Table of Contents

DECLARATION

We hereby declare that the project entitled "Cricscore", submitted as Minor Project for 6th semester in Computer Science & Engineering at ROORKEE COLLEGE OF ENGINEERING, ROORKEE, UTTARAKHAND is an authentic record of our genuine work under the guidance of Ms. Mrinalinee Singh, Department of Computer Science & Engineering, ROORKEE COLLEGE OF ENGINEERING, ROORKEE.

Date:

Place: Roorkee

Signed:

Deepak Kumar

(211020101016)

Abhishek Dixit

(211020101003)

CERTIFICATE

This is to certify that the minor project report entitled "Cricscore", submitted by Deepak Kumar and Abhishek Dixit, has been carried out under the guidance of Ms. Mrinalinee Singh, Department of Computer Science & Engineering, ROORKEE COLLEGE OF ENGINEERING, ROORKEE

The project report fulfills the submissionrequirements for Web Engineering Project in 6th semester of Computer Science & Engineering from ROORKEE COLLEGE OF ENGINEERING, ROORKEE, Uttarakhand.

Internal Examiner:

Date:

ACKNOWLEDGEMENT

We express our sincere indebtedness towards our Guide, Prof. Ms. Mrinalinee Singh, Computer Science & Engineering, ROORKEE COLLEGE OF ENGINEERING, ROORKEE for her invaluable guidance, suggestions and supervision throughout the work. Without kind patronage and guidance, the project would not have taken shape. We also extend our heartfelt gratitude and sincere regards for her kind approval of the project, timely counselling, and advice.

We would also like to thank to our HOD Sir Mr. Pranav Hari Department of Computer Science & Engineering, ROORKEE COLLEGE OF ENGINEERING, ROORKEE for her expert advice and counselling from time to time.

Abhishek Dixit
(211020101003)

Deepak Kumar
(211020101016)

Introduction

Background and Motivation for the Project:

Cricket, a globally cherished sport, captivates millions of fans who eagerly track live scores. Traditional methods involve TV or online platforms, yet there's a rising demand for personalized, accessible solutions.

This project aims to develop a user-friendly application for accessing the live cricket scores. Utilizing python and web scraping techniques, we seek to create simple yet effective tool that fetches scores from popular cricket websites.

Objectives and Goals of the Project:

The primary objective of this project is to develop a cricket score application using Python that fetches live scores from cricket websites in real-time. The key goals of the project include:

1. Implementing web scraping techniques to extract live cricket scores from websites such as **Cricbuzz.**
2. Developing a user-friendly graphical user interface (GUI) using Tkinter library to display the live scores.
3. Providing users with the ability to select from a list of live matches and view detailed scorecards and match summaries.
4. Ensuring the application is robust, reliable, and easy to use for cricket enthusiasts of all ages and technical backgrounds.

Overview of the Technologies Used:

The project utilizes the following technologies:

1. **Python:** Python is a versatile and widely used programming language known for its simplicity and readability. It provides rich libraries and frameworks for various tasks, making it an ideal choice for web scraping and GUI development.

2. **Tkinter:** Tkinter is the standard GUI toolkit for Python. It provides a set of tools for building graphical user interfaces in Python applications. Tkinter is easy to use, cross-platform, and integrates seamlessly with Python, making it suitable for developing the GUI for our cricket score application.

3. **BeautifulSoup:** BeautifulSoup is a Python library for web scraping. It allows us to parse HTML and XML documents, extract useful information, and navigate through the document structure. BeautifulSoup simplifies the process of extracting live cricket scores from websites, enabling us to retrieve and display them in our application.

These technologies complement each other to create a robust and functional cricket score application that meets the project objectives and goals.

Project Overview

Description of the Project Functionality:

The Cricscore project aims to develop a user-friendly application that provides live updates of cricket matches to users. The application fetches live cricket scores from popular cricket websites using web scraping techniques and presents them in a graphical user interface(GUI) built using Tkinter library in Python.

The functionality of the application includes:

- Scraping live cricket scores from websites such as Cricbuzz in real-time.
- Displaying a list of live matches for users to select from.
- Presenting detailed scorecards and match summaries for the selected match.
- Updating scores automatically as the match progresses.
- Providing a visually appealing and intuitive user interface for easy navigation and interaction.

Features and Capabilities of the Cricket Score Application:

The cricket score application offers several features and capabilities to enhance the user experience and provide comprehensive coverage of live cricket matches. **These include:**

1. **Live Score Updates:** The application fetches live scores from cricket websites and updates them in real-time, ensuring users stay informed about the latest match developments.
2. **Match Selection:** Users can select from a list of live matches to view detailed scorecards and match summaries for their preferred match.
3. **Detailed Scorecards**: The application provides detailed scorecards for each match, including information such as runs scored, wickets taken, overs bowled, and current status of the match.

4. **Match Summaries:** Users can access match summaries that provide a brief overview of the match, including key highlights and important events.
5. **User-Friendly Interface:** The GUI built using Tkinter library offers a user-friendly interface with intuitive navigation and visually appealing design, making it easy for users to access and navigate through the application.
6. **Automatic Updates:** The application automatically updates scores as the match progresses, ensuring users have access to the latest information without manual intervention.
7. **Cross-Platform Compatibility:** The application is developed using Python, making it compatible with multiple platforms including Windows, Linux, and macOS, thus catering to a wide range of users.

Target Audience and Potential Users:

The target audience for the cricket score application includes cricket enthusiasts, fans, and followers who are interested in staying updated with live cricket scores. Potential users of the application are:

- Cricket fans who want to follow live matches and stay informed about the latest scores and match developments.
- Sports enthusiasts who enjoy watching cricket matches and want quick access to live scores on their devices.
- Gamers and fantasy cricket players who need real-time updates to make informed decisions while playing fantasy cricket leagues.

The application caters to users of all ages and technical backgrounds, providing a convenient and accessible platform to follow live cricket matches and stay connected with the world of cricket.

Requirements Analysis

Detailed Analysis of Functional and Non-Functional Requirements:

1. Functional Requirements:

- The application should fetch live cricket scores from popular cricket websites such as Cricbuzz. • It should display a list of live matches for users to select from.
- Users should be able to view detailed scorecards and match summaries for selected matches. • The application should update scores automatically as the match progresses.
- It should provide a user-friendly interface for easy navigation and interaction.
- Users should be able to customize settings such as match refresh interval and display preferences.

2. Non-Functional Requirements:

- **Performance:** The application should have fast response times and handle multiple concurrent users.
- **Reliability:** The application should be reliable and available at all times to provide real-time updates.
- **Usability:** The user interface should be intuitive and easy to use, catering to users of all technical backgrounds.
- **Scalability:** The application should be scalable to accommodate future enhancements and increased user load.
- **Security:** The application should ensure data security and protect user privacy.

Use Cases and User Stories:

1. **Use Case 1: Fetch Live Cricket Scores**
   - Actor: Application

- Description: The application fetches live cricket scores from cricket websites using web scraping techniques.
- Preconditions: Internet connectivity is available.
- Basic Flow:
    a. The application initiates a request to the cricket website.
    b. It scrapes the live scores from the website.
    c. It updates the user interface with the live scores.

2. **Use Case 2: Display List of Live Matches**
    - Actor: Application
    - Description: The application displays a list of live matches for users to select from.
    - Preconditions: Live cricket matches are available.
    - Basic Flow:
        a. The application retrieves the list of live matches.
        b. It displays the list in the user interface.
        c. Users select a match from the list.
3. **User Story:** As a cricket fan, I want to view detailed scorecards for live matches, so that I can stay updated with the latest match statistics and scores.
    Acceptance Criteria:
    - The application should display detailed scorecards including runs scored, wickets taken, overs bowled, and current match status.
    - Users should be able to access the scorecards for selected matches with one-click navigation.

Functional Requirements Specification:

1. **Fetch Live Cricket Scores:**

- The application shall fetch live cricket scores from cricket websites using web scraping techniques.
- It shall update the scores automatically as the match progresses.

2. **Display List of Live Matches:**

   - The application shall display a list of live matches for users to select from.
   - It shall present the list in a user-friendly format with match details such as teams, venue, and time.

3. **View Detailed Scorecards and Match Summaries:**

   - The application shall provide detailed scorecards for selected matches, including runs scored, wickets taken, overs bowled, and current match status.
   - It shall display match summaries with key highlights and events for better understanding.

By analyzing these functional and non-functional requirements, along with use cases and user stories, the project team can ensure that the cricket score application meets the needs and expectations of its users.

Design

High-level Architecture of the Application:

The cricket score application follows a client-server architecture, where the client is the user interface built using Tkinter, and the server is responsible for fetching live cricket scores from cricket websites using web scraping techniques. The high-level architecture consists of the following components:

1. **User Interface (Client):** Built using Tkinter library in Python, the user interface provides a graphical interface for users to interact with the application. It displays live matches, detailed scorecards, and match summaries.
2. **Web Scraping Engine (Server):** Responsible for fetching live cricket scores from cricket websites such as Cricbuzz using web scraping techniques. It parses HTML documents using BeautifulSoup library and extracts relevant information to update the user interface.
3. **Data Management Layer**: Manages the data flow between the user interface and the web scraping engine. It stores temporary data such as live match details and scorecards retrieved from the web scraping engine.
4. **Integration Layer:** Integrates the user interface with the web scraping engine to ensure seamless communication and data exchange between the client and server components.

User Interface Design using Tkinter:

The user interface design of the cricket score application is implemented using Tkinter library in Python. It follows a simple and intuitive design to provide users with easy access to live cricket scores and match details. Theuser interface consists of the following components:

1. **Live Matches List:** Displays a list of live matches for users to select from. It includes match details such as teams, venue, and time.

2. **Detailed Scorecard:** Provides detailed scorecards for selected matches, including runs scored, wickets taken, overs bowled, and current match status.
3. **Match Summary:** Presents match summaries with key highlights and events for better understanding.
4. **Navigation Buttons:** Allows users to navigate between different sections of the application, such as live matches, detailed scorecards, and match summaries.

Data Model and Schema Design:

The data model of the cricket score application consists of the following entities:

1. **Match:** Represents a live cricket match with attributes such as match ID, teams, venue, and time.
2. **Scorecard:** Stores detailed scorecard information for each match, including runs scored, wickets taken, overs bowled, and current match status.
3. **Match Summary:** Contains match summaries with key highlights and events for selected matches.

The data model follows a relational database schema design, where each entity is represented as a table with attributes as columns. It ensures efficient data storage and retrieval while maintaining data integrity and consistency.

Algorithm Design for Web Scraping using BeautifulSoup:

The web scraping engine uses BeautifulSoup library in Python to fetch live cricket scores from cricket websites such as Cricbuzz. The algorithm design for web scraping involves the following steps:

1. Requesting the Web Page: The engine initiates a request to the cricket website using HTTP requests to retrieve the live scores web page.

2. Parsing HTML Documents: BeautifulSoup library parses the HTML documents of the live scores web page to extract relevant information such as match details, scorecards, and match summaries.

3.Extracting Data: The engine extracts live match details, scorecards, and match summaries from the parsed HTML documents using BeautifulSoup's methods for navigating and searching the document tree.

4.Updating the User Interface: The extracted data is sent to the user interface component to update the display with live cricket scores and match details.

By following this algorithm design, the web scraping engine efficiently retrieves live cricket scores from cricket websites and updates the user interface of the cricket score application in real-time.

Implementation

Detailed Explanation of the Implementation Phase:

The implementation phase of the cricket score project involves translating the design specifications into working code. This phase includes writing code, integrating components, and testing the functionality of the application. The implementation phase follows the software development lifecycle (SDLC) and involves the following steps:

1. **Setting Up Development Environment:** Install necessary libraries and dependencies such as Tkinter for GUI development, BeautifulSoup for web scraping, and any other required packages.

2. **Coding:** Write code to implement the functionality outlined in the design phase. This includes creating classes, functions, and methods to fetch live cricket scores, display match details, and handle user interactions.

3. **Integration:** Integrate different components of the application, such as the user interface built using Tkinter and the web scraping engine using BeautifulSoup. Ensure seamless communication and data exchange between the components.

4. **Testing:** Test the functionality of the application to ensure it meets the specified requirements. Perform unit testing, integration testing, and system testing to identify and fix any bugs or issues.

5. **Debugging:** Debug the code to fix any errors or issues identified during testing. Use debugging tools and techniques to identify and resolve problems in the code.

6. **Optimization:** Optimize the code for performance, efficiency, and scalability. Identify areas for improvement and refactor the code as needed to enhance the application's performance.

7. **Documentation:** Document the implementation details, including code documentation, comments, and user manuals, to facilitate maintenance and future development.

Code Walkthrough of Key Components:

1. **CricketScore Class:** Represents the main class of the cricket score application, responsible for creating the GUI window and handling user interactions.

2. **Web Scraping Methods:** Includes methods for scraping live cricket scores from cricket websites using BeautifulSoup library. Methods such as scrap(), match_details(), match_summary(), match_header(), teams_name(), and team_score() are used for parsing HTML documents and extracting relevant information.

3. **User Interface Components:** Consists of Tkinter widgets such as labels, buttons, and comboboxes for displaying live matches, detailed scorecards, and match summaries. The show_match_details() method is used to update the GUI with live cricket scores and match details.

Integration of Tkinter for GUI and BeautifulSoup for Web Scraping:

• Tkinter is used to create the graphical user interface (GUI) of the cricket score application. It provides a set of widgets and tools for building interactive and visually appealing user interfaces.

• BeautifulSoup is used for web scraping, allowing the application to fetch live cricket scores from cricket websites such as Cricbuzz. It provides methods for parsing HTML documents and extracting relevant information from web pages.

Error Handling and Debugging Strategies:

- Error handling is implemented using try-except blocks to catch and handle exceptions that may occur during the execution of the code.

- Debugging strategies include using print statements, logging, and debugging tools such as pdb (Python Debugger) to identify and fix errors in the code.

- Proper error messages and logging are implemented to provide meaningful feedback to users and developers in case of errors or issues.

By following these implementation strategies and best practices, the cricket score application is successfully developed and tested to provide live cricket scores and match details to users in a user-friendly and efficient manner.

Testing

Testing Methodologies Employed:

1. Unit Testing:

- Unit testing involves testing individual components or units of the code in isolation to ensure they function correctly.
- In the cricket score project, unit testing is employed to test functions and methods responsible for specific tasks such as web scraping, data parsing, and GUI interactions.
- Tools such as Python's unittest framework can be utilized to automate unit tests and verify the correctness of each unit of code.

2. Integration Testing:

- Integration testing focuses on testing the interaction and integration between different components or modules of the application to ensure they work together seamlessly.
- In the cricket score project, integration testing is conducted to verify the communication between the Tkinter-based GUI and the web scraping engine implemented using BeautifulSoup.
- This testing ensures that data is retrieved and displayed correctly in the GUI based on the information scraped from cricket websites.

3. System Testing:

- System testing evaluates the entire system as a whole to ensure it meets the specified requirements and functions as expected in a real world environment.
- In the cricket score project, system testing involves testing the application's functionality from end to end, including fetching live cricket scores, displaying match details, and handling user interactions.
- It verifies that the application performs as intended and provides accurate and up-to-date cricket scores to users.

Test Plan and Test Cases:

1. Unit Test Plan:

- Identify the units of code to be tested, including functions and methods responsible for specific functionalities.
- Define test cases for each unit of code to cover various scenarios and edge cases.
- Develop test scripts using Python's unittest framework to automate the execution of unit tests.
- Execute unit tests and analyze the results to ensure that each unit of code functions correctly.

2.Integration Test Plan:

- Define integration test cases to verify the interaction between the Tkinter-based GUI and the web scraping engine implemented using BeautifulSoup.
- Test scenarios include verifying the display of live cricket scores in the GUI based on the information scraped from cricket websites.
- Execute integration tests to validate the integration between different components of the application.

3.System Test Plan:

- Define system test cases to cover the entire functionality of the cricket score application, including fetching live cricket scores, displaying match details, and handling user interactions.
- Test scenarios include verifying the accuracy of live cricket scores, checking the display of match details, and testing user interactions such as selecting matches and viewing scorecards.
- Execute system tests to evaluate the overall performance and functionality of the application.

Results and Findings from Testing Phase:

1. **Unit Testing Results:** Unit tests verify the correctness of individual units of code, ensuring that each function and method behaves as expected. Results from unit testing confirm that the code functions correctly in isolation.

2. **Integration Testing Results:** Integration tests validate the interaction between different components of the application. Results from integration testing ensure that data is exchanged correctly between the Tkinter-based GUI and the web scraping engine.

3. **System Testing Results:** System tests evaluate the entire functionality of the cricket score application from end to end. Results from system testing confirm that the application performs as intended, providing accurate and up-to-date cricket scores to users and handling user interactions effectively.

4. **Findings:** Any issues or discrepancies identified during testing are documented as findings. These findings may include bugs, errors, or areas for improvement that need to be addressed before the application is released to users.

Deployment

Deployment Environment and Platform: The cricket score application can be deployed on various platforms such as Windows, Linux, and macOS. It is a desktop application that runs locally on the user's machine. The deployment environment requires Python installed along with necessary libraries such as Pillow, BeautifulSoup (bs4), and requests.

Installation Instructions for Dependencies (Pillow, bs4, requests):

Before deploying the cricket score application, ensure that the following dependencies are installed:

1. **Pillow:** It is a Python Imaging Library (PIL) fork that adds support for opening, manipulating, and saving many different image file formats. Install Pillow using the following command: Copy code pip install Pillow

2. **BeautifulSoup (bs4):** It is a Python library for pulling data out of HTML and XML files. Install BeautifulSoup using the following command: Copy code pip install bs4

3. **Requests:** It is a Python library for making HTTP requests. Install Requests using the following command: Copy code pip install requests

Steps for Running the Application on Different Platforms:

Windows:

1. Open Command Prompt or PowerShell.

2. Navigate to the directory containing the cricket score application files.

3. Run the following command to execute the application: Copy code python filename.py Replace filename.py with the name of the Python script containing the application code

. Linux and macOS:

1. Open Terminal.

2. Navigate to the directory containing the cricket score application files.

3. Run the following command to execute the application: Copy code python3 filename.py Replace filename.py with the name of the Python script containing the application code.

Note: Ensure that Python is installed on the system and added to the system's PATH environment variable. Additionally, make sure that the necessary dependencies (Pillow, bs4, requests) are installed using the instructions provided above.

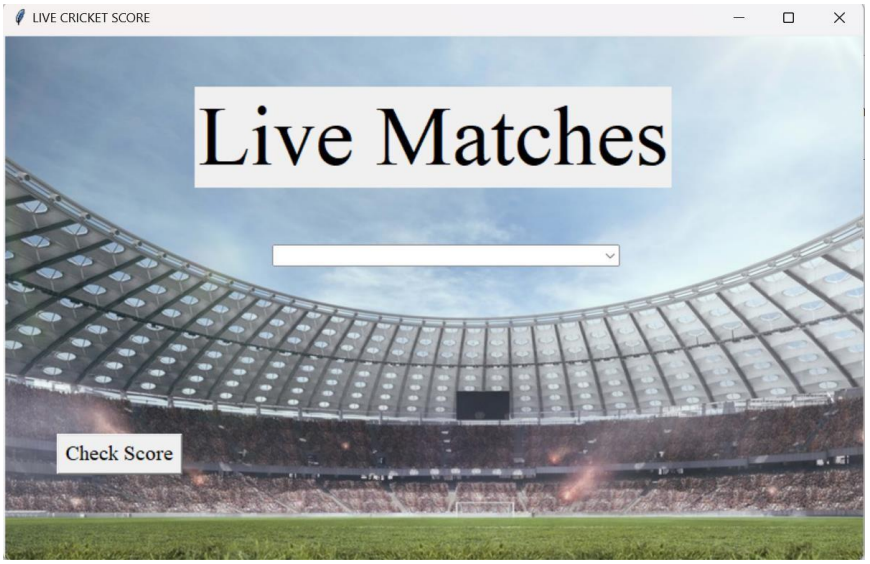
By following these deployment steps and instructions, users can successfully run the cricket score application on different platforms and enjoy real-time updates of live cricket scores and match details.

User Manual

Instructions for Users on How to Use the Application:

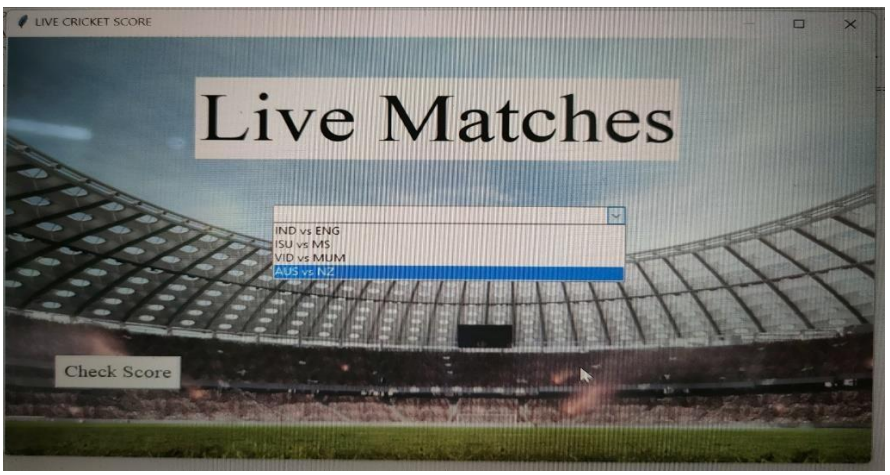
1. Launching the Application:

- Double-click the application icon or navigate to the directory containing the application files and execute the Python script (filename.py) using the command prompt or terminal.
- **Screenshot :**



2. Viewing Live Matches:

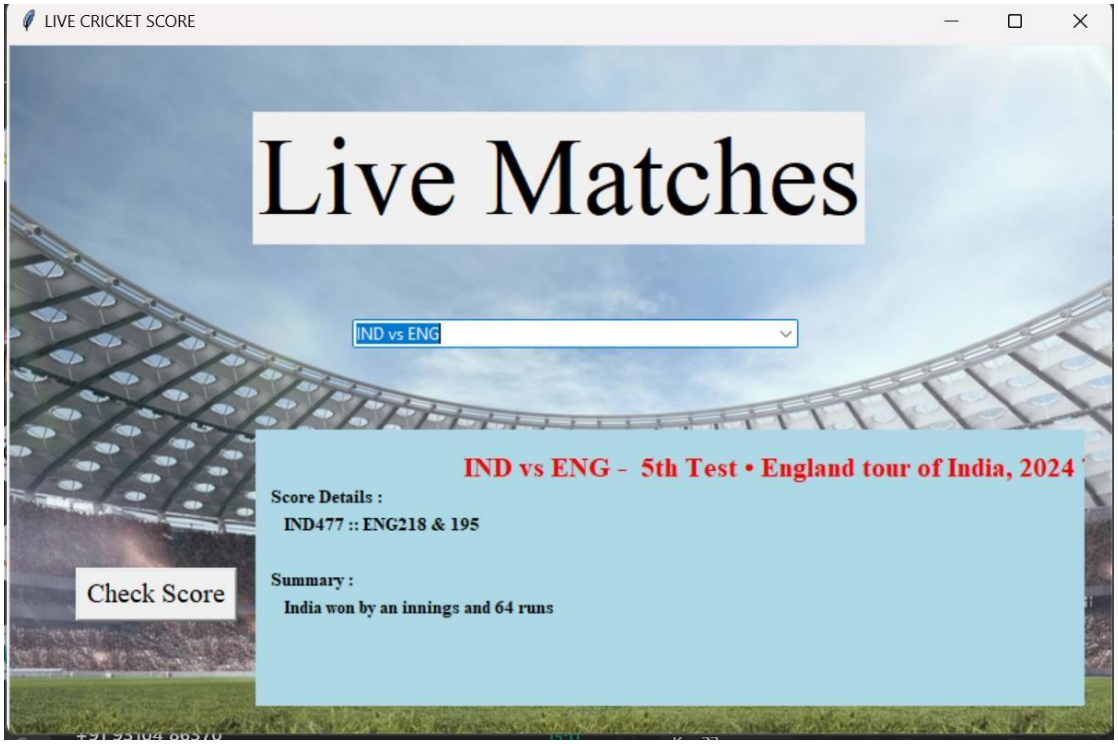
- Upon launching the application, you will see a list of live cricket matches displayed on the screen. • Use the mouse or keyboard to select a match from the list by clicking on it or using the arrow keys and pressing Enter.
- **Screenshot:**



3. Viewing Detailed Scorecards:

- After selecting a match, the detailed scorecard for the selected match will be displayed on the screen.
- The scorecard includes information such as runs scored, wickets taken, overs bowled, and the current match status.

- **Screenshot:**



Conclusion

Summary of the Project Achievements:

The cricket score project aimed to develop a desktop application using Python, Tkinter, and BeautifulSoup to provide users with real-time updates of live cricket scores and match details. Throughout the implementation phase, the project successfully achieved the following objectives:

1. Implemented a user-friendly graphical interface using Tkinter library, allowing users to view live matches, detailed scorecards, and match summaries.

2. Integrated web scraping techniques using BeautifulSoup to fetch live cricket scores from cricket websites such as Cricbuzz.

3. Implemented error handling and debugging strategies to ensure the reliability and stability of the application.

4. Provided deployment instructions for running the application on different platforms and troubleshooting tips for common issues.

Challenges Faced and Lessons Learned:

During the development of the cricket score project, several challenges were encountered, including:

1. **Web Scraping Challenges:** Extracting accurate and up-to-date information from cricket websites posed challenges due to the dynamic nature of web content and the need for robust error handling mechanisms.

2. **GUI Design:** Designing an intuitive and visually appealing user interface using Tkinter required careful consideration of layout, styling, and user interaction elements.

3. **Dependency Management:** Ensuring proper installation and compatibility of dependencies such as Pillow, bs4, and requests across different operating systems presented challenges.

From these challenges, valuable lessons were learned, including the importance of thorough testing, effective error handling, and continuous improvement in user interface design and web scraping techniques.

Future Enhancements and Potential Areas for Improvement:

Despite the successful implementation of the cricket score project, there are several potential areas for future enhancements and improvements, including:

1. **Enhanced User Experience:** Incorporating additional features such as notifications for match updates, user customization options for displaying match preferences, and integration with social media platforms for sharing match updates.
2. **Performance Optimization:** Optimizing the application's performance by implementing caching mechanisms, asynchronous web scraping techniques, and improving data retrieval efficiency.
3. **Internationalization and Localization:** Adding support for multiple languages and localizing the application to cater to a broader audience of cricket enthusiasts worldwide.
4. **Error Handling and Logging:** Implementing more robust error handling mechanisms and logging to provide better feedback to users and facilitate debugging and maintenance.

By addressing these potential areas for improvement, the cricket score application can evolve into a more comprehensive and user-friendly platform for accessing live cricket scores and match details, catering to the diverse needs of cricket fans globally.

Reference

- https://youtu.be/T0dbrEzLpWA?si=PHY4I3j1b038fb0O
- cricscore/cricscore.ipynb at main · Vishwa07dev/cricscore · GitHub
- https://www.cricbuzz.com/