AJEENKYA
D Y PATIL UNIVERSITY
THE INNOVATION UNIVERSITY

School of Engineering

“CRICSCORE GUI”

submitted in partial fulfillment of the requirements

of the degree of

Bachelor of Technology

By

Vinita Choudhary- 2021-B-09192003

Dhanashree Pogade – 2021-B-11032003C

Monika Choudhary -2021-B-25092003A

November 2023

School of Engineering

Ajeenkya D Y Patil University, Pune

Overview:

Live cricket scores refer to real-time updates on the ongoing cricket matches around the world. These scores provide ball-by-ball updates, displaying information such as the runs scored, wickets fallen, overs bowled, and other relevant details. They are available through various mediums:

Sports Websites: Websites like ESPN Cricinfo, Cricbuzz, and the official websites of cricket boards usually offer live scores.

Mobile Apps: There are several mobile apps dedicated to providing live cricket scores. Apps like Cricbuzz, ESPNcricinfo, and the official ICC app are popular choices.

Television: Some sports channels broadcast live scores alongside the ongoing matches.

Social Media: Platforms like Twitter often have accounts that provide live updates on matches through tweets.

These live scores are crucial for fans who can't watch the match live but want real-time updates on the game's progress. They help keep track of the match's momentum, key events, and performances of players.

Imports:

<u>The required libraries/modules are imported:</u>

Tkinter: Tkinter is a Python library used for creating graphical user interfaces (GUIs). It's a standard GUI toolkit that comes with Python, making it easily accessible for developers to create desktop applications with buttons, labels, text boxes, and other GUI elements

PIL (Pillow): PIL (Python Imaging Library) has transitioned into Pillow, an actively maintained fork that provides extensive image processing capabilities in Python. Here's a detailed explanation of Pillow: Pillow is a powerful library for image processing in Python. It enables developers to perform various operations on images, including opening, manipulating, enhancing, and saving different image file formats.

BeautifulSoup: BeautifulSoup is a popular Python library used for web scraping and parsing HTML and XML documents. It simplifies the process of extracting data from web pages by providing tools to navigate, search, and modify the parsed HTML/XML content.

Requests: The Requests library in Python is a powerful HTTP library used to send HTTP requests and handle responses. It simplifies the process of making HTTP requests and interacting with web services by providing a user-friendly API.

Re: The **re** (regular expression) library in Python is a powerful tool for working with regular expressions, enabling pattern matching and manipulation within strings. It allows for sophisticated text processing by defining patterns and rules to search, match, and manipulate text data.

Class CricketScore:

Constructor (__init__): The CricketScore class is designed to create a graphical application that displays live cricket match details fetched from the cricbuzz website.

Constructor (__init__): The constructor method __init__ is the first method called when an instance of the class is created. It initializes the attributes and sets up the initial state of the Cricket Score object.

Methods:

Scrap() Method:

The **Scrap()** method is responsible for fetching live match details from the cricbuzz website using BeautifulSoup for web scraping. It performs the following steps:

1. HTTP Request and Parsing:
- Utilizes the requests library to make an HTTP request to the cricbuzz.com website.
- Parses the obtained HTML content using BeautifulSoup to extract live match data.
2. Data Extraction:
- Searches and retrieves the specific HTML elements containing live match details using BeautifulSoup methods (find(), find_all(), etc.).
3. Returns:
- Returns the scraped results, typically a list of HTML elements containing live match information.

Match_details() Method:

The **match_details()** method is responsible for parsing the scraped data obtained from scrap() to extract various match details like team names, match summary, and scorecard. It performs the following:

1. Parsing and Data Extraction:
- Parses the HTML elements obtained from scrap().
- Extracts relevant match details such as team names, summary, and scorecard for each live match.
2. Data Structuring:

- Structures the extracted data into a dictionary format,mapping each match to its respective details.

3. Returns:

- Returns a dictionary containing match details with keys as team names and values as dictionaries of specific match information.

Helper Methods (e.g., team_score, teams_name, match_summary, match_header):

These methods are auxiliary functions used within match_details() for extracting specific information from the scraped HTML content. Each helper method performs specific parsing tasks, extracting details like team scores, team names, match summaries, and match headers by utilizing BeautifulSoup's methods tailored to retrieve the required information.

Show_match_details() Method:

The show_match_details() method is responsible for displaying the selected match's details within a frame in the GUI. It performs the following:

1. Frame Creation:

- Creates a frame within the GUI window to display the match details.

2. Fetching Selected Match Data:

- Retrieves the details of the selected match from the previously structured data obtained by match_details().

3. GUI Display:

- Displays the fetched match details (team names, scorecard, summary, etc.) within the created frame using Tkinter's Label widgets.

Functionality:

- Web Scraping: Utilizes BeautifulSoup to extract live match details from cricbuzz.com. The CricketScore class harnesses the power of BeautifulSoup to conduct web scraping, a technique employed to extract live match details from the cricbuzz.com website. This process involves initiating an HTTP GET request to the site, receiving an HTML response, and parsing this content using BeautifulSoup. By navigating the HTML structure, the class locates and extracts specific elements containing crucial match information, such as team names, match summaries, and scorecard details.

- Data Handling: Parses HTML content to extract essential match information like team names, summary, and scorecard. Following the extraction of live match details through web scraping, the CricketScore class engages in comprehensive data handling tasks. It meticulously parses the HTML content retrieved from cricbuzz.com, organizing and structuring this data into a manageable format. This involves decoding the HTML structure to extract essential match information, refining it, and organizing it systematically. The class ensures the extracted data is cleansed of extraneous characters or formatting inconsistencies, ensuring a coherent and organized dataset ready for display.

- GUI Interaction: Displays fetched match details in the GUI and enables user interaction to view specific match data. The CricketScore class seamlessly integrates with Tkinter, a prevalent Python GUI library, to present the fetched match details within a graphical user interface (GUI). Leveraging Tkinter's capabilities, the class creates an interactive and visually appealing interface, incorporating elements like labels, frames, and buttons. These GUI components serve as vessels to display the live match details fetched and processed earlier. Users can navigate and interact with the GUI, allowing them to select specific matches and delve deeper into individual match data through user-initiated actions like button clicks or selection events.

Potential Improvements:

Error Handling in Python

Python offers a robust error handling mechanism through the try...except block, allowing developers to manage and respond to unexpected errors or exceptions during program execution. The try block encapsulates the code where potential errors might occur, while the except block catches and handles these exceptions gracefully.

By employing error handling techniques, developers can preemptively identify potential issues within their code and respond to them in a controlled manner. For instance, in the context of the CricketScore class, error handling within the web scraping process using BeautifulSoup and requests library can involve catching exceptions related to network connectivity

issues, invalid HTML structures, or unexpected changes in the website's layout.

Handling these exceptions ensures that the application remains resilient even in the face of unforeseen circumstances. It allows the program to gracefully recover from errors, providing users with a smoother and more reliable experience. Python's error handling mechanisms empower developers to write more robust code, enhancing the stability and usability of applications.

Enhancing User Experience with Additional Features :

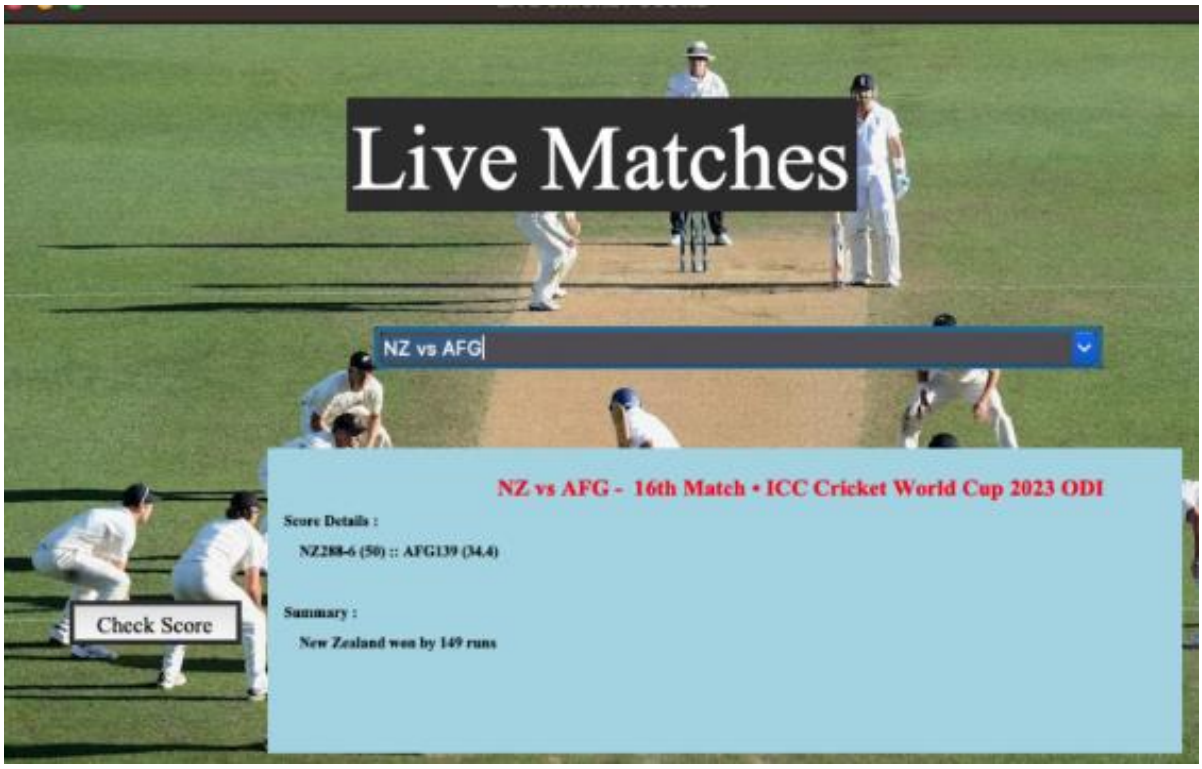
Expanding the functionalities of the CricketScore class by implementing additional features can significantly enrich the user experience. For instance, integrating live score updates that fetch real-time match data at periodic intervals could keep users informed about ongoing matches without manually refreshing the interface. This live update feature enhances user engagement and ensures they receive the most recent match information promptly.

Moreover, incorporating match schedules or more detailed statistics about individual matches, player performances, or historical data could provide users with comprehensive insights into cricket events. By presenting match schedules, users can plan and anticipate upcoming matches of interest, while detailed statistics offer a deeper understanding of team dynamics and player performances, catering to the preferences of avid cricket enthusiasts.

Overall, augmenting the CricketScore application with these additional features not only enhances its functionality but also elevates the user experience, making it more informative,

engaging, and versatile. Integrating live updates, match schedules, or in-depth statistics empowers users with a holistic view of cricket events, catering to a wider spectrum of preferences and interests.

Outcomes:



Conclusion:

The provided Python code combines web scraping techniques with a Tkinter-based GUI to display live cricket scores. It successfully retrieves data from cricbuzz.com and presents it to the user in a graphical interface. However, it's essential to monitor potential changes in the website structure for continued functionality.