Object Oriented Programming

# CricBuzz

# Lab instructor: HANNAN FAROOQ

# Course instructor: ASIF AMEER

# Semester: FALL 2022

**Submitted By:**

# UMAIR AFZAL

# 21F-9612 BCS-3D

# 

# **National University of Computer and Emerging Sciences – FAST CFD**

# A picture containing text, gear, metalware Description automatically generated

Cric Buzz

# **Cricbuzz.h**

#pragma once

#include<iostream>

using namespace std;

struct IccRankings

{

int odi;

int t20;

int testMatches;

};

struct Date {

int day;

int month;

int year;

};

enum MatchStatus {

upcoming,

recent

};

enum AudienceStatus {

notAllowed,

allowed

};

class Team {

protected:

string teamName;

IccRankings rankings;

int noOfPlayers;

int totalWon;

int totalLost;

int odiWorldCupWon;

int t20WorldCupWon;

string teamCaptain;

string teamCoach;

string adminUsername;

string adminPassword;

public:

void displayMatches();

void updateCaptain();

void updateCoach();

void displayTeam();

};

class Player:public Team {

protected:

string playerName;

Date dateOfbirth;

int shirtNo;

int average;

IccRankings rankings;

int totalRuns;

int MatchesPlayed; //\*

int totalWickets;

string battingStyle;

string bowlingStyle;

string Role; //\*

public:

void addPlayer();

bool removePlayer();

bool searchPlayer();

bool updatePlayer();

};

class Match {

protected:

string team1;

string team2;

Date date;

string venue;

string matchType;

string tournamentName;

int noOfCommentators;

int noOfUmpires;

MatchStatus matchStatus;

AudienceStatus audienceStatus;

public:

void conductMatch();

void scheduleMatch();

void updateWorldRecords();

void updateTeamRanking();

void updatePlayerRanking(); //

void displayUpcomingMatches();

void displayRecentMatches();

};

class News : public Player, public Match {

public:

void newsOfprevMatch();

void newsOfUpcomMatch();

void teamRankings();

void playerRankings();

};

# **Cricbuzz.cpp**

#include <iostream>

#include <fstream>

#include <string>

#include "CricBuzz.h"

#include <Windows.h>

void Player::addPlayer()

{

cout << "Enter team Name of player: ";

cin >> teamName;

cout << "Enter name of player: ";

cin >> playerName;

cout << "Enter Date of birth in the format of (dd mm yyyy): ";

cin >> dateOfbirth.day >> dateOfbirth.month >> dateOfbirth.year;

cout << "Enter Shirt number: ";

cin >> shirtNo;

cout << "Enter Average: ";

cin >> average;

cout << "Enter Total runs: ";

cin >> totalRuns;

fstream addfile;

addfile.open("Added and Updated Players.txt", ios::app);

addfile << teamName << "\t" << playerName << "\t" << dateOfbirth.day << "-" << dateOfbirth.month << "-" << dateOfbirth.year << "\t" << shirtNo << "\t" << average << "\t" << totalRuns << endl;

}

bool Player::removePlayer()

{

string temp;

bool playerFound = false;

cout << "Enter first name of player: ";

cin >> playerName;

ifstream read;

read.open("players.txt");

while (!read.eof() && !playerFound) {

read >> temp;

if (playerName == temp) {

cout << "Player found!\n";

temp = "";

cout << "Data deleted\n";

playerFound = true;

}

}

if (!playerFound) {

cout << "Player not found\n";

}

return false;

}

bool Player::searchPlayer()

{

string temp;

bool playerFound = false;

cout << "Enter first name of player: ";

cin >> playerName;

ifstream read;

read.open("players.txt");

while (!read.eof() && !playerFound) {

read >> temp;

if (playerName == temp) {

cout << "Player found\n";

getline(read, temp);

cout << "last name\tDate of Birth\t\tShirt No\tAverage\t\tTotal Runs\n";

cout << temp << endl;

playerFound = true;

return true;

}

}

if (!playerFound) {

cout << "Player not found!\n";

return false;

}

}

bool Player::updatePlayer()

{

string temp;

bool playerFound = false;

cout << "Enter first name of player: ";

cin >> playerName;

ifstream read;

read.open("players.txt");

while (!read.eof() && !playerFound) {

read >> temp;

if (playerName == temp) {

cout << "Player found\n";

playerFound = true;

cout << "Enter data to update his previous data: ";

addPlayer();

return true;

}

}

if (!playerFound) {

cout << "Player not found!\n";

return false;

}

}

void Team::displayMatches()

{

ifstream read;

read.open("matches.txt");

string str;

while (!read.eof()) {

getline(read, str);

cout << str << endl;

}

}

void Team::updateCaptain()

{

string temp;

bool captFound = false;

cout << "Enter first name of captain: ";

cin >> teamCaptain;

ifstream read;

read.open("players.txt");

while (!read.eof() && !captFound) {

read >> temp;

if (teamCaptain == temp) {

cout << "Captain found\n";

captFound = true;

cout << "Enter the name to update: ";

cin >> teamCaptain;

cout << "Captain's name updated: ";

}

}

if (!captFound) {

cout << "Captain not found!\n";

}

}

void Team::updateCoach()

{

string temp;

bool coachFound = false;

cout << "Enter first name of coach: ";

cin >> teamCoach;

ifstream read;

read.open("players.txt");

while (!read.eof() && !coachFound) {

read >> temp;

if (teamCaptain == temp) {

cout << "Coach found\n";

coachFound = true;

cout << "Enter the name to update: ";

cin >> teamCoach;

cout << "Coach's name updated: ";

}

}

if (!coachFound) {

cout << "Coach not found!\n";

}

}

void Team::displayTeam()

{

string temp;

bool teamFound = false;

cout << "Enter name of team: ";

cin >> teamName;

ifstream read;

read.open("team.txt");

while (!read.eof() && !teamFound) {

read >> temp;

if (teamName == temp) {

getline(read,temp);

cout << "\tMatches Won\tMatches lost\tODI WC Won\tCaptain\t\t\tCoach\n";

cout << temp << endl;

teamFound = true;

}

}

read.close();

if (!teamFound) {

cout << "Team not found!\n";

}

}

void Match::conductMatch()

{

fstream file;

file.open("Matches.txt", ios::app);

//file<<

cout << "Enter Team 1: ";

getline(cin, team1);

cout << "Enter Team 2: ";

getline(cin, team2);

cout << "Enter Date (dd mm yyyy): ";

cin >> date.day >> date.month >> date.year;

cout << "Enter venue: ";

getline(cin, venue);

cout << "Enter match type(t20, ODI, test): ";

cin >> matchType;

cout << "Enter tournament name: ";

cin >> tournamentName;

cout << "Enter number of commentators: ";

cin >> noOfCommentators;

cout << "Enter number of Umpires: ";

cin >> noOfUmpires;

int st;

cout << "Enter match status: \nupcoming\tPress 1\nrecent\tPress 2\n:";

cin >> st;

switch (st)

{

case 1:

matchStatus = upcoming;

break;

case 2:

matchStatus = recent;

break;

default:

cout << "Invalid Entry!\n";

break;

}

cout << "Audience status: \nnotAllowed\tPress 1\nallowed\tPress 2\n: ";

cin >> st;

switch (st)

{

case 1:

audienceStatus = notAllowed;

break;

case 2:

audienceStatus = allowed;

break;

default:

cout << "Invalid Entry!\n";

break;

}

file << team1 << " vs " << team2 << "\t" << date.day << date.month << date.year << venue << matchType << tournamentName << noOfCommentators << noOfUmpires << matchStatus;

}

void Match::scheduleMatch()

{

ifstream read;

char chr;

read.open("matches.txt");

string str,temp="";

for (int i = 1; !read.eof() && temp != "Recent";i++) {

if(temp!="Recent"){

getline(read, str);

cout << temp <<" " <<str << endl ;

}

read >> temp;

}

cout << "Want to schedule a match?\tPress 'Y', else enter any character: ";

cin >> chr;

if (chr == 'y' || chr == 'Y') {

conductMatch();

}

}

void Match::updateWorldRecords()

{

ofstream write;

string recordTitle, name;

int odi, t20, test;

write.open("world records.txt", ios::app);

cout << "Enter record title: ";

getline(cin, recordTitle);

cout << "Enter record holder: ";

getline(cin, name);

write << recordTitle << ":\t" << name << endl;

write.close();

}

void Match::updateTeamRanking()

{

ofstream write;

string name;

int odi, t20, test;

write.open("Ranking.txt", ios::app);

cout << "Enter name of team: ";

getline(cin, name);

cout << "Enter ODI, T20 and Test rankings: ";

cin >> odi >> t20 >> test;

write << name << "\t|\t" << odi << "\t|\t" << t20 << "\t|\t" << test << endl;

write.close();

}

void Match::updatePlayerRanking()

{

ofstream write;

string name;

write.open("players Rankings.txt", ios::app);

cout << "Enter name of player: ";

getline(cin, name);

write << name << endl;

write.close();

}

void Match::displayUpcomingMatches()

{

ifstream read;

bool stop = false;

read.open("Upcoming.txt");

string str;

while (!read.eof() && !stop) {

getline(read, str);

cout << str << endl;

}

}

void Match::displayRecentMatches()

{

ifstream read;

bool stop = false;

read.open("Recent.txt");

string str;

while (!read.eof() && !stop) {

getline(read, str);

cout << str << endl;

}

}

void News::newsOfprevMatch()

{

cout << "Enter team 1: ";

cin >> team1;

cout << "Enter team 2: ";

cin >> team2;

string temp;

bool prevMatchFound = false, enter = false;

ifstream read;

read.open("matches.txt");

while (!read.eof() && !prevMatchFound) {

read >> temp;

if (temp == "Recent" || enter) {

enter = true;

if (team1 == temp) {

read >> temp;

if (team2 == temp) {

getline(read, temp);

cout << "That was a fantastic match! The information of match is: \n";

cout << "\tDate\t\tVenue\tMatch Type\tTournament Name\tCommentators\tUmpires\t\tAudience Status\n";

cout << temp << endl;

prevMatchFound = true;

}

}

}

}

if (!prevMatchFound) {

cout << "No news found about given match!\n";

}

}

void News::newsOfUpcomMatch()

{

cout << "Enter team 1: ";

cin >> team1;

cout << "Enter team 2: ";

cin >> team2;

string temp;

bool upMatch = false, enter = false;

ifstream read;

read.open("matches.txt");

while (!read.eof() && !upMatch && temp!="Recent") {

read >> temp;

if (team1 == temp) {

read >> temp;

if (team2 == temp) {

cout << "Yes, the wait is about to over, the news about most awaited match is: \n";

getline(read, temp);

cout << "\tDate\t\tVenue\tMatch Type\tTournament Name\tCommentators\tUmpires\t\tAudience Status\n";

cout << temp << endl;

upMatch = true;

}

}

}

if (!upMatch) {

cout << "No news found about given match!\n";

}

}

void News::teamRankings()

{

ifstream read;

bool stop = false;

read.open("Ranking.txt");

string str;

cout << "Here is the ICC rankings of a few teams!\n";

while (!read.eof() && !stop) {

getline(read, str);

cout << str << endl;

}

}

void News::playerRankings()

{

ifstream read;

bool stop = false;

read.open("player Rankings.txt");

string str;

cout << "Breaking News!!!!\nBabar Azam has crossed Virat Kohli and taken the 1st Place\n";

while (!read.eof() && !stop) {

getline(read, str);

cout << str << endl;

}

}

# **main.cpp**

#include <SFML/Graphics.hpp>

#include<SFML\Audio.hpp>

#include <iostream>

#include <fstream>

#include "CricBuzz.h"

#include <Windows.h>

#include <string>

using namespace std;

using namespace sf;

int main() {

bool invalid = false, enter = false, stop = false;

int choice;

char chr;

int c2, c3;

fstream writeFile;

ifstream readFile;

string email, password, temp;

bool logged = false;

Team teamObj;

Player playerObj;

Match matchObj;

News newsObj;

ifstream read;

ifstream readTeamRankings;

RenderWindow window(VideoMode(650, 900), "Cric Buzz");

Font font;

if (!font.loadFromFile("OpenSans-Bold.ttf"))

{

cout << "file not found!\n";

}

Music sound;

if (!sound.openFromFile("Y2Mate.ogg"))

{

cout << "file not found!\n";

}

RectangleShape sound\_Button(Vector2f(650.0f, 900.0f));

sound\_Button.setPosition(0.0, 0.0);

Texture sound\_ButtonTexture;

sound\_ButtonTexture.loadFromFile("Login.png");

sound\_Button.setTexture(&sound\_ButtonTexture);

RectangleShape starter(Vector2f(650.0f, 900.0f));

starter.setPosition(0.0, 0.0);

Texture starter\_texture;

starter\_texture.loadFromFile("starter.png");

starter.setTexture(&starter\_texture);

RectangleShape pic3(Vector2f(650.0f, 900.0f));

pic3.setPosition(0.0, 0.0);

Texture pic3\_texture;

pic3\_texture.loadFromFile("Signup.png");

pic3.setTexture(&pic3\_texture);

RectangleShape classMenu(Vector2f(650.0f, 900.0f));

classMenu.setPosition(0.0, 0.0);

Texture classMenu\_texture;

classMenu\_texture.loadFromFile("Class Menu.png");

classMenu.setTexture(&classMenu\_texture);

RectangleShape teamMenu(Vector2f(650.0f, 900.0f));

teamMenu.setPosition(0.0, 0.0);

Texture teamMenu\_texture;

teamMenu\_texture.loadFromFile("team menu2.png");

teamMenu.setTexture(&teamMenu\_texture);

RectangleShape teamPk(Vector2f(650.0f, 900.0f));

teamPk.setPosition(0.0, 0.0);

Texture teamPk\_texture;

teamPk\_texture.loadFromFile("Pakistan.png");

teamPk.setTexture(&teamPk\_texture);

RectangleShape teamInd(Vector2f(650.0f, 900.0f));

teamInd.setPosition(0.0, 0.0);

Texture teamInd\_texture;

teamInd\_texture.loadFromFile("India.png");

teamInd.setTexture(&teamInd\_texture);

RectangleShape teamEng(Vector2f(650.0f, 900.0f));

teamEng.setPosition(0.0, 0.0);

Texture teamEng\_texture;

teamEng\_texture.loadFromFile("England.png");

teamEng.setTexture(&teamEng\_texture);

RectangleShape teamNz(Vector2f(650.0f, 900.0f));

teamNz.setPosition(0.0, 0.0);

Texture teamNz\_texture;

teamNz\_texture.loadFromFile("New Zealand.png");

teamNz.setTexture(&teamNz\_texture);

RectangleShape teamBan(Vector2f(650.0f, 900.0f));

teamBan.setPosition(0.0, 0.0);

Texture teamBan\_texture;

teamBan\_texture.loadFromFile("Bangladesh.png");

teamBan.setTexture(&teamBan\_texture);

RectangleShape teamRankings(Vector2f(650.0f, 900.0f));

teamRankings.setPosition(0.0, 0.0);

Texture teamRankings\_texture;

teamRankings\_texture.loadFromFile("team rankings.png");

teamRankings.setTexture(&teamRankings\_texture);

RectangleShape recNews(Vector2f(650.0f, 900.0f));

recNews.setPosition(0.0, 0.0);

Texture recNews\_texture;

recNews\_texture.loadFromFile("recNews.png");

recNews.setTexture(&recNews\_texture);

RectangleShape UpNews(Vector2f(650.0f, 900.0f));

UpNews.setPosition(0.0, 0.0);

Texture UpNews\_texture;

UpNews\_texture.loadFromFile("UpNews.png");

UpNews.setTexture(&UpNews\_texture);

RectangleShape playerNews(Vector2f(650.0f, 900.0f));

playerNews.setPosition(0.0, 0.0);

Texture playerNews\_texture;

playerNews\_texture.loadFromFile("playerNews.png");

playerNews.setTexture(&playerNews\_texture);

RectangleShape RankingsNews(Vector2f(650.0f, 900.0f));

RankingsNews.setPosition(0.0, 0.0);

Texture RankingsNews\_texture;

RankingsNews\_texture.loadFromFile("RankingsNews.png");

RankingsNews.setTexture(&RankingsNews\_texture);

RectangleShape f2Menu(Vector2f(650.0f, 900.0f));

f2Menu.setPosition(0.0, 0.0);

Texture f2Menu\_texture;

f2Menu\_texture.loadFromFile("F2 menu.png");

f2Menu.setTexture(&f2Menu\_texture);

RectangleShape newsMenu(Vector2f(650.0f, 900.0f));

newsMenu.setPosition(0.0, 0.0);

Texture newsMenu\_texture;

newsMenu\_texture.loadFromFile("News Menu.png");

newsMenu.setTexture(&newsMenu\_texture);

RectangleShape liveMatch(Vector2f(650.0f, 900.0f));

liveMatch.setPosition(0.0, 0.0);

Texture liveMatch\_texture;

liveMatch\_texture.loadFromFile("Live Match.png");

liveMatch.setTexture(&liveMatch\_texture);

printf("\033[1;32m");

cout << "\*\_\*\_\*\_\*\_\*\_\*\_\*\_\*\_\*\_\*\_\*\_\*\_\*\_\* C R I C\tB U Z Z \*\_\*\_\*\_\*\_\*\_\*\_\*\_\*\_\*\_\*\_\*\_\*\_\*\_\*\n"; //

cout << "Login\tPress 1\nSignUp\tPress 2\n:";

cin >> choice;

cout << endl;

printf("\033[0m");

try

{

switch (choice)

{

case 1:

printf("\033[1;31m");

cout << "Enter your email: ";

cin >> email;

cout << "Enter password: ";

cin >> password;

printf("\033[1;33m");

loading();

readFile.open("Emails and Passwords.txt");

while (!readFile.eof() && !logged) {

readFile >> temp;

if (email == temp) {

readFile >> temp;

if (password == temp) {

cout << "Logged in!\n";

logged = true;

enter = true;

}

}

}

if (!logged)

cout << "Incorrect email or password\n";

break;

case 2:

writeFile.open("Emails and Passwords.txt", ios::app);

cout << "Enter your email: ";

cin >> email;

cout << "Enter your password: ";

cin >> password;

writeFile << email << endl << password << endl;

cout << "Your account has been signed up!\n";

enter = true;

writeFile.close();

break;

default:

cout << "Invalid Entry!\n";

invalid = true;

throw 'a';

break;

}//

if (enter) {

while (enter)

{

printf("\033[1;34m");

cout << "On which class do you want to work?\nTeam\tPress 1\nPlayer\tPress 2\nMatches\tPress 3\nNews\tPress 4\n: ";

cin >> choice;

switch (choice)

{

case 1:

printf("\033[1;35m");

cout << "Update Captain\tPress 1\nUpdate Coach\tPress 2\nDisplay Matches\tPress 3\nDisplay Team\tPress 4\n: ";

cin >> c2;

switch (c2)

{

case 1:

teamObj.updateCaptain();

break;

case 2:

teamObj.updateCoach();

break;

case 3:

teamObj.displayMatches();

break;

case 4:

teamObj.displayTeam();

break;

default:

cout << "Invalid Entry\n";

throw 'a';

break;

}

break;

case 2:

printf("\033[1;36m");

cout << "Add Player\tPress 1\nRemove Player\tPress 2\nUpdate Player\tPress 3\n: ";

cin >> c2;

switch (c2)

{

case 1:

playerObj.addPlayer();

break;

case 2:

playerObj.removePlayer();

break;

case 3:

playerObj.updatePlayer();

break;

default:

cout << "Invalid Entry\n";

throw 'a';

break;

}

break;

case 3:

printf("\033[1;37m");

cout << "Conduct Matches\tPress 1\nSchedule Matches\tPress 2\nUpdate WorldRecord\tPress 3\nUpdate TeamRankings\tPress 4\nUpdate PlayerRankings\tPress 5\nDisplay UpcomingMatches\tPress 6\nDisplay Recentmatches\tPress 7\n: ";

cin >> c2;

switch (c2)

{

case 1:

matchObj.conductMatch();

break;

case 2:

matchObj.scheduleMatch();

break;

case 3:

matchObj.updateWorldRecords();

break;

case 4:

matchObj.updateTeamRanking();

break;

case 5:

matchObj.updatePlayerRanking();

break;

case 6:

matchObj.displayUpcomingMatches();

break;

case 7:

matchObj.displayRecentMatches();

break;

default:

cout << "Invalid Entry\n";

throw 'a';

break;

}

break;

case 4:

printf("\033[1;31m");

cout << "What kind of news do you want to see?\nNews about Previous Match\tPress 1\nNews about Upcoming Match\tPress 2\nNews about Team Rankings\tPress 3\nNews about Players Rankings\tPress 4\n: ";

cin >> c2;

switch (c2)

{

case 1:

newsObj.newsOfprevMatch();

break;

case 2:

newsObj.newsOfUpcomMatch();

break;

case 3:

newsObj.teamRankings();

break;

case 4:

newsObj.playerRankings();

break;

default:

throw 'a';

cout << "Invalid Entry\n";

break;

}

break;

default:

cout << "Invalid Entry\n";

throw 'a';

break;

}

printf("\033[1;32m");

cout << "Press 's' to quit the program, else enter any character!\n";

cin >> chr;

system("cls");

if (chr == 's') {

enter = false;

}

}

}

}

catch (char chr) {

cout << "Exception can also be handled if you enter character or string!\n";

}

if (!invalid) {

printf("\033[1;33m");

cout << "Time to view this app Graphically!\nPlease Enter Right Shift to start the app: ";

Sleep(1000);

read.open("teams and players.txt");

string str[8][15];

for (int i = 0; !read.eof() && i < 8; i++) {

for (int j = 0; j < 15; j++) {

getline(read, str[i][j]);

}

}

read.close();

Text players[8][15];

int y;

for (int i = 0; i < 8; i++) {

y = 150;

for (int j = 0; j < 15; j++) {

players[i][j].setFont(font);

players[i][j].setCharacterSize(18);

players[i][j].setFillColor(Color::White);

players[i][j].setOutlineThickness(1);

players[i][j].setOutlineColor(Color::Black);

players[i][j].setPosition(2, y);

players[i][j].setString(str[i][j]);

y = y + 40;

}

}

readTeamRankings.open("Ranking.txt");

string str2[30];

int index = 0;

while (!readTeamRankings.eof())

{

getline(readTeamRankings, str2[index]);

index++;

}

Text teamRank[30];

y = 180;

for (int i = 0; i < 30; i++) {

teamRank[i].setFont(font);

teamRank[i].setCharacterSize(25);

teamRank[i].setFillColor(Color::White);

teamRank[i].setOutlineThickness(2);

teamRank[i].setOutlineColor(Color::Black);

teamRank[i].setPosition(80, y);

teamRank[i].setString(str2[i]);

y = y + 40;

}

readTeamRankings.close();

ifstream readWR;

readWR.open("world records.txt");

string str3[30];

index = 0;

while (!readWR.eof())

{

getline(readWR, str3[index]);

index++;

}

Text worldRec[30];

y = 180;

for (int i = 0; i < 30; i++) {

worldRec[i].setFont(font);

worldRec[i].setCharacterSize(25);

worldRec[i].setFillColor(Color::White);

worldRec[i].setOutlineThickness(2);

worldRec[i].setOutlineColor(Color::Black);

worldRec[i].setPosition(80, y);

worldRec[i].setString(str3[i]);

y = y + 40;

}

readWR.close();

ifstream readplayerRec;

readplayerRec.open("player Rankings.txt");

string str4[30];

index = 0;

while (!readplayerRec.eof())

{

getline(readplayerRec, str4[index]);

index++;

}

Text playerRec[30];

y = 180;

for (int i = 0; i < 30; i++) {

playerRec[i].setFont(font);

playerRec[i].setCharacterSize(25);

playerRec[i].setFillColor(Color::White);

playerRec[i].setOutlineThickness(1);

playerRec[i].setOutlineColor(Color::Black);

playerRec[i].setPosition(80, y);

playerRec[i].setString(str4[i]);

y = y + 40;

}

readplayerRec.close();

ifstream readUpcom;

readUpcom.open("upcom.txt");

string str5[30];

index = 0;

while (!readUpcom.eof())

{

getline(readUpcom, str5[index]);

index++;

}

Text upcom[30];

y = 180;

for (int i = 0; i < 30; i++) {

upcom[i].setFont(font);

upcom[i].setCharacterSize(25);

upcom[i].setFillColor(Color::White);

upcom[i].setOutlineThickness(1);

upcom[i].setOutlineColor(Color::Black);

upcom[i].setPosition(15, y);

upcom[i].setString(str5[i]);

y = y + 40;

}

readUpcom.close();

ifstream readRecent;

readRecent.open("rec.txt");

string str6[30];

index = 0;

while (!readRecent.eof())

{

getline(readRecent, str6[index]);

index++;

}

Text rec[30];

y = 180;

for (int i = 0; i < 30; i++) {

rec[i].setFont(font);

rec[i].setCharacterSize(25);

rec[i].setFillColor(Color::White);

rec[i].setOutlineThickness(1);

rec[i].setOutlineColor(Color::Black);

rec[i].setPosition(15, y);

rec[i].setString(str6[i]);

y = y + 40;

}

readRecent.close();

Text startUp;

startUp.setFont(font);

startUp.setCharacterSize(25);

startUp.setFillColor(Color::White);

startUp.setOutlineThickness(2);

startUp.setOutlineColor(Color::Black);

startUp.setPosition(80, 80);

startUp.setString("Press Right Shift!");

while (window.isOpen())

{

if (Keyboard::isKeyPressed(Keyboard::RShift)) {

window.draw(starter);

}

if (Keyboard::isKeyPressed(Keyboard::Enter)) {

window.draw(sound\_Button);

}

if (Keyboard::isKeyPressed(Keyboard::Num2)) {

window.draw(pic3);

}

if (Keyboard::isKeyPressed(Keyboard::Num3)) {

window.draw(classMenu);

}

if (Keyboard::isKeyPressed(Keyboard::Num1)) {

window.draw(classMenu);

}

if (Keyboard::isKeyPressed(Keyboard::F1))

{

window.draw(teamMenu);

}

if (Keyboard::isKeyPressed(Keyboard::M))

{

window.draw(liveMatch);

}

if (Keyboard::isKeyPressed(Keyboard::F2))

{

window.draw(f2Menu);

}

if (Keyboard::isKeyPressed(Keyboard::F3))

{

window.draw(newsMenu);

}

if (Keyboard::isKeyPressed(Keyboard::P)) {

window.draw(teamPk);

for (int i = 0, j = 1; j < 15; j++) {

window.draw(players[i][j]);

}

}

if (Keyboard::isKeyPressed(Keyboard::I)) {

window.draw(teamInd);

for (int i = 2, j = 1; j < 15; j++) {

window.draw(players[i][j]);

}

}

if (Keyboard::isKeyPressed(Keyboard::E)) {

window.draw(teamEng);

for (int i = 1, j = 1; j < 15; j++) {

window.draw(players[i][j]);

}

}

if (Keyboard::isKeyPressed(Keyboard::N)) {

window.draw(teamNz);

for (int i = 5, j = 1; j < 15; j++) {

window.draw(players[i][j]);

}

}

if (Keyboard::isKeyPressed(Keyboard::B)) {

window.draw(teamBan);

for (int i = 4, j = 1; j < 15; j++) {

window.draw(players[i][j]);

}

}

if (Keyboard::isKeyPressed(Keyboard::R)) {

window.draw(teamRankings);

for (int i = 0; i < 30; i++) {

window.draw(teamRank[i]);

}

}

if (Keyboard::isKeyPressed(Keyboard::Z)) {

window.draw(RankingsNews);

for (int i = 0; i < 30; i++) {

window.draw(teamRank[i]);

}

}

if (Keyboard::isKeyPressed(Keyboard::W)) {

window.draw(teamRankings);

for (int i = 0; i < 30; i++) {

window.draw(worldRec[i]);

}

}

if (Keyboard::isKeyPressed(Keyboard::X)) {

window.draw(playerNews);

for (int i = 0; i < 30; i++) {

window.draw(playerRec[i]);

}

}

if (Keyboard::isKeyPressed(Keyboard::L)) {

window.draw(teamRankings);

for (int i = 0; i < 30; i++) {

window.draw(playerRec[i]);

}

}

if (Keyboard::isKeyPressed(Keyboard::C)) {

window.draw(UpNews);

for (int i = 0; i < 30; i++) {

window.draw(upcom[i]);

}

}

if (Keyboard::isKeyPressed(Keyboard::Q)) {

window.draw(teamRankings);

for (int i = 0; i < 30; i++) {

window.draw(upcom[i]);

}

}

if (Keyboard::isKeyPressed(Keyboard::O)) {

window.draw(teamRankings);

for (int i = 0; i < 30; i++) {

window.draw(rec[i]);

}

}

if (Keyboard::isKeyPressed(Keyboard::V)) {

window.draw(recNews);

for (int i = 0; i < 30; i++) {

window.draw(rec[i]);

}

}

if (Keyboard::isKeyPressed(Keyboard::Space)) {

sound.play();

}

if (Keyboard::isKeyPressed(Keyboard::Num0)) {

sound.pause();

}

Event event;

while (window.pollEvent(event))

{

if (event.type == Event::Closed)

window.close();

}

window.display();

}

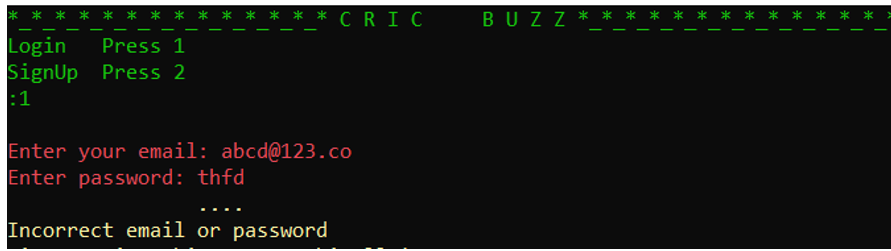
}

return 0;

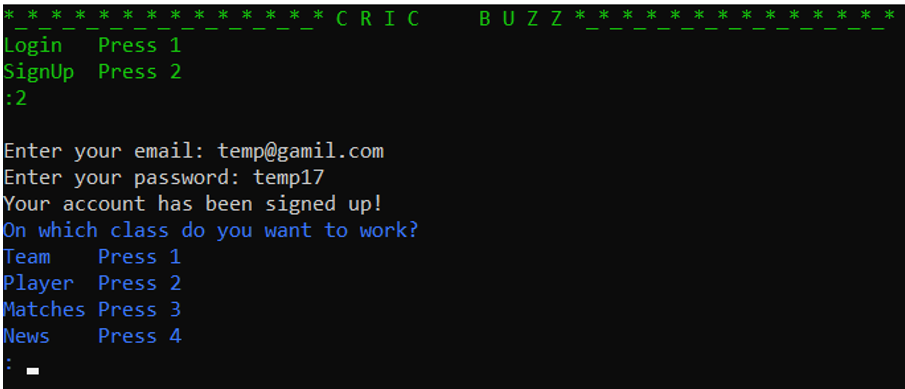
}

# **Outputs:**

## Login

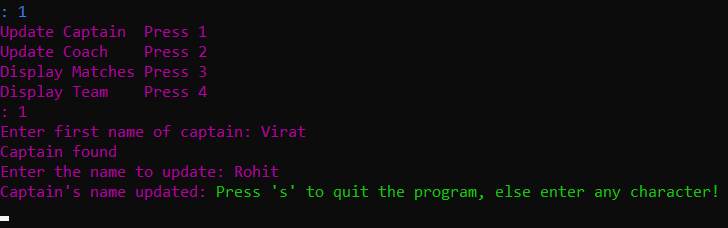


## Sign Up

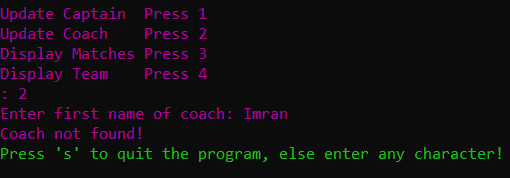


## **Team or Case 1**

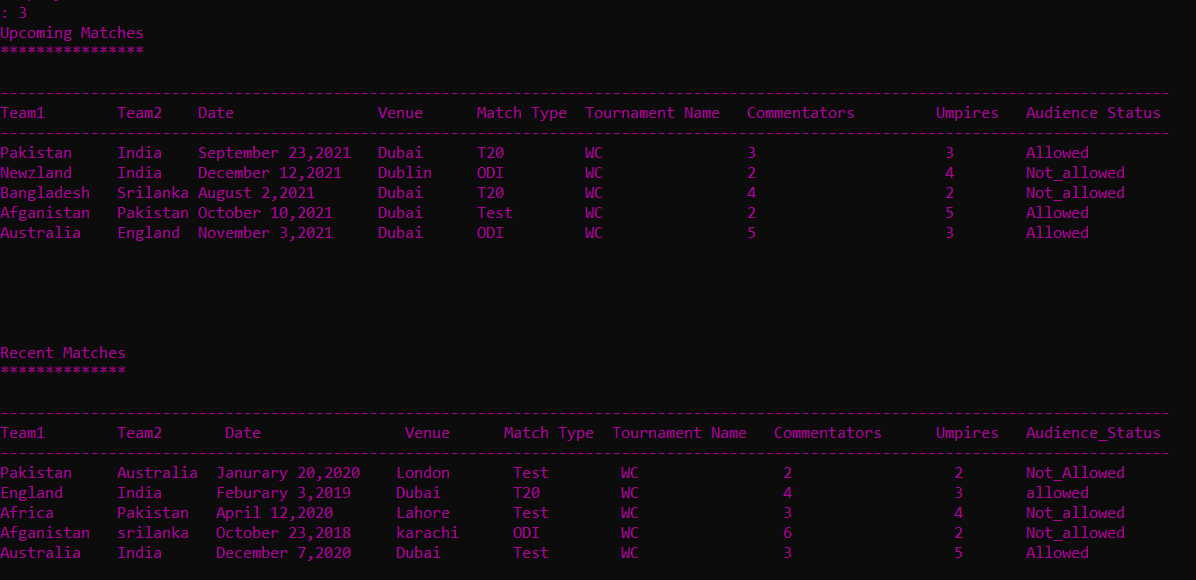
## Case 1



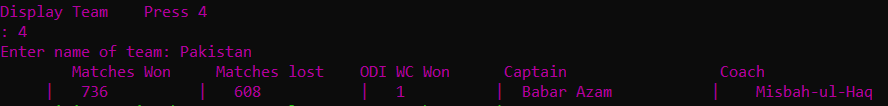
## Case 2



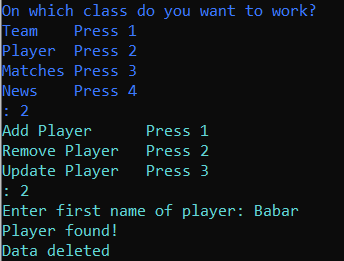
## Case 3



## Case 4

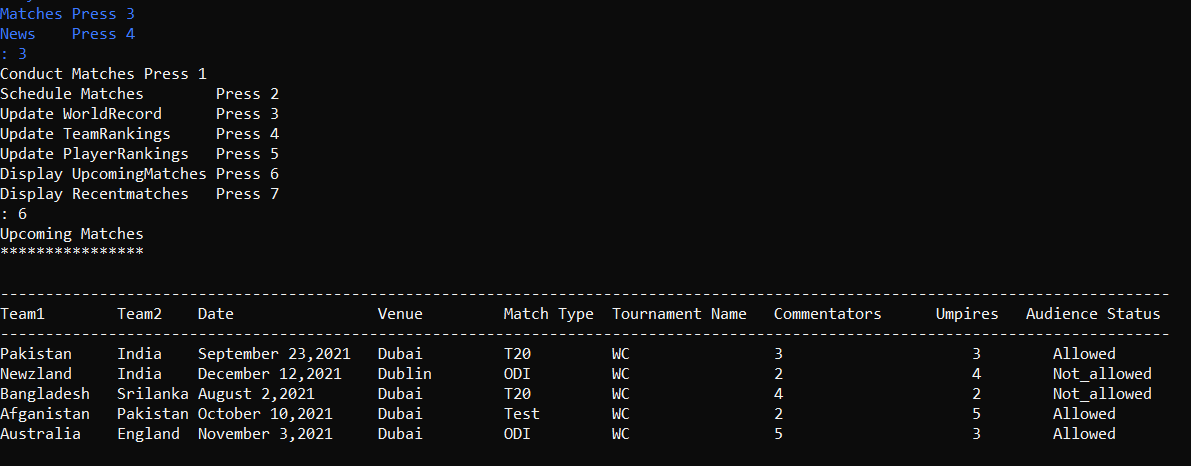


## **Player or Case 2**

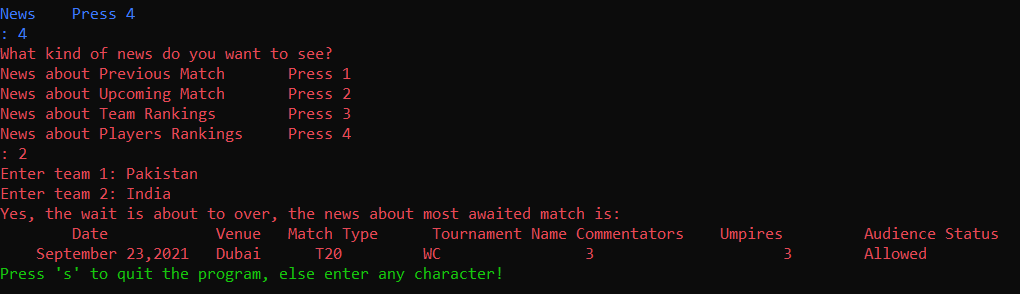


Every function works, you can check by your own.

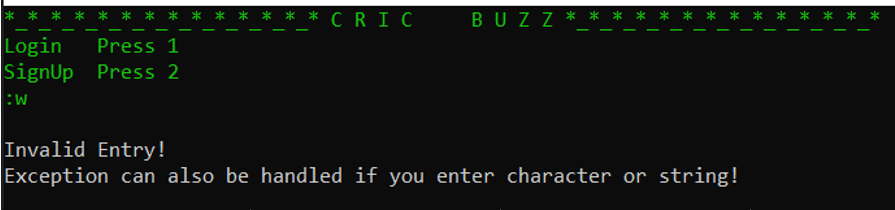
## **Matches or Case 3**



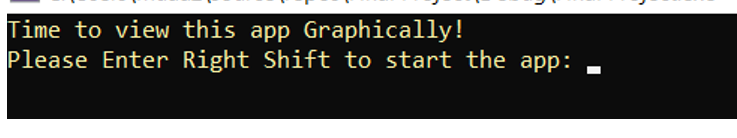
## **Matches or Case 4**

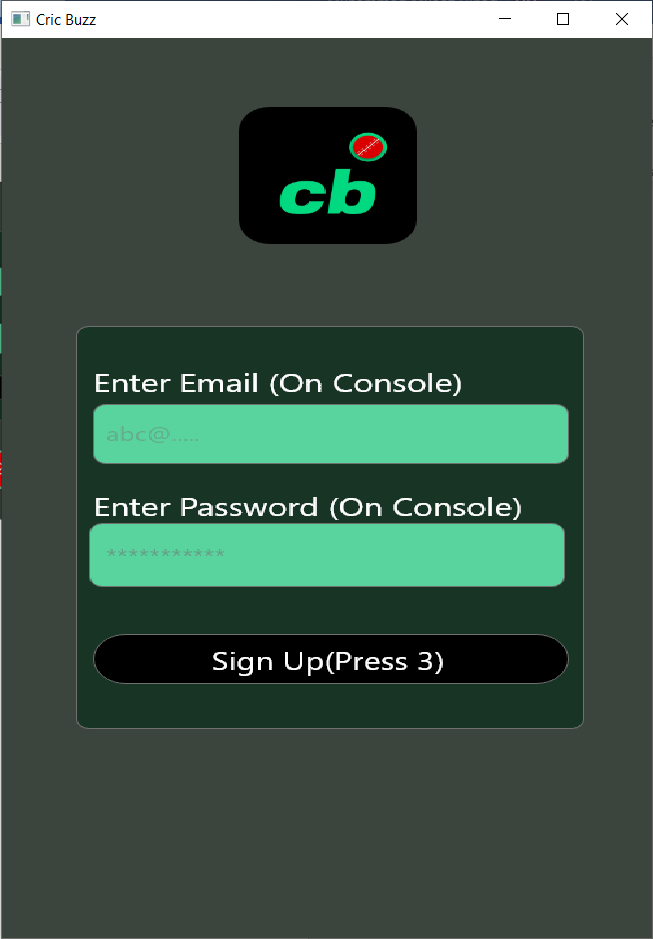
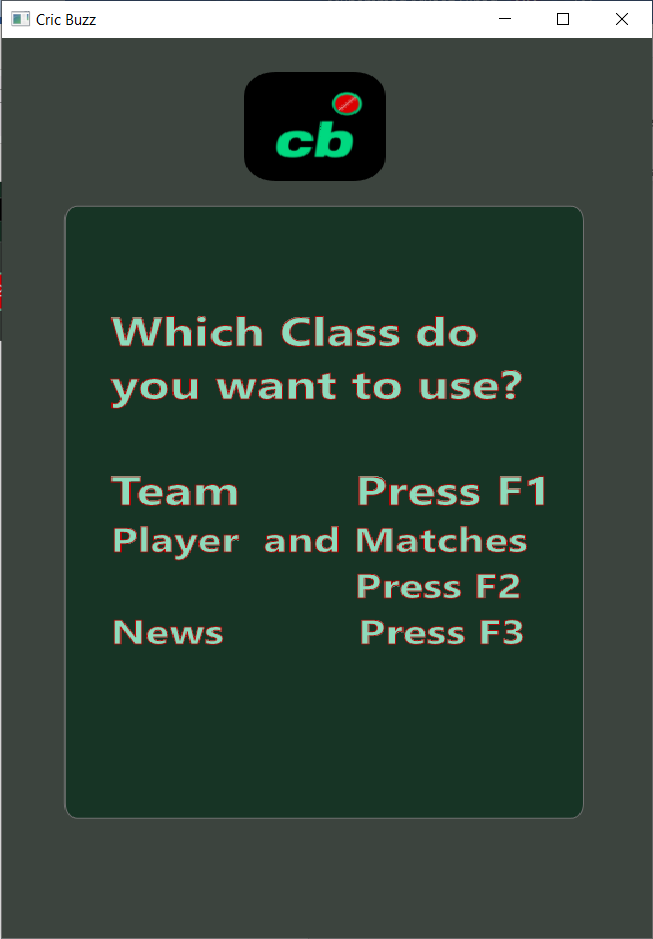
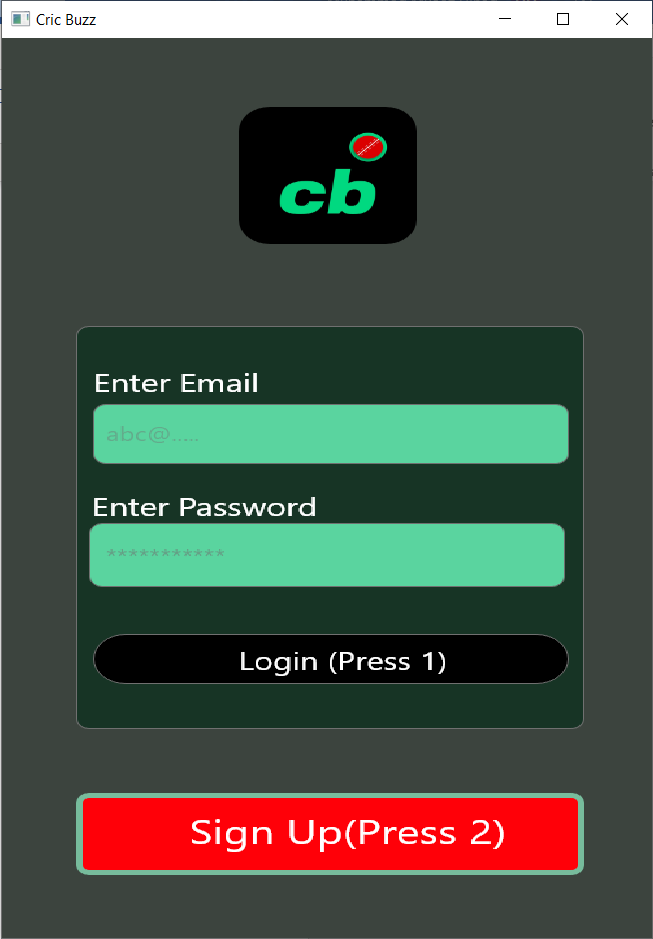


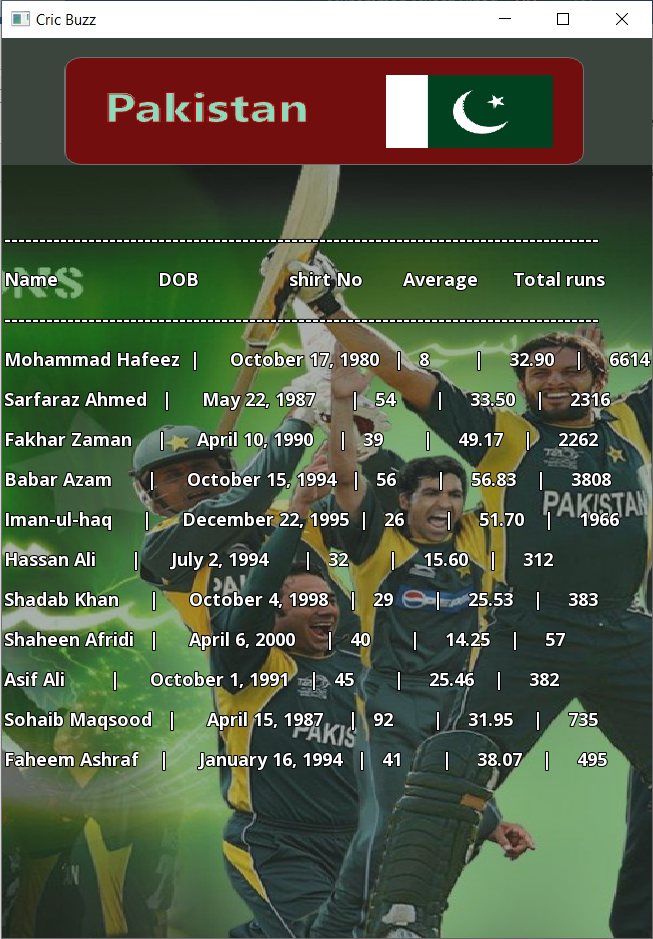
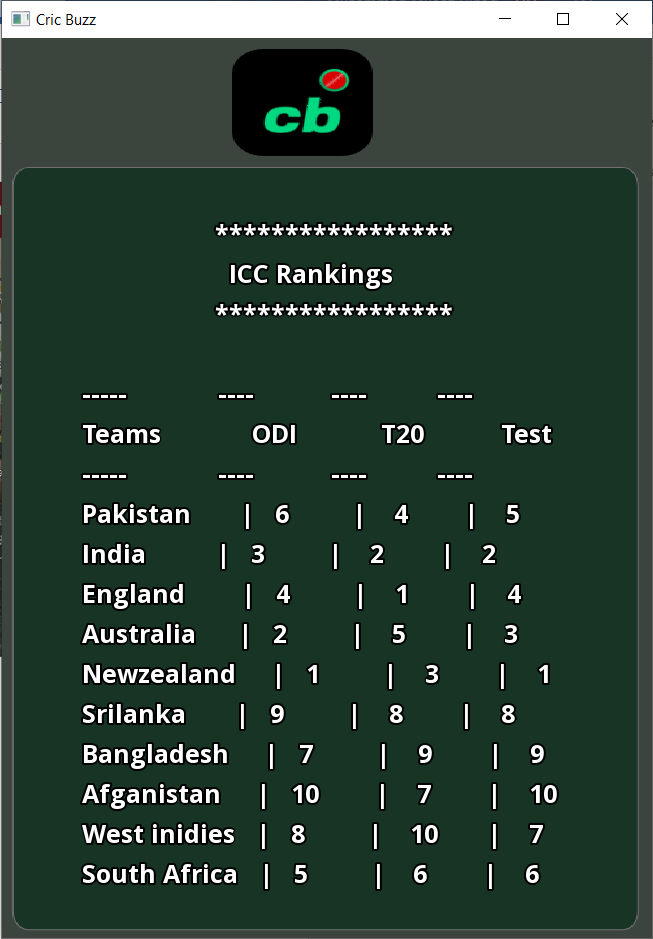
## Exception Handling Case

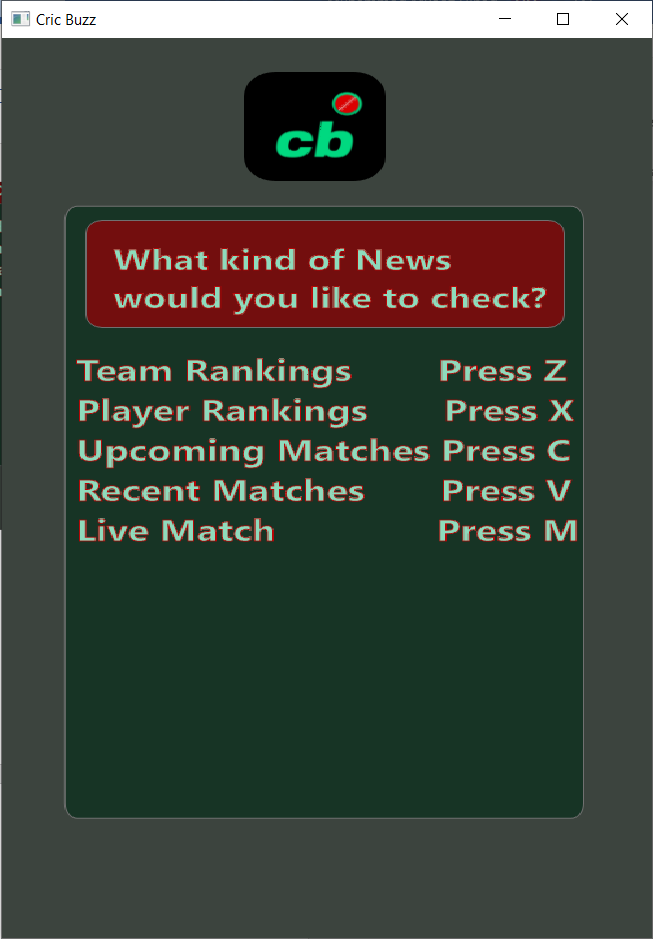
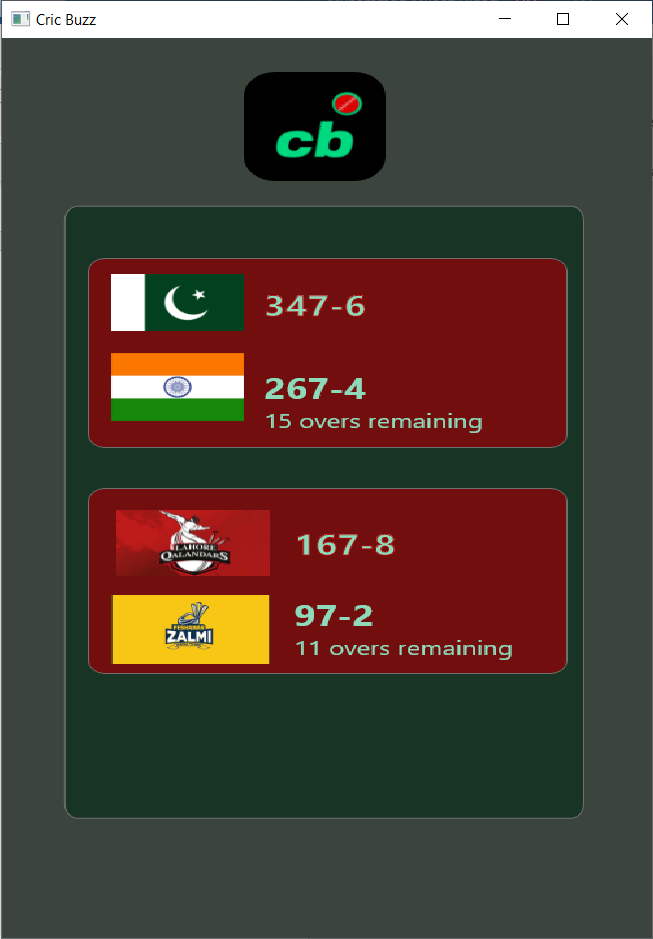


# **SFML Outputs**









You can run the code and test the sfml by yourself for better experience.