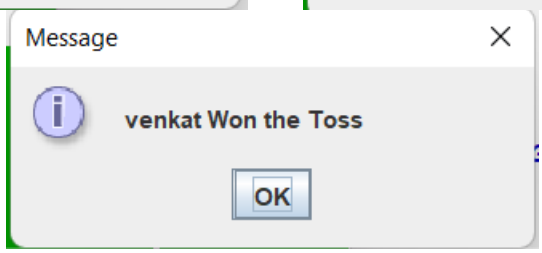
Once the user successfully logged in, then you will get a game to proceed to make a toss. And now you will be able to see the Make a Toss button along with Checkers Board.



When you click the make a toss button it will pop up a message to enter player1 name and player2 name and then you can click OK to make a toss.



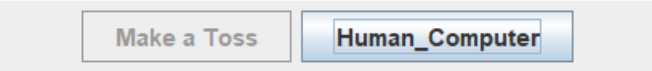




6. Toss Disable Functionality:

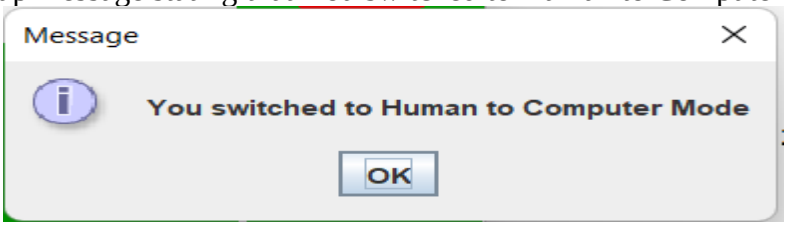
Whoever wins the toss will get the chance to make a move first and the assigned color for toss winner is Black.

And both users can only be able to make a toss one time and after making a toss, it becomes disable automatically.



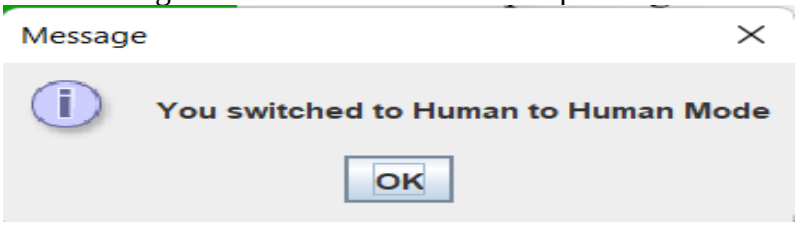
7. Switching to Computer Mode:

And now if you want to switch from human mode to computer mode then you need to click on Human vs Computer Button beside the toss button on top of the game UI. If you click on Human to Computer Button, you will get a pop-up message stating that You switched to Human to Computer Mode.



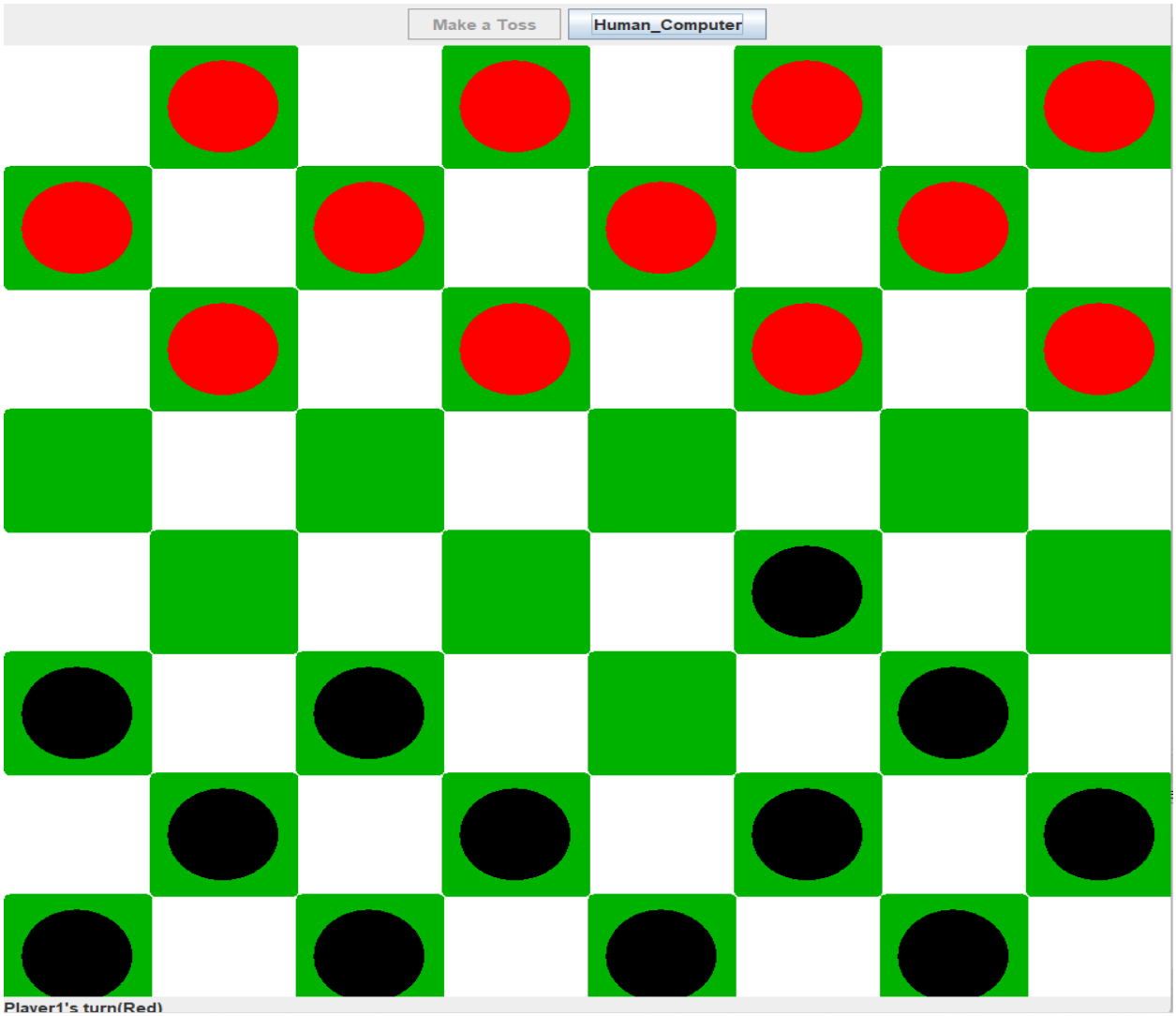
8. Switching to Human Mode from Computer Mode:

And if you want to go back to normal mode or Human Mode, you can click again on Human vs computer button then it will change to Human-to-Human Mode. If you want to continue with the computer opponent, you can continue till the end of the game. It will work and our computer code will defeat the worst human player.

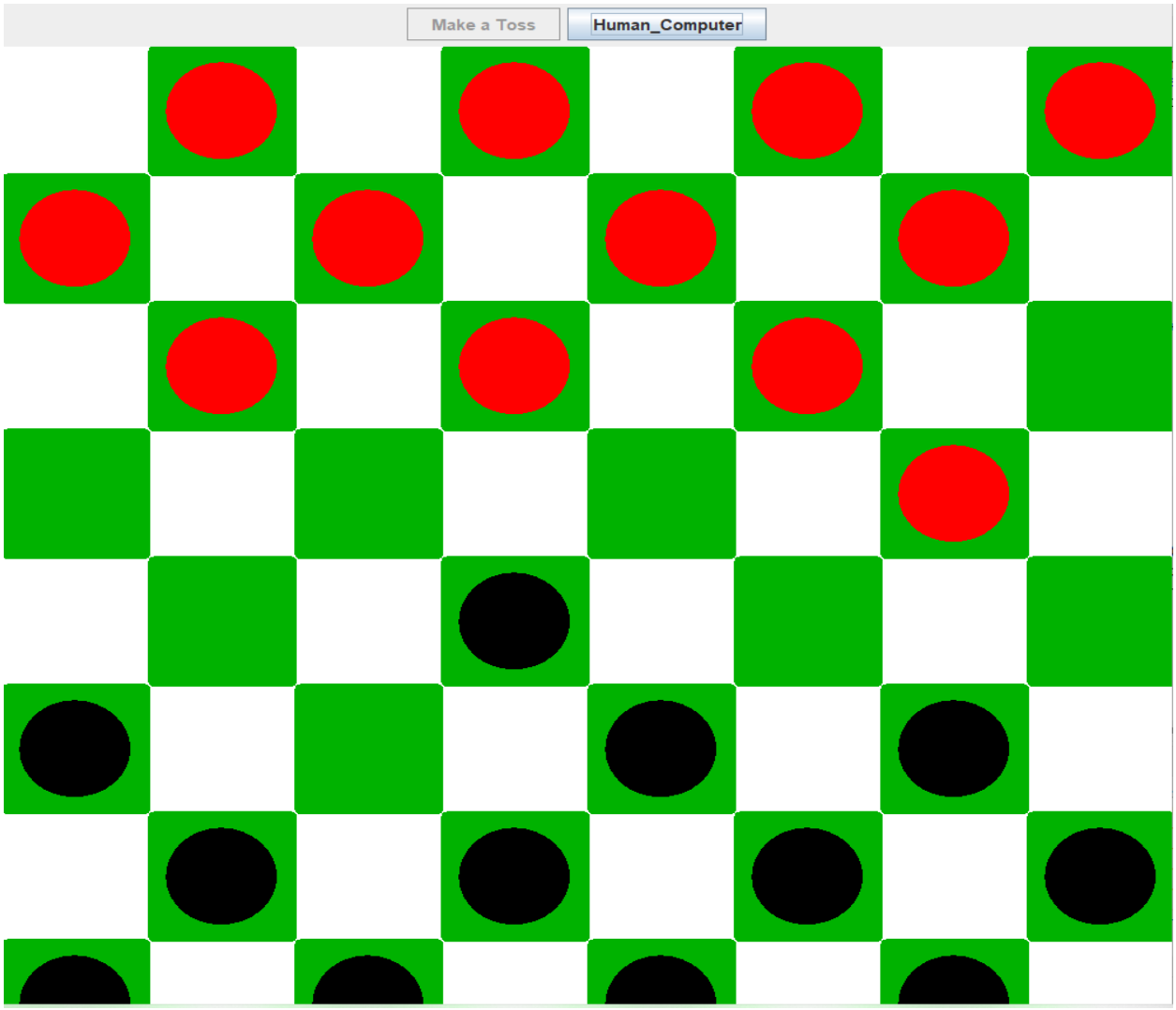


9. Game Play:

Now you can be able to play the game either with 2 registered users or registered user with computer. Black will make a first move one step diagonally upward or downward direction. And then red will make a one-step move diagonally upward. And every time, user can only be able to make a diagonal move unless it is a hop move.

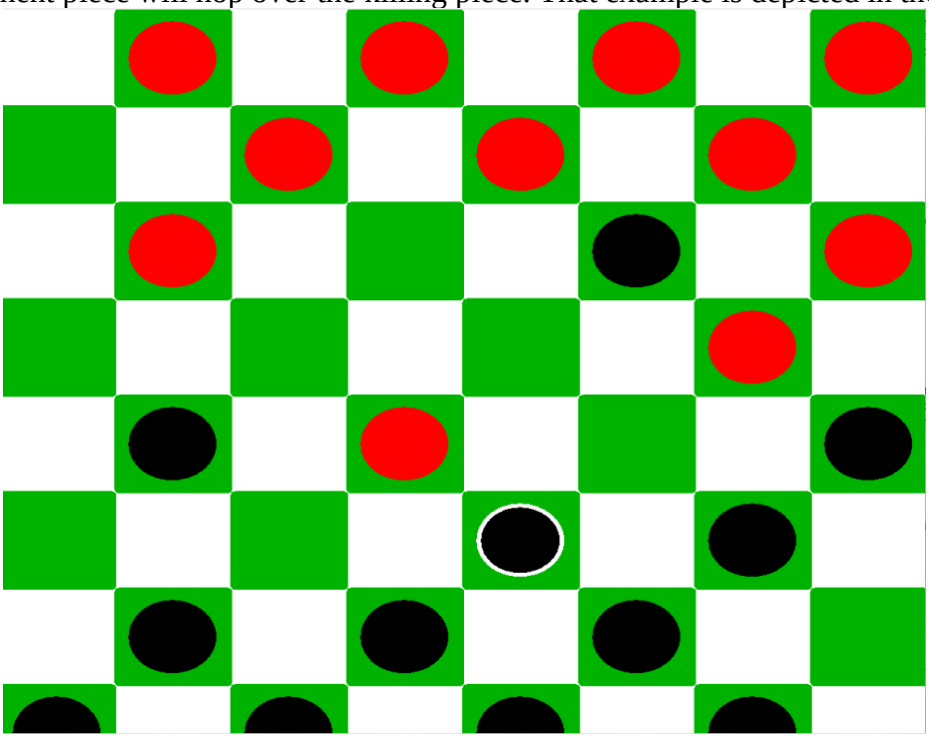


Red Move followed by Black Move:

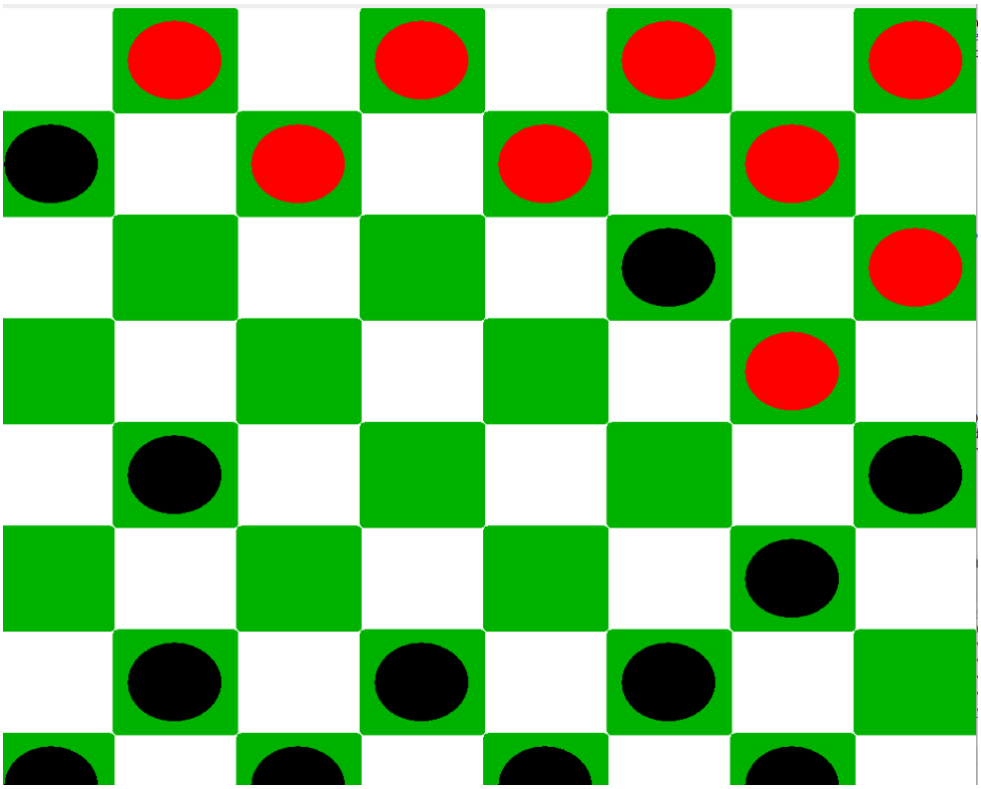


We have the seen the movements of black and red diagonally upward direction. Whenever there is a change for killing a piece then opponent piece will hop over the killing piece. That example is depicted in the below image.

Before Hop:

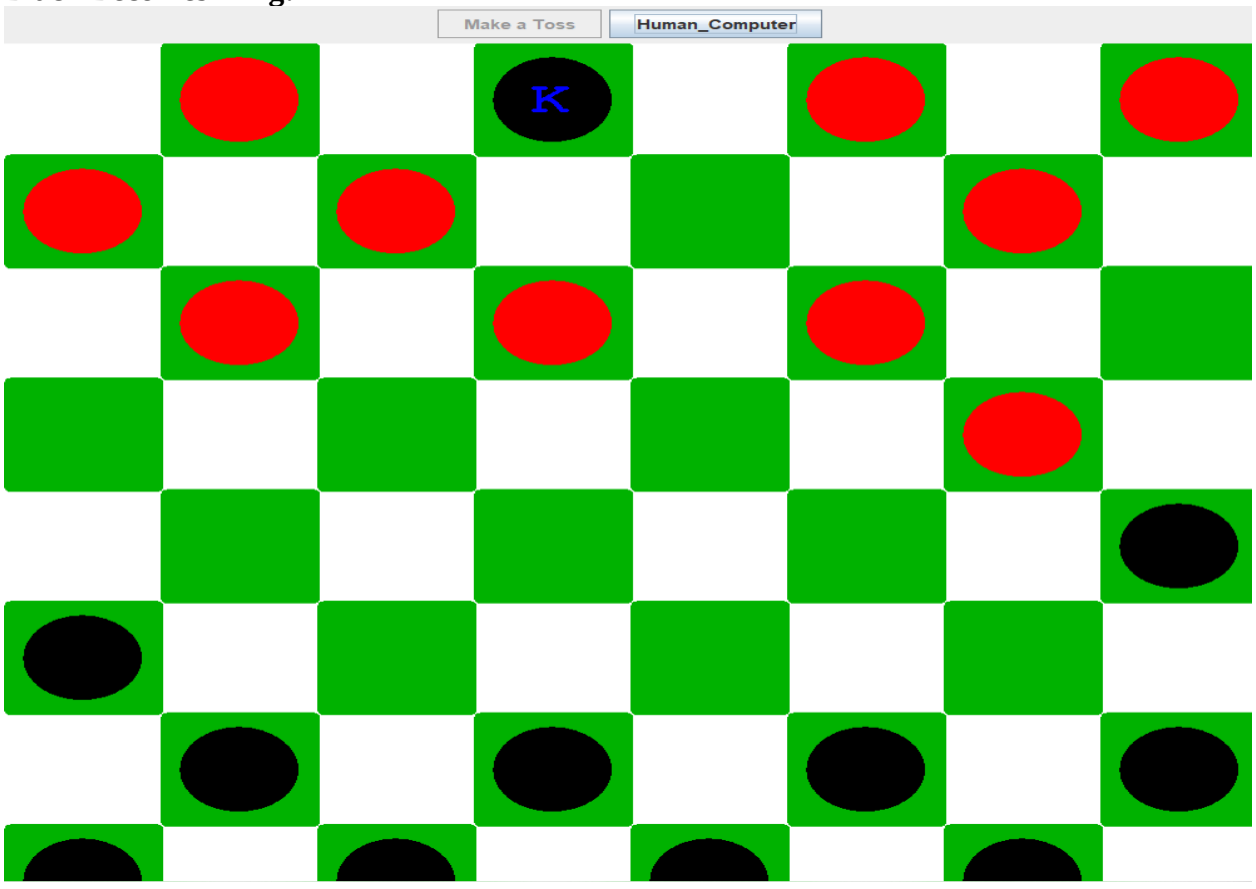


After Hop:

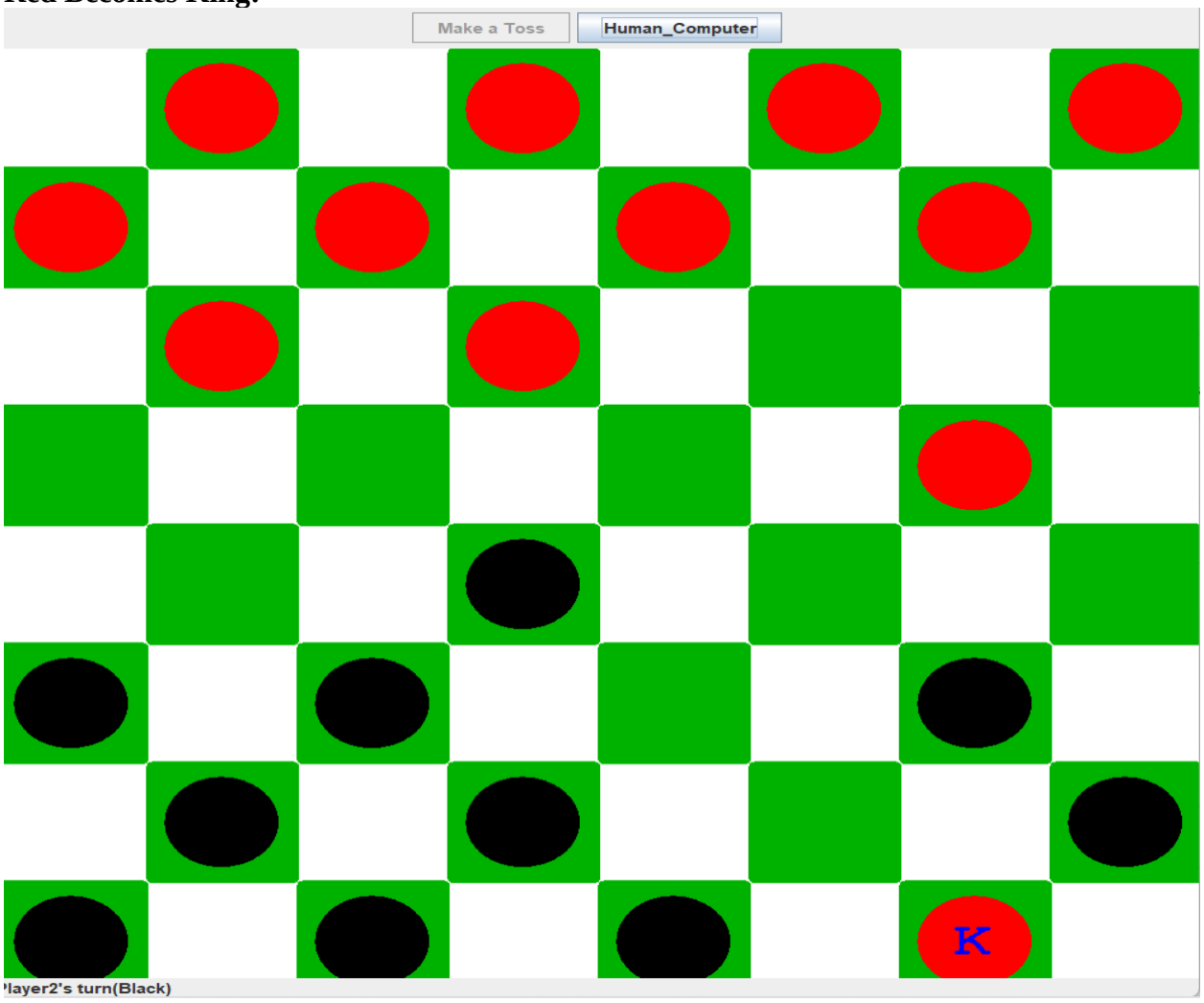


And, whenever a piece reaches the last row of the opponent it will become the King Piece and it will have some added functionality like a king can either move one step diagonally upward or downward direction.

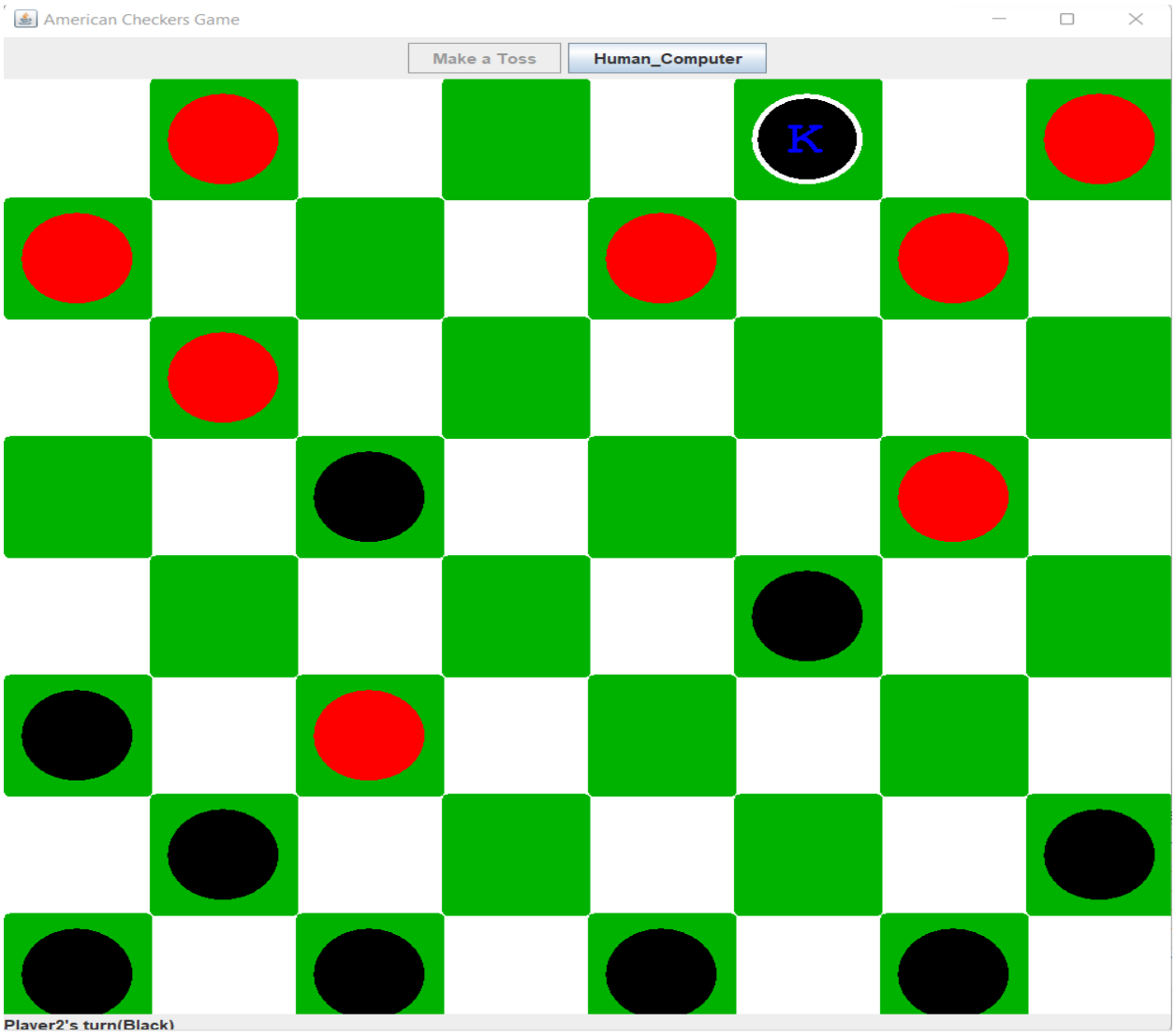
Black Becomes King:



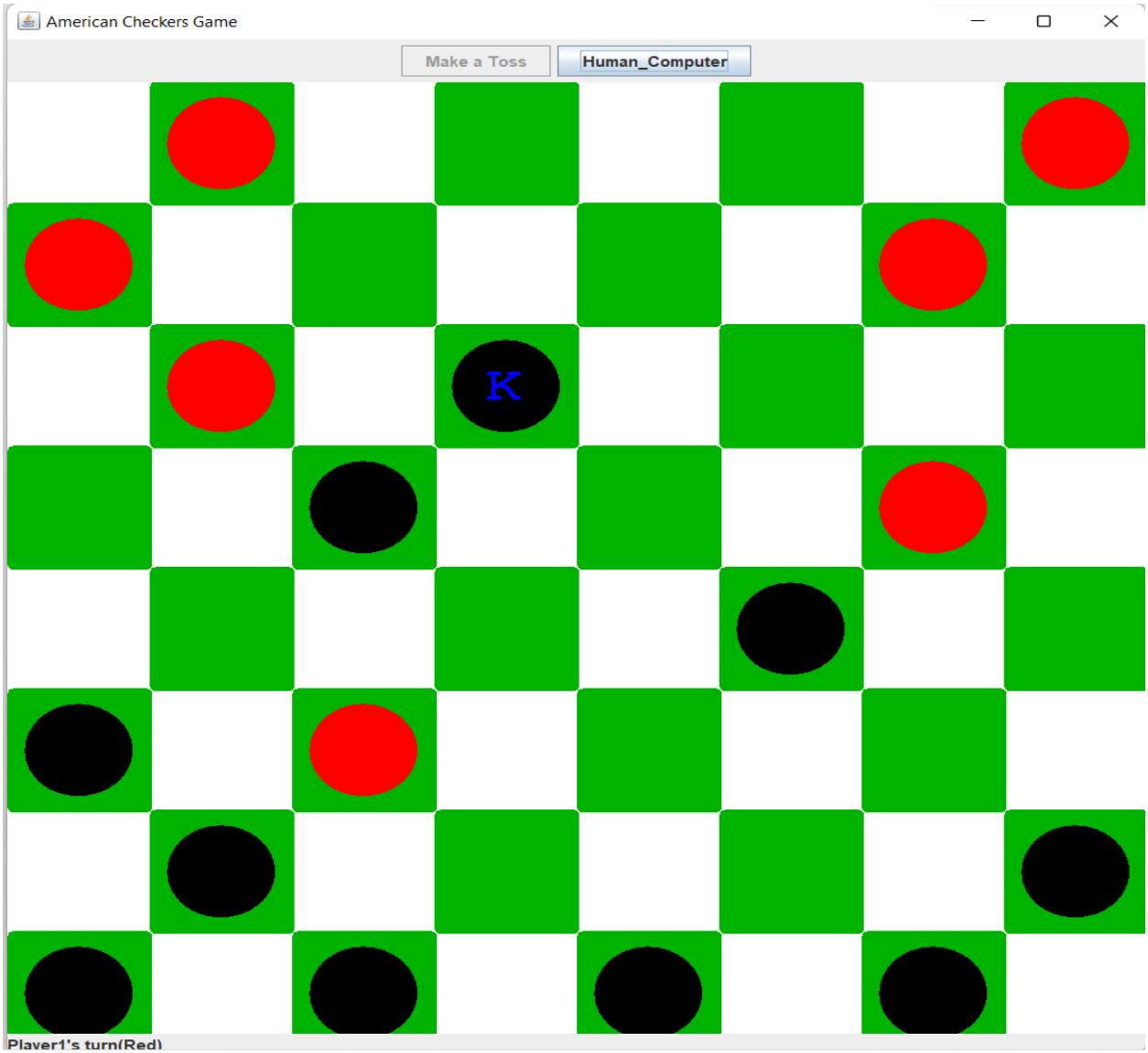
Red Becomes King:



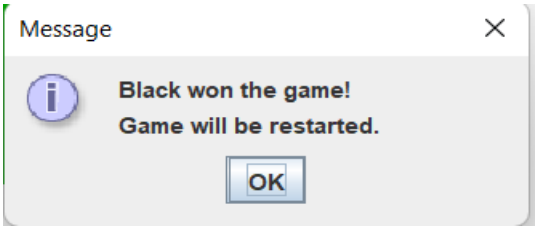
And killing is same as the normal piece in the checker's game. We have added some screenshots with some steps that will show how a normal piece becomes a king and how it moves upward and downward direction. And how it kills the other opponents.



Black King making downward step and killing in downward direction. After hoping it becomes,



User will become as a winner if he kills all of his opponent pieces in the game. If he kills all piece then game will ends by stating a message “User is the Winner”.



And if a user clicks OK, then the game will reset, and game data gets stored. If the same registered users want to play again, they can play again with the current console. If not, then can logout. And if new user wants to play then he should start the process from the beginning and play with other registered players.

Software architecture

Contributors: Eeshwara Sai Tota, Sai Mukesh Parvathaneni

The design for this game has 2 main categories.
1. AmericanCheckersGame
2. CheckersPiece
3. AmericanCheckersGUI

The above-mentioned classes are related to American Checkers Game user interface and game mechanics.
1. UserAccount
2. UserAccountManger
3. EditProfileWindow
4. UserAccountManagerUI

These classes are related to User Management of players, their information and game records which are saved on MySQL database

AmericanCheckersGame
It consists of methods and variables related to game mechanics which include functionality for valid piece movement, piece promotion and a bot with whom a human can play when there are no active players logged in.

validMove():
To check if the user is trying to make a valid move.

makeMove():
Once a move is verified as a valid move, the move can be made using makeMove() method.

changeTurn():
To end the turn of the user and give opponent a chance to play

findValidBotMove():
To find a bot move when it is the turn of the bot.

CheckersPiece
This object represents the physical piece on the board which has attributes representing color and piece promotion status. And their actual position on board is memorized using grid array.

setKing():
To promote a normal checkers piece to a king piece.

AmericanChecksGUI
This class acts as representation class for the above two classes and is built using java applets. It consists of board and pieces through which players interact with the game.

GameBoardCanvas():
The actual canvas layout where checkers board and pieces are placed.

UserAccount and UserAccountManager
These classes deal with user records and their games played. It manages user records actions such as creation, update and deletion are performed on stored on MySQL database.

registerNewUser():
To register a new user and give him privilege to play the game.

login():
Help registered users to login and play the game.

UserAccountManagerUI and EditProfileWindow
These are pop up windows and help users to register and play the game. Once registered, users can login and play the game.

buttonHandlers():
Contains mostly button handlers and has details about the action need to be performed when buttons are clicked.

Algorithm Design

Contributors: Venkata Subba Rao Inti, Sai Kishore Reddy Surabhi

The computer bot is designed based on brute approach and can beat a human being who is bad playing the game.

Pseudo code

```
BEGIN
checkerboard
turn ← bot
possibleMoves ← 4;
WHILE row 1 : 8
        WHILE column 1 : 8
                botCheckerPiece ← (row,colums)
                botCheckersPiece != empty AND botCheckersPiece.color = RED
                IF botCheckersPiece.king = true
                        possibleMoves ← 8
                ENDIF
                WHILE move 1: possibleMoves
                        IF checkerboard(move.endposition) = empty
                                botCheckersPiece.position = move.endposition
                                IF move.killmove = true
                                        humanCheckerPiece = remove
                                ENDIF
                                Turn← human
                        ENDIF
                ENDWHILE
        ENDWHILE
ENDWHILE
END
```

Algorithm working

The algorithm is based on brute force approach where we scan the checkers board from top left and continue row wise for a RED colored piece (computer bot always has red colored pieces). Once we find a such a piece we try to see if there is any legal move to make, 4 possible moves for a normal piece and 8 possible moves for a king promoted piece which include normal move and a hop move which removes a human piece from the board. If there is no such move, we continue to next checkers piece of bot and find a legal move for that.

In this way computer bot makes a move and when there is only one human's checkers piece and stays diagonally 1 square away from computer bot piece, the computer bot kills the opponent's piece and wins the game.

Extensibility:

Contributors: Venkata Subba Rao Inti, Eeshwara Sai Tota

- The 8x8 American Checkers Board user interface that we created can be extended further to make 8x8 Chess Game or any other similar board games like Tic Toe Game. As we implemented the generic method to make the Game Board UI, it can be extended in the future to either small board games like 3x3 or large board games with 12x12 grids or other 8x8 board games. For any other extended board game, we just need to add a new class which mimic the AmericanCheckersGame.java like ChessGame.java or TicToeGame.java
- In order to add the new functionality to other extended games or to existing game then we just need to change the validMove() & makeMove() methods in our project. Which makes it easier to add the new features or new rules to the existing game.
- We want to extend the existing application further to host the application in the public domain and make it available for the users to play the game with another online player which we assumed to take atleast 20 hours to complete this module.
- To the existing game, we created private database connection to query tables in database. Further, we can add the new queries in other methods using the same database connection with username and password.
- We have implemented the database to store the user data when plays, once we extended the application to host online, we want to add the new features to database like storing the player moves data and displaying the top 10 statistics of the game.
- We have also implemented a class UserManagerGUI to follow Open Closed Principle. All the methods that we have in this class are declared as private which will be available for extension and closed for doing any modifications further. Similarly, we also have one key variable named bot which also follows OCP Principle.

Findings from the Code Review Exercise

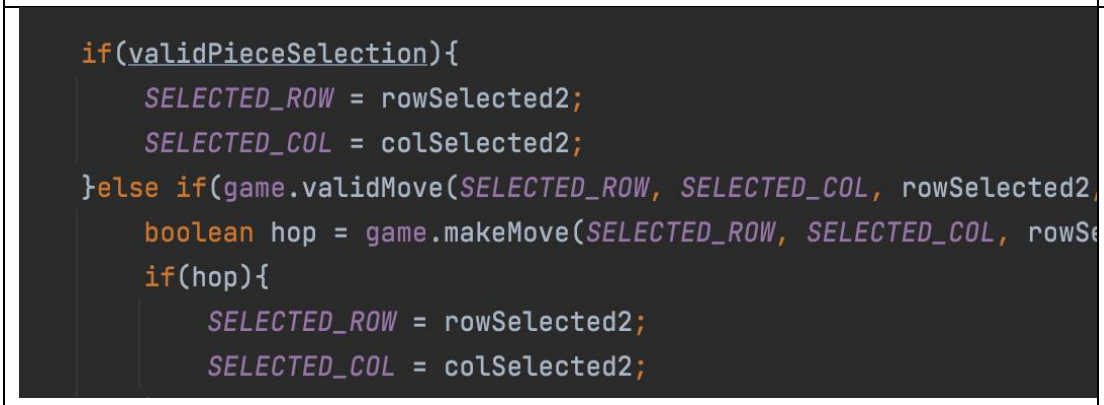
Use the following template to document the findings from the code review of **each** class.

Participant names: Eeshwara Sai Tota, Sai Kishore Reddy Surabhi, Venkata Subba Rao Inti
Class that was reviewed:

Check list	Checklist Item	Findings
Coding Standards	Naming conventions	• Naming convention for constants with multiple words is violated instead of using underscore between words, we have used a single word with capitalizing the first letter of the new word. • Test methods naming convention was not consistent as the class was

		developed by multiple developers in the team.
	Ordering convention of method arguments	The ordering convention was well maintained and whenever there were more than 4 arguments, an object is passed to keep the method simple.
	Meaningful and valid comments	Few unnecessary comments were removed and precondition and post condition for all complex methods were documented.
	Consistent style of code blocks	Yes, same style curly braces were used across the project, where open curly brace is written on the line where the function is declared,
	Consistent indentation	Whenever there were loops and blocks, a single tab space was used to indicate the block of code is nested.
	…	
Design Principles	Good class abstraction and interface	We have used private methods which restricts the visibility of all the sensitive data.
	Appropriate visibility of each variable, method, and class	Getters and setters are made public so that they can b used by UI class. Similarly, other variables and methods are maintained accordingly.
	Design by contract (pre/post-conditions)	Most of the public methods are used by UI classes and when the inputs are taken through mouse clicks, their output limits can be predicted, which makes the precondition reasonable and available
	Is the Open-Closed Principle violated?	When have implemented the open close principle in the UserManagerGUI.java and AmericanCheckersGame.java and it

		violated in other few methods of other classes.
	Is the Single Responsibility Principle violated?	No, single responsibility principle is not violated . American checkers can work independently as it contains pure functions. Similarly, user management works independently whose purpose is to register and login users.
Code Smells	Magic numbers	No magic numbers were used in our project as we used constant and local variables to check the conditions.
	Unnecessary global / class variable	No, there are no unnecessary global or class variables. Priority was given to make it a function variable when it is used by multiple methods, only then it is moved to class variable.
	Duplicate code	DRY principle was followed strictly while writing the code. So, Duplicate code is mostly avoided.
	Long methods	No, there are no long methods, whenever there is a chance, long methods were broken into multiple methods.
	Long parameter list	Whenever there were more than 4 parameters, a single object parameter was used to maintain simplicity.
	Over-complex expression	There is only one over complicated expression while writing algorithm for the Bot. however it is explained clearly with reasonable comments.
	Switch or if-then-else that needs to be replaced with polymorphism	No, as the project is meant to be a single game, there was no need to replace with polymorphism behavior. Polymorphism would

		make more sense when multiple different games need to be built using the core game mechanics.	
	Variable or method name whose intent is unclear	checkAndMakeMove methods intent is unclear but its intent is made clear though comments and method documentation	
	Any similar methods in other classes?	No similar methods are found when compared to other classes as each class has narrow functionality.	
	…		
Bugs	**Buggy code snippet**	**What is the bug?**	**Why is it a bug?**
		When no legal move can be made by player, but he has still pieces on board. The game is stuck.	This case should be handled and opponent should be declared winner in this case as well.

Source Code Summary

You must submit all source code. The team will receive no credit for sprint 3 if the source code is not submitted or the following tables are incomplete. A team member will receive no credit for sprint 3 if his/her name is not shown in the following tables.

Summary of all source code files and individual contributions

Source Code File Name	# Lines of code	Eeshwara Sai Tota contribution: #Lines of code	Venkata Subba Rao Inti contribution: #Lines of cod	Sai Kishore Reddy Surabhi contribution: #Lines of cod	Sai Mukesh Parvathaneni contribution: #Lines of cod	Hemanth Bandi contribution: #Lines of cod
Production Code						
AmericanCheckersGame	187	71	61	55	0	0
CheckersPiece	40	0	0	0	40	0
UserAccount	183	73	52	58	0	0
UserAccountManager	152	0	0	49	60	43
AmericanCheckersGUI	385	134	126	125	0	0
EditProfileWindow	169	0	0	0	67	102
UserManagerGUI	193	61	40	0	62	20
Test Code						
BotMoveTest	80	80	0	0	0	0
CheckersBoardTest	71	71	0	0	0	0
CheckMoveTest	83	0	83	0	0	0

EndGameTest	70	0	70	0	0	0
HopTest	55	0	55	0	0	0
KingPromotionTest	50	0	0	0	20	30
LoginTests	35	0	0	0	35	0
RegisterNewUserTests	57	0	0	0	20	37
BotGameTest	54	0	0	0	0	54
CheckersPiecePositionTest	30	0	0	30	0	0
CheckMoveTest	54	0	0	0	54	0
EmailAddressTest	30	0	0	30	0	0
EndGameTest	86	0	0	0	86	0
FirstNameTest	30	30	0	0	0	0
HopTest	62	0	0	62	0	0
KingPromotionTest	61	0	0	61	0	0
LastNameTests	37	0	0	0	0	37
PasswordTests	37	0	0	0	0	37
PhoneNumberTests	34	0	0	0	0	34
UserNameTests	31	0	0	0	0	31

Summary of each member's contribution

Team member	# total number of lines of code contributed	Program files to which the team member contributed
Eeshwara Sai Tota	520	AmericanCheckersGame, UserAccount, AmericanCheckersGUI, UserManagerGUI, BotMoveTest, CheckersBoardTest, FirstNameTest
Venkata Subba Rao Inti	487	AmericanCheckersGame, UserAccount, AmericanCheckersGUI, UserManagerGUI, CheckMoveTest, EndGameTest, HopTest
Sai Kishore Reddy Surabhi	470	AmericanCheckersGame, UserAccount, UserAccountManager, AmericanCheckersGUI, CheckersPiecePositionTest, EmailAddressTest, HopTest, KingPromotionTest
Sai Mukesh Parvathaneni	444	CheckersPiece, UserAccountManager, EditProfileWindow, UserManagerGUI, KingPromotionTest, LoginTests, RegisterNewUserTests, CheckMoveTest, EndGameTest
Hemanth Bandi	425	UserAccountManager, EditProfileWindow, UserManagerGUI, KingPromotionTest, RegisterNewUserTests, BotGameTest, LastNameTests, PasswordTests, PhoneNumberTests, UserNameTests

Meeting Minutes

Report the minutes of all meetings, including, but not limited to project/sprint planning meeting, stand-up meeting, backlog grooming, retrospective meeting, and pair programming session.

Date	**Time and Duration**	**Place**	**Participant Names**	**Purpose of the Meeting**	**Specific Action Items**

SPRINT 0					
10th September 2021	5 PM 30 Min	520 E Study Room	Eeshwara Sai Tota, Inti Venkata Subba Rao, Sai Kishore Reddy Surabhi, Sai Mukesh Parvathaneni, Hemanth Bandi	1. To understand skillset of every individual 2. Understand and play checkers game	1. Learn how to use Java GUI libraries. 2. Familiarize with git commands.
16th September 2021	3:30 PM 30 Min	520 E Study Room	Eeshwara Sai Tota, Inti Venkata Subba Rao, Sai Kishore Reddy Surabhi, Sai Mukesh Parvathaneni, Hemanth Bandi	Discuss implementation details & tools to be used	Creation of user stories
09-25-2021	5:30 PM, 3 Hours	Zoom Video Meeting	Sai Kishore Reddy Surabhi, Eeshwara Sai Tota, Venkata Subba Rao Inti, Sai Mukesh Parvathaneni, Hemanth Bandi	Discuss User Stories and Acceptance Criteria for the project.	Implement user stories, document user stories properly. • Implement Acceptance Criteria for each user story and document them properly.
SPRINT 1					
09-30-2021	5:30 PM, 4 Hours	520 E Study Room	Sai Kishore Reddy Surabhi, Eeshwara Sai Tota, Venkata Subba Rao Inti, Sai Mukesh Parvathaneni, Hemanth Bandi	Implement the production code in java and prepared one page description of the project.	Implement the production code for the documented user stories and acceptance criteria. • Implement graphical user interface (GUI) for the game with board and pieces.

10-07- 2021	5:30 PM, 4 Hours	Zoom Video Meeting	Sai Kishore Reddy Surabhi, Eeshwara Sai Tota, Venkata Subba Rao Inti, Sai Mukesh Parvathaneni , Hemanth Bandi	Implement test code and rectify any errors in the code.	Implement test code for the production code. • Rectify the errors in the production code using test code. • Prepare video description for the project
SPRINT 2					
10-14-2021	5:30 PM, 3 Hours	Zoom Video Meeting	Sai Kishore Reddy Surabhi, Eeshwara Sai Tota, Venkata Subba Rao Inti, Sai Mukesh Parvathaneni, Hemanth Bandi	Analyze and Design human vs human gameplay	• Piece should be highlighted when selected. • Normal Piece Movement for black and red
10-21-2021	5:30 PM, 4 Hours	520 E Study Room	Sai Kishore Reddy Surabhi, Eeshwara Sai Tota, Venkata Subba Rao Inti, Sai Mukesh Parvathaneni, Hemanth Bandi	Implement the production code in java for king promotion.	• Implement UI for king piece • Implement production code for king promotion when reached opponent's first row • Valid King piece movement
10-28-2021	5:30 PM, 4 Hours	Zoom Video Meeting	Sai Kishore Reddy Surabhi, Eeshwara Sai Tota, Venkata Subba Rao Inti, Sai Mukesh Parvathaneni, Hemanth Bandi	Implement production code for piece killing and end game.	• Implement production code for killing of opponent's piece • Show message when game has ended.

11-11-2021	5:30 PM, 4 Hours	Zoom Video Meeting	Sai Kishore Reddy Surabhi, Eeshwara Sai Tota, Venkata Subba Rao Inti, Sai Mukesh Parvathaneni, Hemanth Bandi	Implement toss functionality and test code	• Implemented production code for toss functionality. • Implemented test code and improved code coverage.
SPRINT 3					
11-17-2021	5:30 PM, 4 Hours	Zoom Video Meeting	Sai Kishore Reddy Surabhi, Eeshwara Sai Tota, Venkata Subba Rao Inti, Sai Mukesh Parvathaneni, Hemanth Bandi	Implementation of Human vs Computer game and testing the code by writing relevant test cases.	• Discussed on the algorithm for game between human and computer. • Implemented the game between human vs computer. • Tested the game between human vs computer by writing relevant test cases.
11-24-2021	5:30 PM, 4 Hours	520 E Study Room	Sai Kishore Reddy Surabhi, Eeshwara Sai Tota, Venkata Subba Rao Inti, Sai Mukesh Parvathaneni, Hemanth Bandi	Implementation of User Account Management using MySQL database and testing the database using relevant test cases.	• Implemented the User Account Management using MySQL database. • Implemented the testing code for the user account management using Junit.
12-1-2021	5:30 PM, 4 Hours	Zoom Video Meeting	Sai Kishore Reddy Surabhi, Eeshwara Sai Tota, Venkata Subba Rao Inti, Sai Mukesh Parvathaneni, Hemanth Bandi	Review of the whole project.	• Arranged a code review and noted down the findings. • Did relevant refactoring and completed final optimization of code for better performance. • Prepared suede code for human vs computer game and documented it.

12-6-2021	5:30 PM, 4 Hours	Zoom Video Meeting	Sai Kishore Reddy Surabhi, Eeshwara Sai Tota, Venkata Subba Rao Inti, Sai Mukesh Parvathaneni, Hemanth Bandi	Code Review Session, completed final documentation of project.	• Prepared class diagrams and noted down the explanation for it in the documentation. • Preparation of user interface design, discussed on extensibility

Buddy Rating

Rating Giver	Rating Receiver				
	Eeshwara Sai Tota	Inti Venkata Subba Rao	Sai Kishore Reddy Surabhi	Sai Mukesh Parvathaneni	Hemanth Bandi
Eeshwara Sai Tota	X	1	1	1	1
Inti Venkata Subba Rao	1	X	1	1	1
Sai Kishore Reddy Surabhi	1	1	X	1	1
Sai Mukesh Parvathaneni	1	1	1	X	1
Hemanth Bandi	1	1	1	1	X
Average	1	1	1	1	1