Snakes and Ladders with Questions

# Analysis

1. Web Game
2. Ask name of players
3. Normal snake and ladders board
4. Multiplayer
5. One Dice
6. Every dice roll a question is asked if correct, score is multiplied / added
7. Leader board

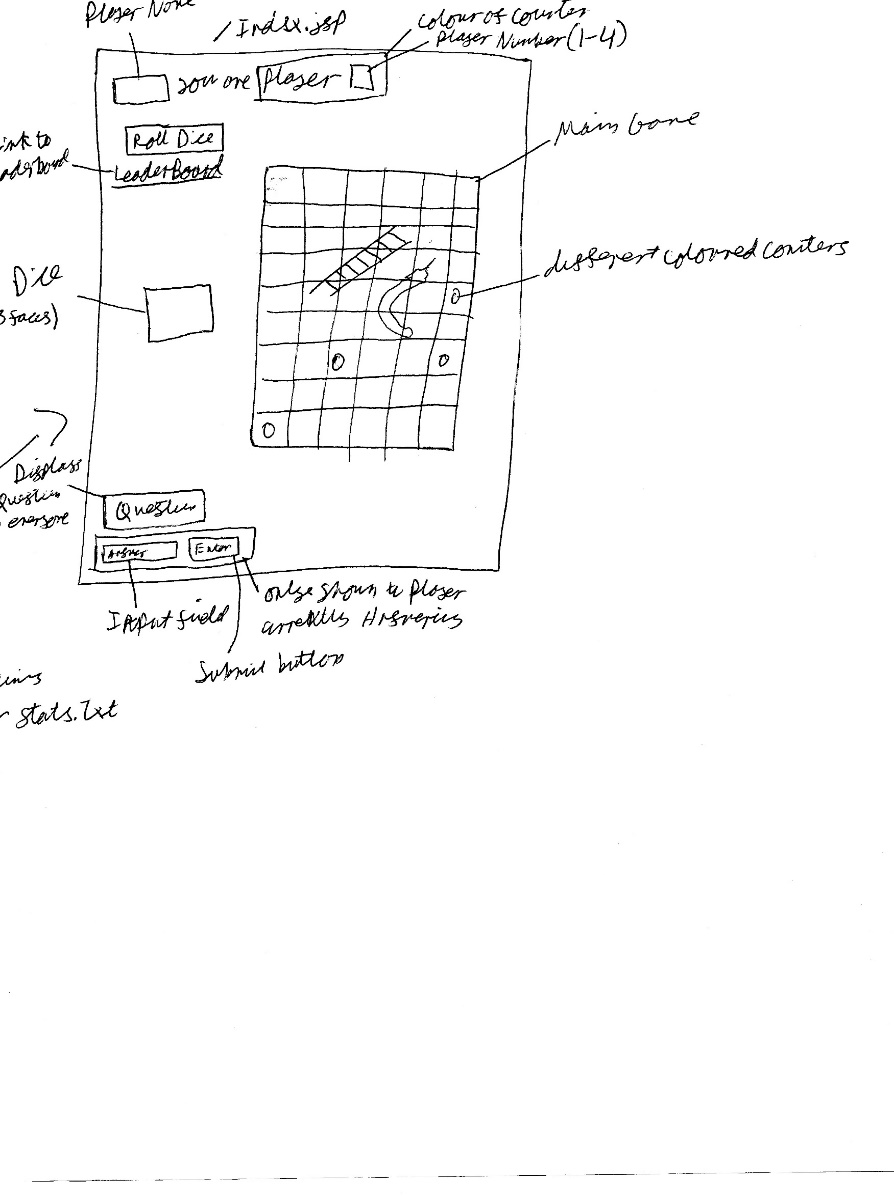
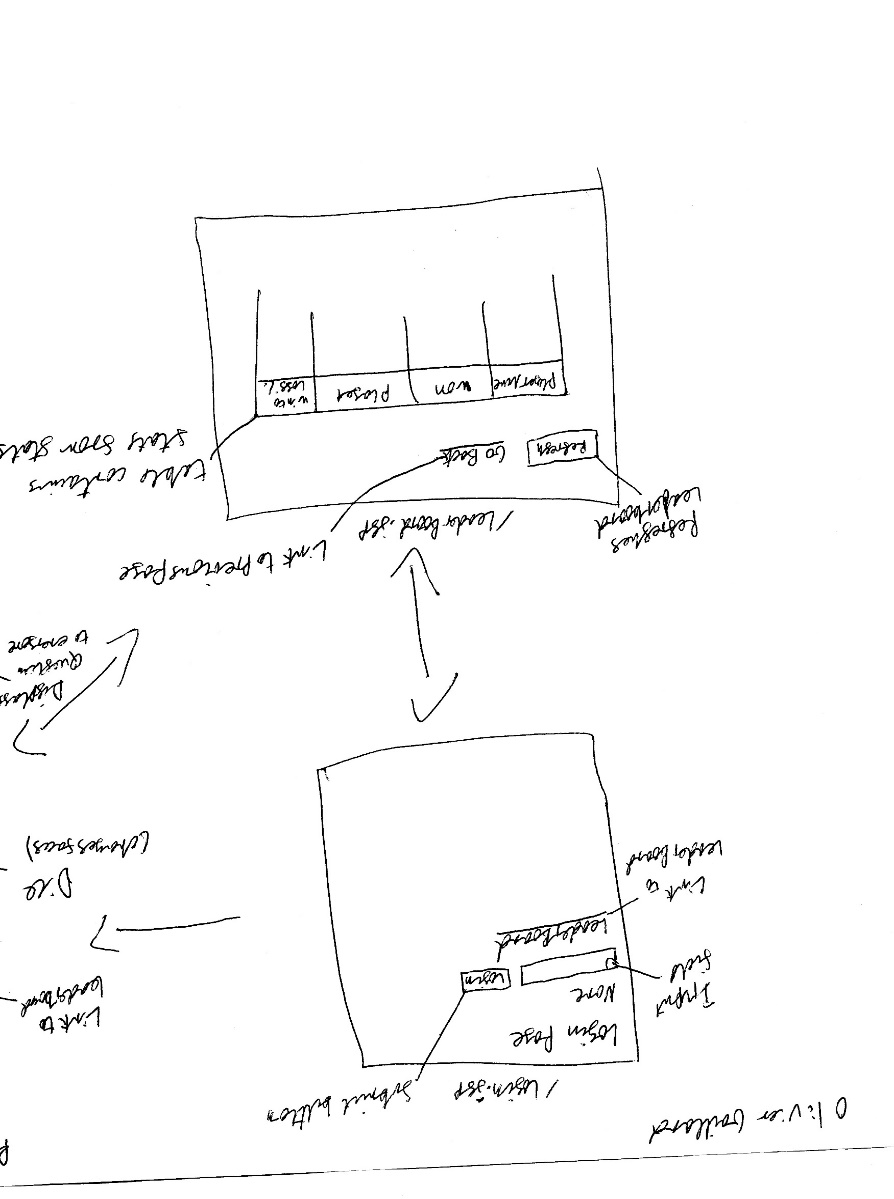
Further functions:

1. Choice of board size
2. Ability to have a 2 dice game
3. Bots that get a probability of answering a question correctly

Objectives:

1. Set up tomcat
2. Get a working canvas
3. Set a background image of canvas as snakes and ladders board
4. Ask how many players and then their names
5. Draw player pieces at the start different colour for each
6. Draw a dice, that can change faces (only 1 face seen)
7. Button to roll dice, disappears when clicked, header to show current players turn
8. Create a txt file with questions and the answer in the next row
9. Use alert and input fields for questions
10. Make a dice roll move players according to roll and if they answered correctly
11. If players land on a special square like going up a ladder etc make the actions work
12. Once question has been answered and time for next player to play, roll button reappears with changed header for that player
13. Create a leader board page based of a txt file
14. When a player wins add a win to the txt file
15. Once a player has won reset the game

# Design



# Technical Solution

Login.jsp

<html>  
<head>  
</head>  
<body>  
 <form action="LoginServlet" method="post">  
 <h1>Login Page</h1>  
 <p>Name</p>  
 <input type="text" placeholder="Name" pattern="[A-Z0-9a-z]+" name="Name" required>  
 <input type="submit" value="Login">  
 </form>  
<a href="LeaderBoard.jsp">Leader Board</a>  
  
<div></div>  
</body>  
</html>

LoginServlet.java

package servlet;  
  
import org.json.simple.JSONArray;  
import org.json.simple.JSONObject;  
import org.json.simple.parser.JSONParser;  
import org.json.simple.parser.ParseException;  
  
import javax.servlet.ServletException;  
import javax.servlet.annotation.WebServlet;  
import javax.servlet.http.HttpServlet;  
import javax.servlet.http.HttpServletRequest;  
import javax.servlet.http.HttpServletResponse;  
import java.io.FileReader;  
import java.io.FileWriter;  
import java.io.IOException;  
  
@WebServlet(  
 name = "LoginServlet",  
 urlPatterns = {"/LoginServlet"}  
)  
public class LoginServlet extends HttpServlet {  
  
 int PlayerCount = 0;  
  
 @Override  
 protected void doPost(HttpServletRequest request, HttpServletResponse response)  
 throws IOException, ServletException {  
  
 JSONParser parser = new JSONParser();  
  
  
 String name = request.getParameter("Name").trim();  
  
 JSONObject data = new JSONObject();  
 FileReader fileR = new FileReader("src/main/webapp/data.json");  
 try {  
 data = (JSONObject) parser.parse(fileR);  
 } catch (ParseException e) {  
 e.printStackTrace();  
 }  
 fileR.close();  
 String dest = "index.jsp";  
 if (PlayerCount <=3){  
 ((JSONObject)((JSONArray)data.get("Players")).get(PlayerCount)).put("name",name);  
 ((JSONObject)((JSONArray)data.get("Players")).get(PlayerCount)).put("cell",1);  
 request.getSession().setAttribute("name", name);  
 }else{  
 dest = "full.jsp";  
 }  
 FileWriter fileW = new FileWriter("src/main/webapp/data.json");  
 fileW.write(data.toString());  
 fileW.close();  
 PlayerCount = PlayerCount + 1;  
  
 response.sendRedirect(dest);  
  
  
  
 }  
}

Index.jsp

<html>  
<head>  
  
</head>  
<%  
 String nameS = (String) session.getAttribute("name");  
 try{  
 if(nameS.equals("null")){  
 response.sendRedirect("login.jsp");  
 }  
 }catch(NullPointerException e){  
 response.sendRedirect("login.jsp");  
 }  
 %>  
  
<script>  
  
 var UpdateInterval = null;  
 function updateMe(){  
 if(UpdateInterval != null){  
 window.clearInterval(UpdateInterval);  
 }  
 UpdateInterval = window.setInterval(update, 500);  
  
 }  
  
 var data = {};  
 function drawBG(data, ctx){  
 var img = new Image();  
 img.src = "t-n-1196-snakes-and-ladders-board-game\_ver\_1.png";  
 img.onload = function(){ctx.drawImage(img,150,75,700,700);  
  
 var img3 = new Image();  
 img3.src = "Dice\_"+data.Dice.face+".png";  
 img3.onload = function() {ctx.drawImage(img3, 40, 340,100, 100);}  
 var YourPlayerIndex;  
 var YourPlayerColour;  
 for (var i = 0;i<data.Players.length;i++){  
 var placement = data.Players[i].cell;  
 var row = (Math.floor((placement-1) / 10));  
 var col;  
  
 if(row%2 == 0){  
 col = 10-(placement-1)%10;  
 }else{  
 col = ((placement-1)%10)+1;  
 }  
  
 ctx.beginPath();  
 ctx.arc(850 + (20 \* (i%3)) - (col\*67.5), 750 - (20 \* Math.floor(i/3)) - (row\*67.5) , 10, 0, 2 \* Math.PI);  
 ctx.fillStyle = data.Players[i].colour;  
 ctx.fill();  
 if(data.Players[i].name == "${name}"){  
 YourPlayerIndex = i;  
 YourPlayerColour = data.Players[i].colour;  
 document.getElementById("question").innerHTML = data.Question.Q;  
 if (YourPlayerIndex == data.Turn){  
 document.getElementById("btn").style.display = "block";  
 if(data.Question.AD){  
 document.getElementById("btn2").style.display = "none";  
 document.getElementById("Answer").style.display = "none";  
 if(data.Question.AC){  
 document.getElementById("question").innerHTML = "You got it Correct +2";  
 }else{  
 document.getElementById("question").innerHTML = "You got it Wrong";  
 }  
 }else{  
 document.getElementById("btn2").style.display = "block";  
 document.getElementById("Answer").style.display = "block";  
 }  
  
 }else{  
  
 if(data.Question.AC && data.Question.AD){  
 document.getElementById("question").innerHTML = "<span style=\"color: "+data.Players[data.Turn].color+"\"> Player "+data.Players[data.Turn].name+"</span> got it Correct +2";  
 }else if(data.Question.AD){  
 document.getElementById("question").innerHTML = "<span style=\"color: "+data.Players[data.Turn].color+"\"> Player "+data.Players[data.Turn].name+"</span> got it Wrong";  
 }  
 document.getElementById("btn").style.display = "none";  
 document.getElementById("btn2").style.display = "none";  
 document.getElementById("Answer").style.display = "none";  
 }  
 }  
  
  
  
  
 }  
 document.getElementById("PlayerName").innerHTML = "${name} you are<span style=\"color: "+YourPlayerColour+"\"> Player "+(YourPlayerIndex + 1)+"</span>";  
 if (data.Winner !="none"){  
 document.getElementById("won").innerHTML = data.Winner + " won last Game"  
 }  
  
 }  
 }  
  
 function draw(dataTBD){  
 var canvas = document.getElementById('canvas');  
 var ctx = canvas.getContext('2d');  
 console.log(data);  
 canvas.width = 1000;  
 canvas.height = 900;  
 drawBG(data, ctx);  
 }  
  
 function roll(){  
 var xmlhttp = new XMLHttpRequest();  
 var url = "hello"+window.location.search;  
 xmlhttp.onreadystatechange = function() {  
 if (this.readyState == 4 && this.status == 200) {  
 data = JSON.parse(this.responseText);  
 draw(data);  
 }  
 };  
 xmlhttp.open("GET", url, true);  
 xmlhttp.send();  
 }  
  
 function update(){  
 var xmlhttp = new XMLHttpRequest();  
 var url = "update";  
 xmlhttp.onreadystatechange = function() {  
 if (this.readyState == 4 && this.status == 200) {  
 data = JSON.parse(this.responseText);  
 draw(data);  
 }  
 };  
 xmlhttp.open("GET", url, true);  
 xmlhttp.send();  
 }  
  
 function start(){  
 var xmlhttp=new XMLHttpRequest();  
 var url="start";  
 xmlhttp.onreadystatechange=function(){  
 if(this.readyState==4&&this.status==200){  
 data=JSON.parse(this.responseText);  
 draw(data);  
 }  
 };  
 xmlhttp.open("GET",url,true);  
 xmlhttp.send();  
 }  
  
 function Answered(){  
 var xmlhttp = new XMLHttpRequest();  
 var url = "hello";  
 xmlhttp.open("POST", url);  
 var params = document.getElementById("Answer").value;  
  
 xmlhttp.onreadystatechange = function() {  
 if (xmlhttp.readyState === 4){  
 data=JSON.parse(this.responseText);  
 console.log(data);  
 draw(data);  
 }  
 };  
 xmlhttp.send(JSON.stringify(params));  
 }  
  
</script>  
<div>  
 <h3 id="PlayerName"></h3>  
 <button id = "btn" onclick="roll();">Roll Dice</button>  
 <a href="LeaderBoard.jsp">Leader Board</a>  
 <p id="won"></p>  
</div>  
<body onload="start();updateMe();">  
 <canvas id="canvas">Game</canvas>  
 <h1 id="question"></h1>  
 <input id="Answer" type="text" pattern="[A-Z0-9a-z]+" placeholder="Answer" name="Answer">  
 <button id="btn2" onclick="Answered();">Enter</button>  
</body>  
</html>

StartServlet.java

package servlet;  
  
import org.json.simple.JSONObject;  
import org.json.simple.parser.JSONParser;  
import org.json.simple.parser.ParseException;  
  
import javax.servlet.annotation.WebServlet;  
import javax.servlet.http.HttpServlet;  
import javax.servlet.http.HttpServletRequest;  
import javax.servlet.http.HttpServletResponse;  
import java.io.File;  
import java.io.FileReader;  
import java.io.FileWriter;  
import java.io.IOException;  
import java.util.Random;  
import java.util.Scanner;  
  
@WebServlet(  
 name = "StartServlet",  
 urlPatterns = {"/start"}  
)  
public class StartServlet extends HttpServlet{  
  
 JSONObject data;  
 JSONParser parser = new JSONParser();  
  
 @Override  
 protected void doGet(HttpServletRequest req, HttpServletResponse resp)  
 throws IOException {  
 try {  
 Random rnd = new Random();  
 FileReader fileR = new FileReader("src/main/webapp/data.json");  
 data = (JSONObject) parser.parse(fileR);  
 fileR.close();  
 File file = new File("src/main/webapp/Questions.txt");  
 Scanner myReader = new Scanner(file);  
 int QNOo = rnd.nextInt(3) + 1;  
 for (int i = 0; i < QNOo; i++) {  
 myReader.nextLine();  
 }  
 String[] question = myReader.nextLine().split(",");  
 myReader.close();  
  
 ((JSONObject) data.get("Question")).put("Q", question[0]);  
 ((JSONObject) data.get("Question")).put("A", question[1]);  
 ((JSONObject) data.get("Question")).put("AD", false);  
 ((JSONObject) data.get("Question")).put("AC", false);  
  
 FileWriter fileW = new FileWriter("src/main/webapp/data.json");  
 fileW.write(data.toString());  
 fileW.close();  
  
 } catch (ParseException e) {  
 e.printStackTrace();  
 }  
  
 resp.getWriter().write(String.*valueOf*(data));  
  
 }  
  
}

HelloServlet.java

package servlet;  
  
import org.json.simple.JSONArray;  
import org.json.simple.JSONObject;  
import org.json.simple.parser.JSONParser;  
import org.json.simple.parser.ParseException;  
  
import javax.servlet.ServletException;  
import javax.servlet.annotation.WebServlet;  
import javax.servlet.http.HttpServlet;  
import javax.servlet.http.HttpServletRequest;  
import javax.servlet.http.HttpServletResponse;  
import java.io.\*;  
import java.util.ArrayList;  
import java.util.Random;  
import java.util.Scanner;  
  
@WebServlet(  
 name = "HelloServlet",  
 urlPatterns = {"/hello"}  
 )  
public class HelloServlet extends HttpServlet {  
 JSONParser parser = new JSONParser();  
 boolean winner = false;  
  
 @Override  
 protected void doGet(HttpServletRequest req, HttpServletResponse resp)  
 throws ServletException, IOException {  
 Random rnd = new Random();  
 JSONObject object = new JSONObject();  
 try {  
 FileReader fileR = new FileReader("src/main/webapp/data.json");  
 object = (JSONObject) parser.parse(fileR);  
 fileR.close();  
 } catch (ParseException e) {  
 e.printStackTrace();  
 }  
  
 int roll = rnd.nextInt(6)+1;  
 int turn = ((Long) object.get("Turn")).intValue();  
 int CCell = ((Long) ((JSONObject)((JSONArray) object.get("Players")).get(turn)).get("cell")).intValue();  
 int NCell = CCell + roll;  
 if( (boolean) ((JSONObject)object.get("Question")).get("AC")){  
 NCell = NCell +2;  
 }  
 ((JSONObject)object.get("Question")).put("AD",false);  
 ((JSONObject)object.get("Question")).put("AD",false);  
 *//handle Snakes and Ladders* if(NCell ==19){  
 NCell =4;  
 }else if(NCell ==6){  
 NCell = 24;  
 }else if(NCell ==13){  
 NCell = 7;  
 }else if(NCell ==10){  
 NCell = 12;  
 }else if(NCell ==20){  
 NCell = 38;  
 }else if(NCell ==11){  
 NCell = 33;  
 }else if(NCell ==48){  
 NCell = 14;  
 }else if(NCell ==40){  
 NCell = 59;  
 }else if(NCell ==57){  
 NCell = 36;  
 }else if(NCell ==45){  
 NCell = 54;  
 }else if(NCell ==64){  
 NCell = 78;  
 }else if(NCell ==68){  
 NCell = 49;  
 }else if(NCell ==72){  
 NCell = 91;  
 }else if(NCell ==86){  
 NCell = 96;  
 }else if(NCell ==98){  
 NCell = 84;  
 }else if(NCell ==83){  
 NCell = 61;  
 }else if(NCell ==87){  
 NCell = 66;  
 }else if(NCell ==94){  
 NCell = 88;  
 }else if(NCell >=100){  
 System.*out*.println("Player "+(turn +1) + " wins!!");  
 winner = true;  
 }  
  
 File file = new File("src/main/webapp/Questions.txt");  
 Scanner myReader = new Scanner(file);  
  
 int QNOo= rnd.nextInt(5)+1;  
 for (int i =0;i<QNOo;i++){  
 myReader.nextLine();  
 }  
 String[] question = myReader.nextLine().split(",");  
 myReader.close();  
  
 ((JSONObject)((JSONArray) object.get("Players")).get(turn)).put("cell",NCell);  
  
 ((JSONObject)object.get("Question")).put("Q",question[0]);  
 ((JSONObject)object.get("Question")).put("A",question[1]);  
 ((JSONObject)object.get("Question")).put("AD",false);  
  
 ((JSONObject)object.get("Dice")).put("face",roll);  
 int turnf = turn;  
 turn = (turn +1)%((JSONArray)object.get("Players")).size();  
 object.put("Turn",turn);  
 JSONObject data;  
 data = object;  
  
 if(winner){  
 FileReader fileR = new FileReader("src/main/webapp/data.json");  
 try {  
 data = (JSONObject) parser.parse(fileR);  
 } catch (ParseException e) {  
 e.printStackTrace();  
 }  
 fileR.close();  
  
 ((JSONObject) data.get("Question")).put("Q", question[0]);  
 ((JSONObject) data.get("Question")).put("A", question[1]);  
 ((JSONObject) data.get("Question")).put("AD", false);  
 ((JSONObject) data.get("Question")).put("AC", false);  
 ((JSONObject)((JSONArray) data.get("Players")).get(0)).put("cell",1);  
 ((JSONObject)((JSONArray) data.get("Players")).get(1)).put("cell",1);  
 ((JSONObject)((JSONArray) data.get("Players")).get(2)).put("cell",1);  
 ((JSONObject)((JSONArray) data.get("Players")).get(3)).put("cell",1);  
  
 *//Update leaderboard* BufferedReader fileR2 = new BufferedReader(new FileReader("src/main/webapp/stats.txt"));  
 ArrayList<String[]> stats = new ArrayList<>();  
 ArrayList<String> PlayerNames = new ArrayList<>();  
 ArrayList<String> PlayerNames2 = new ArrayList<>();  
 for(int i=0;i<4;i++){  
 PlayerNames.add((String) ((JSONObject)((JSONArray) object.get("Players")).get(i)).get("name"));  
 PlayerNames2.add((String) ((JSONObject)((JSONArray) object.get("Players")).get(i)).get("name"));  
 }  
 boolean foundW = false;  
 String winnerName = PlayerNames.get(turnf);  
 String line = fileR2.readLine();  
 while(line != null){  
 String[] temp = line.split(",");  
 if(temp[0].equals(PlayerNames2.get(turnf))){  
 foundW = true;  
 *//Winning Index is 1* temp[1] = String.*valueOf*(Integer.*parseInt*(temp[1]) + 1);  
 }  
 if(temp[0].equals(PlayerNames2.get(0)) || temp[0].equals(PlayerNames2.get(1)) ||temp[0].equals(PlayerNames2.get(2))|| temp[0].equals(PlayerNames2.get(3))){  
 temp[2] = String.*valueOf*(Integer.*parseInt*(temp[2]) +1);  
 PlayerNames.remove(temp[0]);  
 }  
 stats.add(temp);  
 line = fileR2.readLine();  
 }  
 fileR2.close();  
 if(!foundW){  
 stats.add(new String[]{PlayerNames2.get(turnf), "1", "1"});  
 PlayerNames.remove(winnerName);  
 }  
 for(int i=0;i<PlayerNames.size();i++){  
 stats.add(new String[]{PlayerNames.get(i),"0","1"});  
 }  
 data.put("Turn",0);  
  
 *//Write new stats to stats no* BufferedWriter fileW2 = new BufferedWriter(new FileWriter("src/main/webapp/stats.txt"));  
 for(int i =0;i<stats.size();i++){  
 fileW2.write(stats.get(i)[0]+","+stats.get(i)[1]+","+stats.get(i)[2]);  
 fileW2.newLine();  
 }  
 fileW2.flush();  
 fileW2.close();  
 *//Why dont you work you piece of* data.put("Winner",winnerName);  
 winner = false;  
 }  
  
 FileWriter fileW = new FileWriter("src/main/webapp/data.json");  
 fileW.write(data.toString());  
 fileW.close();  
  
 resp.getWriter().write(String.*valueOf*(data));  
  
 }  
 @Override  
 protected void doPost(HttpServletRequest req, HttpServletResponse resp)  
 throws IOException, ServletException {  
  
 StringBuilder buffer = new StringBuilder();  
 BufferedReader reader = req.getReader();  
 String line;  
 while ((line = reader.readLine()) != null) {  
 buffer.append(line);  
 buffer.append(System.*lineSeparator*());  
 }  
  
 String Answer = buffer.toString().replaceAll("\"","").trim().toLowerCase();  
  
 System.*out*.println(Answer);  
 JSONObject data = new JSONObject();  
 FileReader fileR = new FileReader("src/main/webapp/data.json");  
 try {  
 data = (JSONObject) parser.parse(fileR);  
 } catch (ParseException e) {  
 e.printStackTrace();  
 }  
 fileR.close();  
  
 ((JSONObject)data.get("Question")).put("AD",true);  
  
 if(((String)((JSONObject)data.get("Question")).get("A")).equals(Answer)){  
 ((JSONObject)data.get("Question")).put("AC",true);  
 }else{  
 ((JSONObject)data.get("Question")).put("AC",false);  
 }  
  
 FileWriter fileW = new FileWriter("src/main/webapp/data.json");  
 fileW.write(data.toString());  
 fileW.close();  
  
 resp.getWriter().write(String.*valueOf*(data));  
  
 }  
  
}

UpdateServlet.java

package servlet;  
  
  
import org.json.simple.JSONObject;  
import org.json.simple.parser.JSONParser;  
import org.json.simple.parser.ParseException;  
  
import javax.servlet.annotation.WebServlet;  
import javax.servlet.http.HttpServlet;  
import javax.servlet.http.HttpServletRequest;  
import javax.servlet.http.HttpServletResponse;  
import java.io.FileReader;  
import java.io.IOException;  
  
  
@WebServlet(  
 name = "UpdateServlet",  
 urlPatterns = {"/update"}  
)  
public class UpdateServlet extends HttpServlet {  
  
 JSONObject data;  
 JSONParser parser = new JSONParser();  
  
 @Override  
 protected void doGet(HttpServletRequest req, HttpServletResponse resp)  
 throws IOException {  
  
 try {  
 FileReader fileR = new FileReader("src/main/webapp/data.json");  
 data = (JSONObject) parser.parse(fileR);  
 fileR.close();  
 }catch (ParseException e){  
 e.printStackTrace();  
 }  
  
 resp.getWriter().write(String.*valueOf*(data));  
  
 }  
  
}

LeaderBoard.jsp

<html>  
<head>  
</head>  
<style>  
table, tr, th{  
 border: 1px solid black;  
 border-collapse: collapse;  
 width: 800px;  
 height: 40px;  
}  
  
  
</style>  
<script>  
  
 function draw(data){  
 document.getElementById("LB").innerHTML = data;  
 }  
  
 function refresh(){  
 var xmlhttp=new XMLHttpRequest();  
 var url="LB";  
 xmlhttp.onreadystatechange=function(){  
 if(this.readyState==4&&this.status==200){  
 data=this.responseText;  
 draw(data);  
 }  
 };  
 xmlhttp.open("GET",url,true);  
 xmlhttp.send();  
 }  
  
  
</script>  
<body onload="refresh()">  
<button id = "btn" onclick="refresh();">Refresh</button>  
<a href="index.jsp">Go Back</a>  
<div><h1></h1></div>  
<table id="LB"></table>  
<div></div>  
</body>  
</html>

LeaderBoardServlet.java

package servlet;  
  
  
import javax.servlet.ServletException;  
import javax.servlet.annotation.WebServlet;  
import javax.servlet.http.HttpServlet;  
import javax.servlet.http.HttpServletRequest;  
import javax.servlet.http.HttpServletResponse;  
import java.io.BufferedReader;  
import java.io.FileReader;  
import java.io.IOException;  
  
  
@WebServlet(  
 name = "LeaderBoardServlet",  
 urlPatterns = {"/LB"}  
)  
public class LeaderBoardServlet extends HttpServlet {  
  
 String data;  
 String line;  
  
 @Override  
 protected void doGet(HttpServletRequest req, HttpServletResponse resp)  
 throws ServletException, IOException {  
  
 BufferedReader fileR = new BufferedReader(new FileReader("src/main/webapp/stats.txt"));  
 data = "<tr><th>Player Name</th><th>Games Won</th><th>Games Played</th><th>Win to Loss %</th></tr>";  
 while((line = fileR.readLine()) != null){  
 String[] linesplit = line.split(",");  
 String lastcol = String.*valueOf*(Math.*round*((Float.*parseFloat*(linesplit[1])/ Float.*parseFloat*(linesplit[2]))\*100));  
 data = data +"<tr><th>"+linesplit[0]+"</th><th>"+linesplit[1]+"</th><th>"+linesplit[2]+"</th><th>"+ lastcol +"</th></tr>";  
 }  
  
 data = data.trim();  
 fileR.close();  
 resp.getWriter().write(data);  
  
 }  
  
}

full.jsp

<html>  
<head>  
</head>  
<body>  
  
<h1>The Game Is Full</h1>  
<a href="login.jsp">Log In</a>  
<div></div>  
</body>  
</html>

Questions.txt

1 + 1 = ?,2  
2 + 2 = ?,4  
3 + 3 = ?,6  
4 + 4 = ?,8  
5 + 5 = ?,10  
6 + 6 = ?,12

Stats.txt

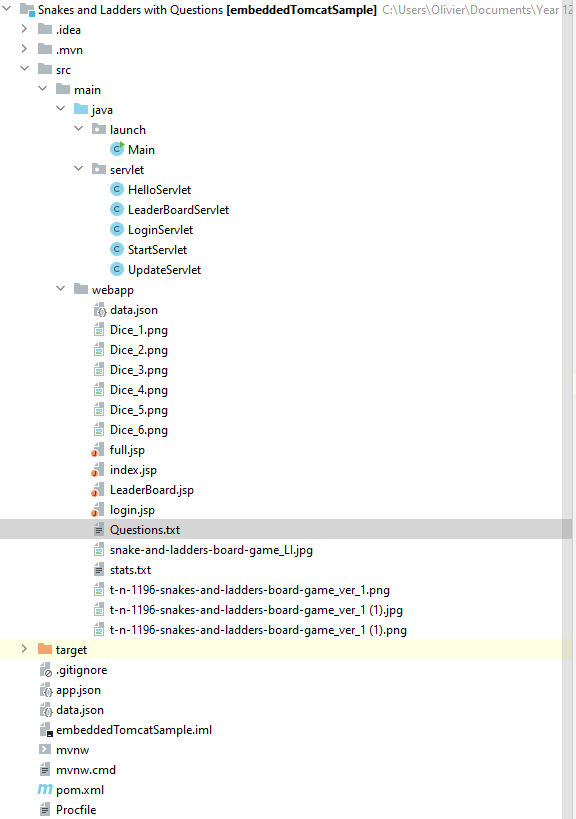
3,11,14  
1,1,14  
2,2,14  
4,0,11  
fdsfse,0,1  
vbgbfrejhgjgfe,0,2  
root,98,100  
root3,1,1  
Bean,1,1  
Loko,0,1  
Gen,0,1  
AA,0,1

Main.java

package launch;  
  
import org.apache.catalina.WebResourceRoot;  
import org.apache.catalina.WebResourceSet;  
import org.apache.catalina.core.StandardContext;  
import org.apache.catalina.startup.Tomcat;  
import org.apache.catalina.webresources.DirResourceSet;  
import org.apache.catalina.webresources.EmptyResourceSet;  
import org.apache.catalina.webresources.StandardRoot;  
  
import java.io.File;  
import java.net.URISyntaxException;  
import java.nio.file.Files;  
import java.nio.file.Path;  
  
public class Main {  
  
 private static File getRootFolder() {  
 try {  
 File root;  
 String runningJarPath = Main.class.getProtectionDomain().getCodeSource().getLocation().toURI().getPath().replaceAll("\\\\", "/");  
 int lastIndexOf = runningJarPath.lastIndexOf("/target/");  
 if (lastIndexOf < 0) {  
 root = new File("");  
 } else {  
 root = new File(runningJarPath.substring(0, lastIndexOf));  
 }  
 System.*out*.println("application resolved root folder: " + root.getAbsolutePath());  
 return root;  
 } catch (URISyntaxException ex) {  
 throw new RuntimeException(ex);  
 }  
 }  
  
 public static void main(String[] args) throws Exception {  
  
 File root = *getRootFolder*();  
 System.*setProperty*("org.apache.catalina.startup.EXIT\_ON\_INIT\_FAILURE", "true");  
 Tomcat tomcat = new Tomcat();  
 Path tempPath = Files.*createTempDirectory*("tomcat-base-dir");  
 tomcat.setBaseDir(tempPath.toString());  
  
 *//The port that we should run on can be set into an environment variable  
 //Look for that variable and default to 8080 if it isn't there.* String webPort = System.*getenv*("PORT");  
 if (webPort == null || webPort.isEmpty()) {  
 webPort = "8081";  
 }  
  
 tomcat.setPort(Integer.*valueOf*(webPort));  
 File webContentFolder = new File(root.getAbsolutePath(), "src/main/webapp/");  
 if (!webContentFolder.exists()) {  
 webContentFolder = Files.*createTempDirectory*("default-doc-base").toFile();  
 }  
 StandardContext ctx = (StandardContext) tomcat.addWebapp("", webContentFolder.getAbsolutePath());  
 *//Set execution independent of current thread context classloader (compatibility with exec:java mojo)* ctx.setParentClassLoader(Main.class.getClassLoader());  
  
 System.*out*.println("configuring app with basedir: " + webContentFolder.getAbsolutePath());  
  
 *// Declare an alternative location for your "WEB-INF/classes" dir  
 // Servlet 3.0 annotation will work* File additionWebInfClassesFolder = new File(root.getAbsolutePath(), "target/classes");  
 WebResourceRoot resources = new StandardRoot(ctx);  
  
 WebResourceSet resourceSet;  
 if (additionWebInfClassesFolder.exists()) {  
 resourceSet = new DirResourceSet(resources, "/WEB-INF/classes", additionWebInfClassesFolder.getAbsolutePath(), "/");  
 System.*out*.println("loading WEB-INF resources from as '" + additionWebInfClassesFolder.getAbsolutePath() + "'");  
 } else {  
 resourceSet = new EmptyResourceSet(resources);  
 }  
 resources.addPreResources(resourceSet);  
 ctx.setResources(resources);  
  
 tomcat.start();  
 tomcat.getServer().await();  
 }  
}

data.json

{"Dice":{"face":6},"Turn":0,"Question":{"Q":"4 + 4 = ?","A":"8","AC":false,"AD":false},"Winner":"none","Players":[{"colour":"#F13710","name":"","cell":1},{"colour":"#10E3F1","name":"","cell":1},{"colour":"#2CF907","name":"","cell":1},{"colour":"#B307F9","name":"","cell":1}]}



JSP pages mostly receive data and it is shown to the user in a user-friendly way (GUI) while Servlet classes handle what data is sent to JSP pages according to how the user interacts with the GUI.

The User first logs on choosing a name no special characters are allowed, that name is stored as a session attribute and the server records their name, they are redirected to index.jsp, here is the main game you have a dice, board, counters and a shown question, depending on if it is your turn or not there is also a prompt of where to answer the question and a roll dice button. There is also a link to see the leaderboard for players.

Index.jsp draws the abstract data (images and text) it has received from an AJAX request stored locally as ‘data’ and sending the appropriate calls to methods in servlets depending on user input. It regularly calls theUpdateServlet so that the game is realtime.

The StartServlet handles the start of a game after just logging in, it sets up flags and the 1st question to be answered.

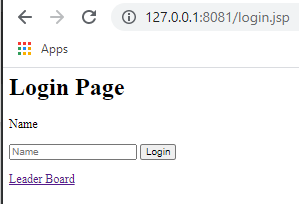
The HelloServlet doGet(), handles the Main Game, from deciding in which cell the player is going to land on to generating questions for the user to answer. Most notably it writes all the data to data.json file so that the UpdateServlet can function. It also handles updating stats.txt when a player wins a game. Its other method doPost(), handles when a user answers a question, it checks if the answer is correct or not and updates data.json accordingly.

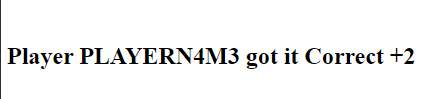
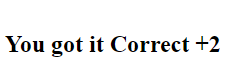
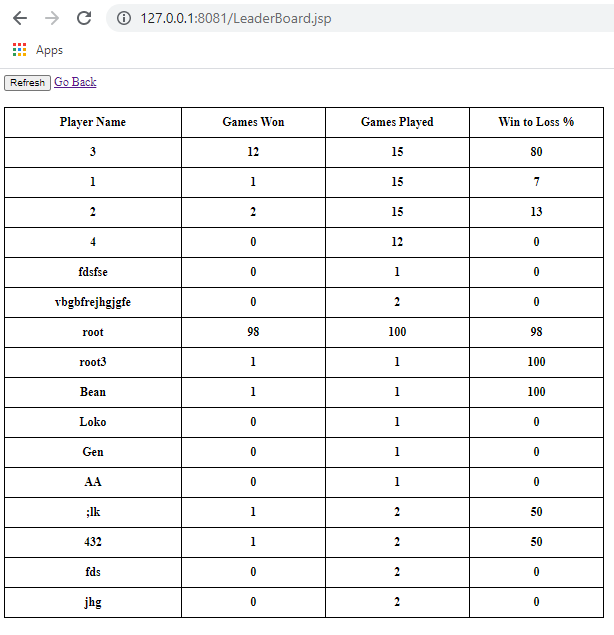
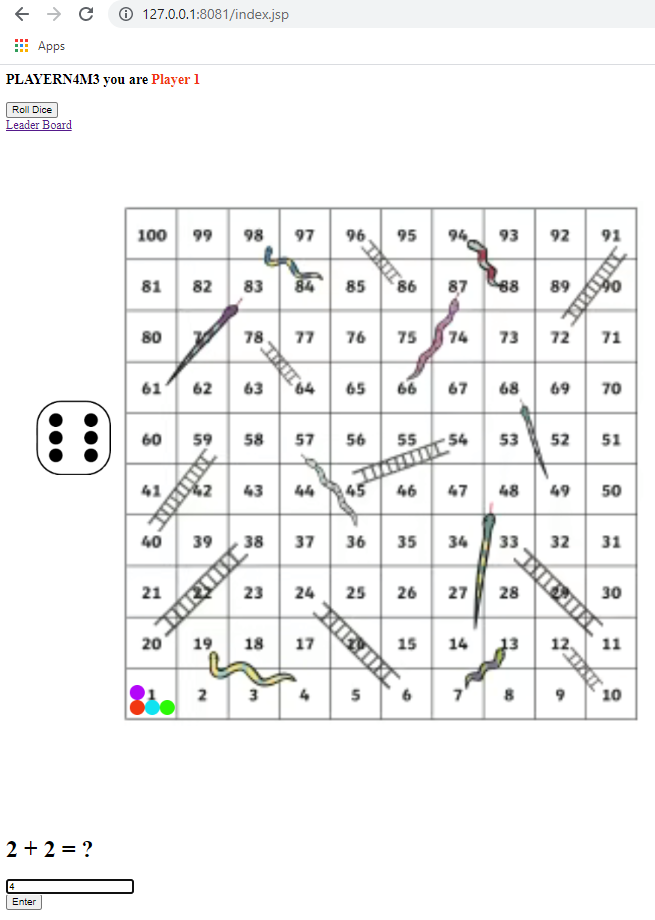
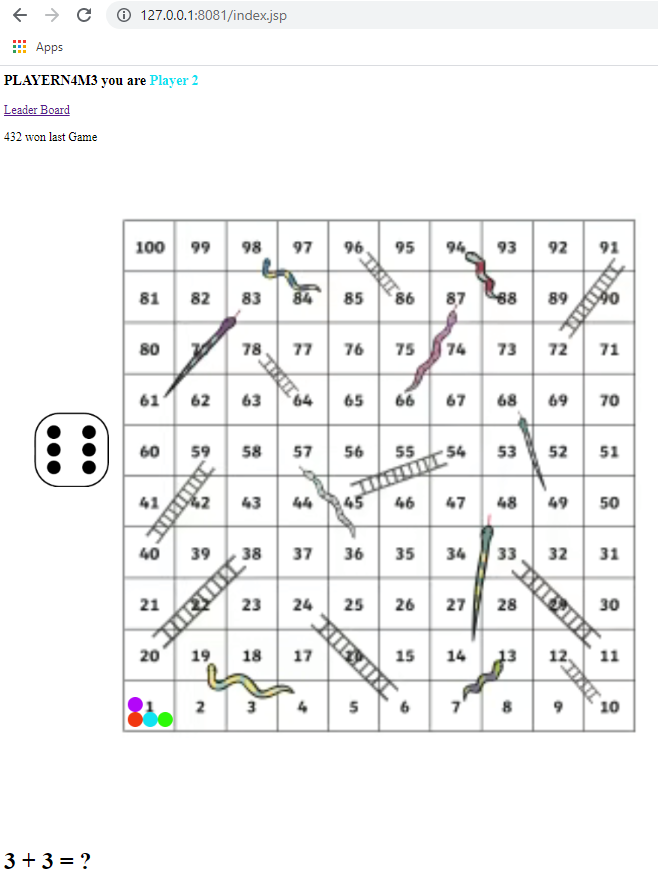
The UpdateServlet is used to retrieve the updated board state this is called upon by use of an AJAX request from index.jsp, this is done as it is a multiplayer game and any actions not done by the user but by other users must be shown to the user to see how the game is progressing.

The LeaderBoard.jsp simply inputs the data received and styles it from an AJAX request to the LeaderBoardServlet into a table format for ease of use.

The LeaderBoardServlet handles reading the data from stats.txt and formatting it for HTML use before passing it on to LeaderBoard.jsp.

Assets like Dice\_$num.png are used to show the user what is happening to the board state as the game progresses.





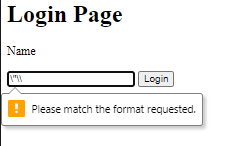
When another player gets a question correct, both are shown instead of the question. Similar message if you get question Wrong.

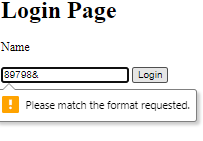
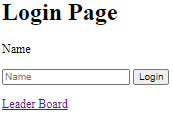
When you get a question correct

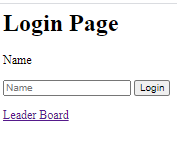
# Testing

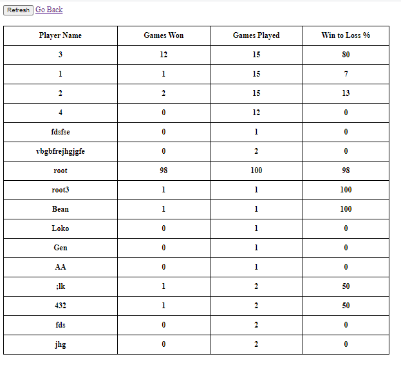
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test No. | Description | What to do | Expected Outcome | Actual Outcome | Screenshot evidence  Reference |
| 1 | The name cannot use special characters or spaces to exploit code somewhere else | Enter many different characters into the Login Page | The page will not accept anything other than numbers and normal letters | Alert comes up and tells user to match formatted request | 1 |
| 2 | References in GUI go to the correct pages | Click on all links to other pages | You are Redirected to the wanted page | All references work perfectly well | 2, 3,10 |
| 3 | Testing all buttons | Click on all the buttons and see how they act | Roll Dice – should make the player move and then disappear,  Enter- Question response should be sent to server and checked and then a message should be displayed to reflect answer to the Question.  Refresh- leader board should be refreshed.  Login – logs you in and redirects you to main game. | The buttons work as they should | 4,5,6,7,8,9 |
| 4 | Test winning updating Leader Board | Win a game, record who wins and the names of all players, check that the stats have been modified correctly | Winner should get 1 more win, all players should get 1 more game played | Stats.txt is updated correctly as needed and this is reflected on the leader board correctly | 11,12 |
| 5 | Testing the multiplayer functionality | Check that all Players have the same board state after a player does something | Rolled the dice on Player 1 should see a change in dice face and position of player 1 | Correct position and dice face show on Player 3 shows that game is updating real-time | 13 |

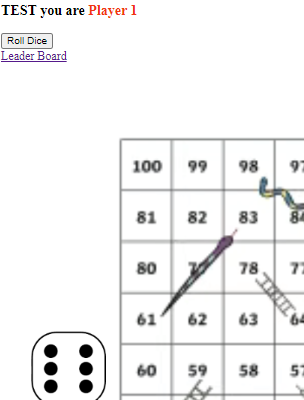
# Screenshot References

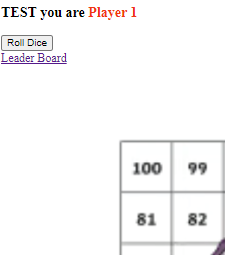
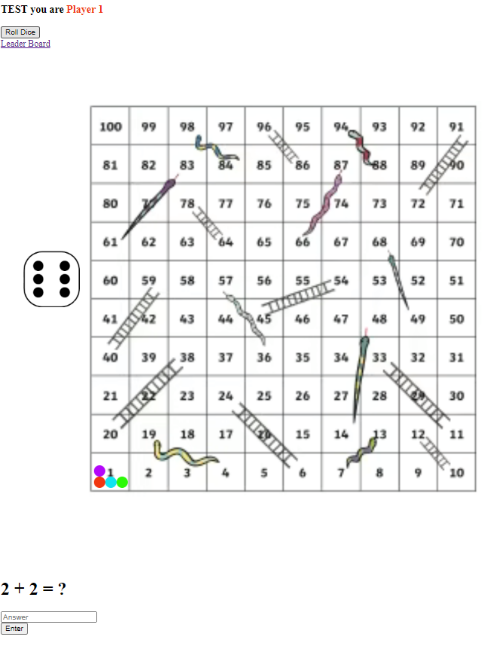
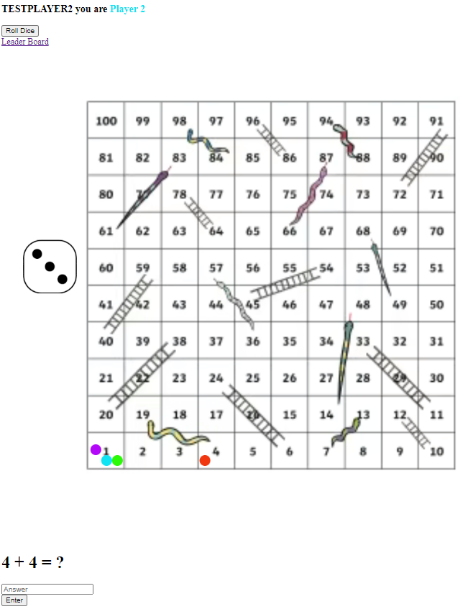
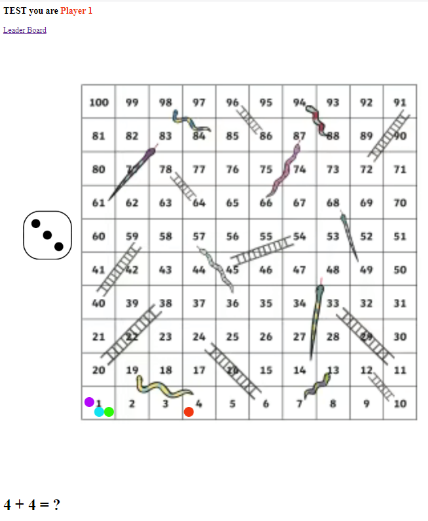
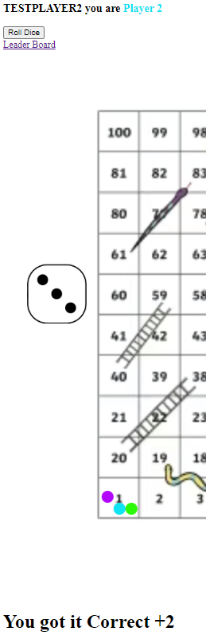
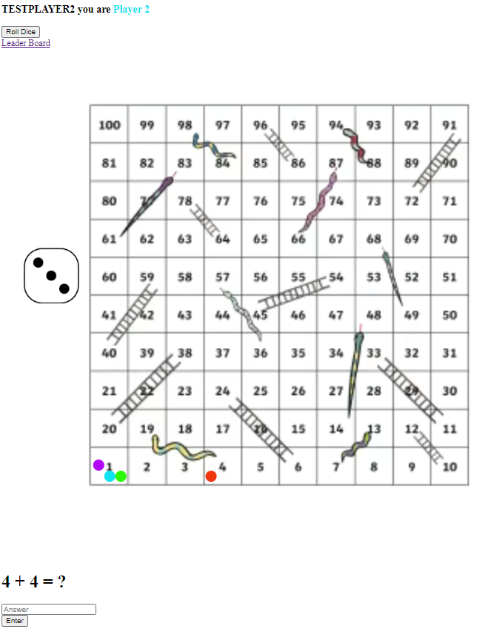
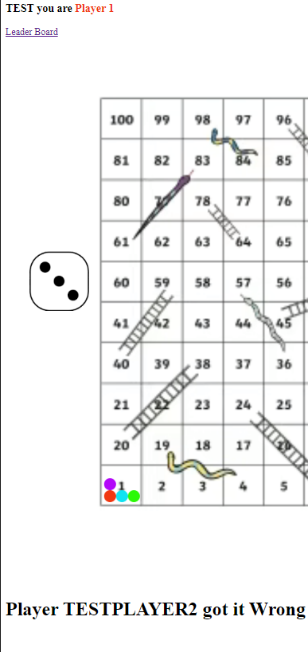
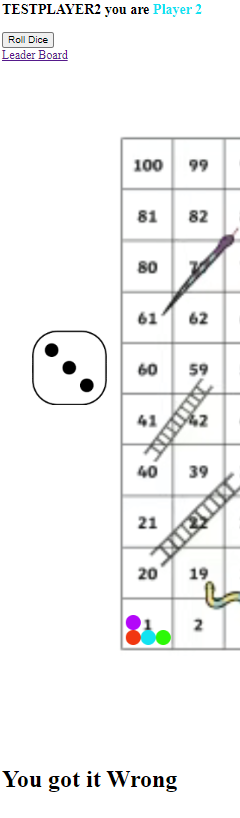
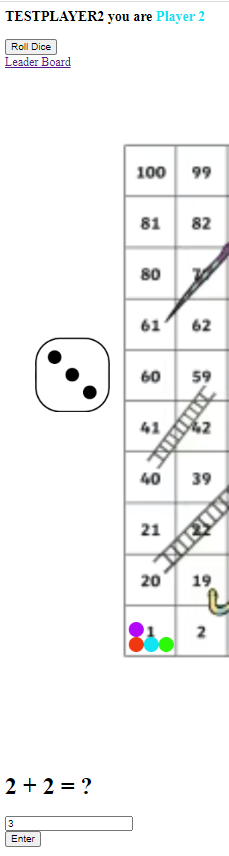
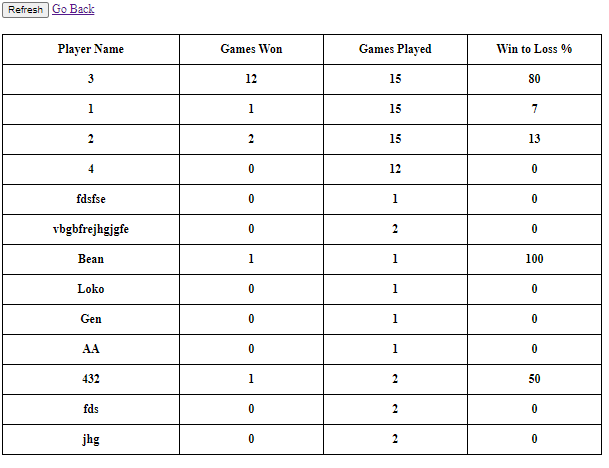
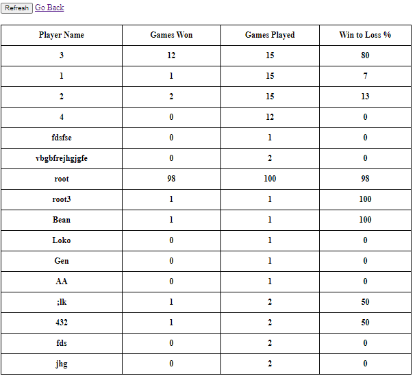
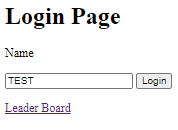
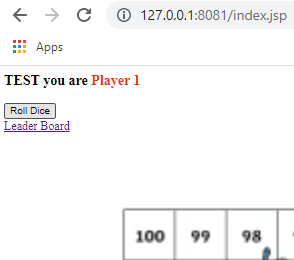
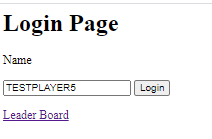


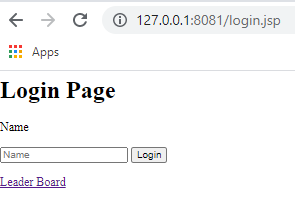
1. 

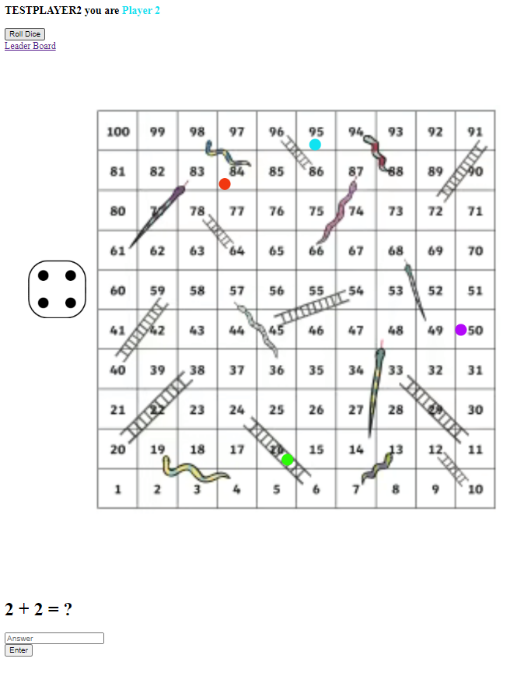
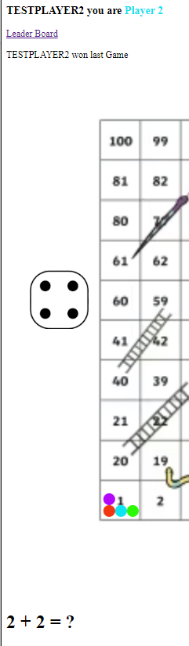
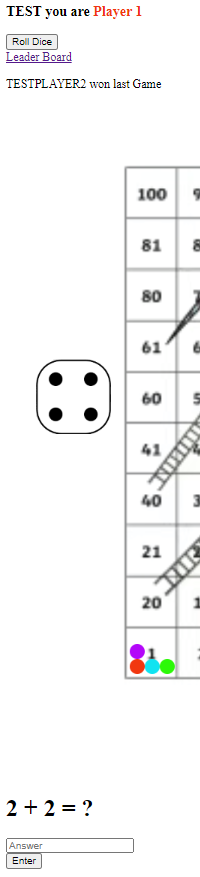
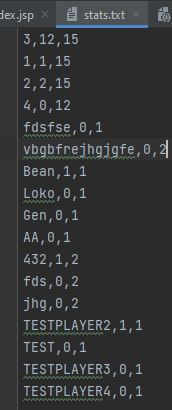


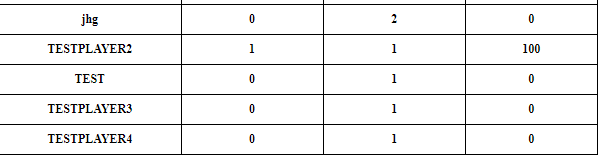
1. 

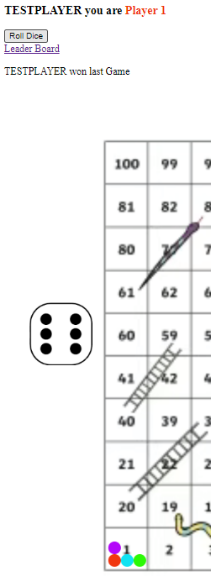


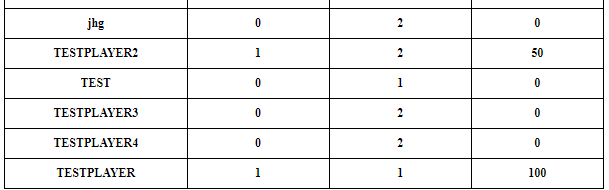
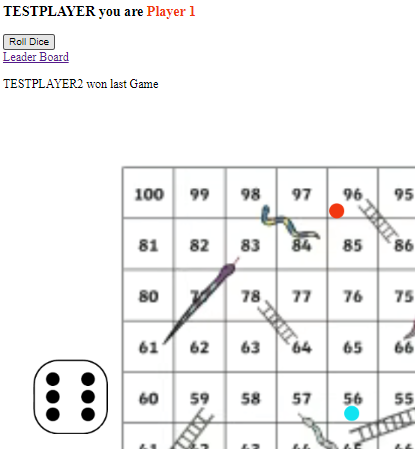
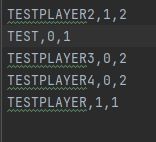
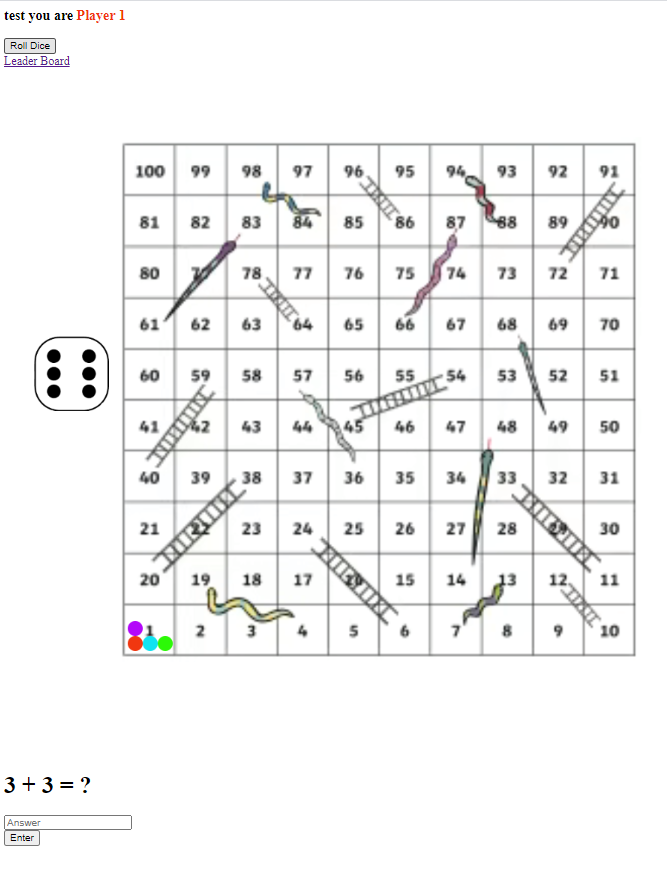
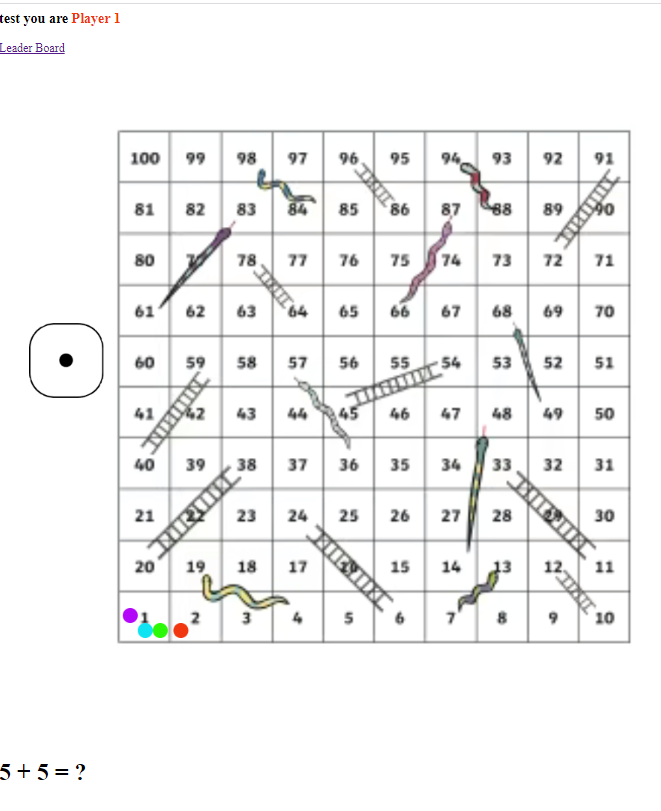
1. 
2. 
3.   
     
     
     
     
     
     
     
     
     
     
     
     
     
   ~
4. 
5. 
6. 
7.   
   

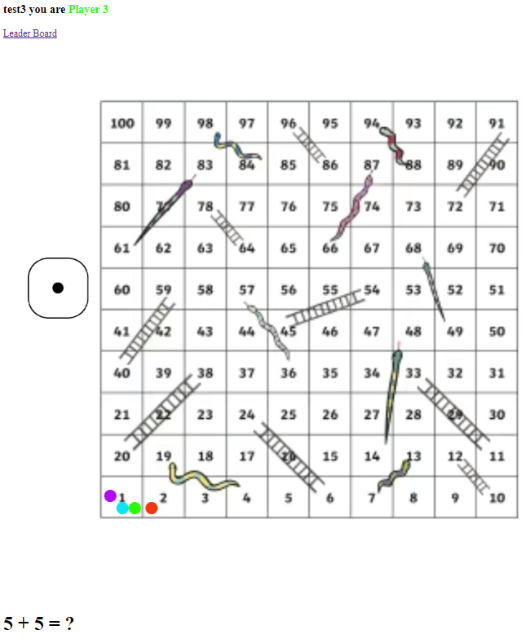


1. 





1. 
2. 



# Evaluation

My main objectives to make the game multiplayer and have an interactive questions GUI have been met, although at first I was thinking of using alerts and input fields for questions I found that just POSTing using an AJAX request was much better in terms of users ease of use, as otherwise the users would have to get rid of the alert every time someone answered a question while in my solution the text is just there and not inconvenient in any way.

I really liked the simplicity of making it real time as all I needed was a function being called twice every second that grabs the data from the server and draws what is needed.

The counter placement could have been done using a dictionary of cell to position but I decided to use logic behind it instead as I felt that I could more easily change the logic if a new board was used than remapping every cell to every position.