1. High-level Design

Describe which architectural pattern you would use to structure the system. Justify your answer.

For our project, we would use pipelining because it is easy to add and remove filters and multiple operations can be performed simultaneously. Pipelining is ideal for this project because there can be multiple stages for different tasks in processing data. Additionally, individual stages can be modified without influencing the other parts of the program. This is important because there might be adjustments required for certain stages to make them more efficient and accurate.

2. Low-level Design

Discuss which design pattern family might be helpful for implementing this project. Justify your answer, providing a code or pseudocode representation and an informal class diagram.

Behavioral Design Pattern:
For a bet bot that is capable of making predictions based on the current player statistics, the Strategy pattern from the Behavioral Design Pattern is a good choice to work on. The reasons for that are mentioned.

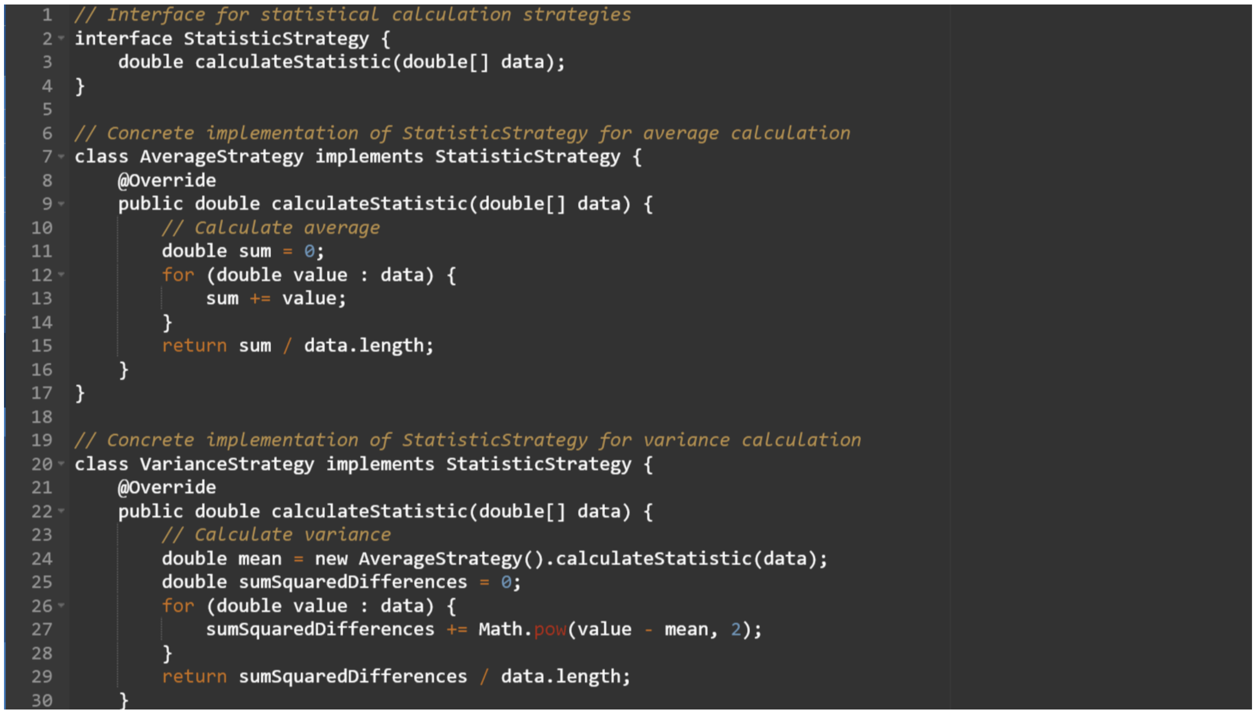
Flexible Algorithm:
This is a strategy that involves us to define a group of algorithms, that encapsulate one another, to form an extra class. This makes these classes interchangeable, The flexibility allows us to make crucial calculations which can be passed through different classes to be processed and returned.

Easy Extendability:
By implementing the strategy pattern adding or deleting existing stats becomes easy. We can implement new algorithms that could make better predictions by easily extending and modifying the core of the betting bot. This promotes maintenance and adding additional features.

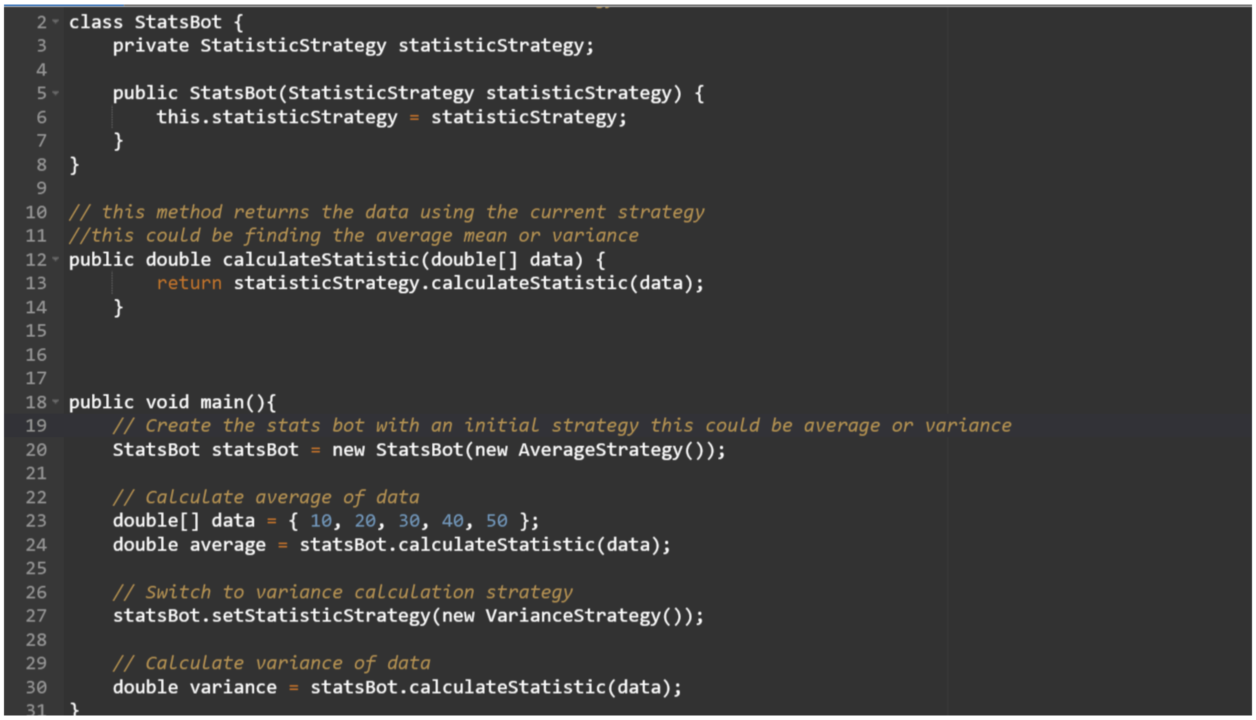
Encapsulation as a strategy:
Created algorithms can be encapsulated in the classes of their own, this reduces the dependability of one class over the other and also allows us to make major changes to one class without making additional changes to the program in any other class.

Pseudocode:

This is a model of how the classes could be implemented to create a betting bot to use multiple algorithm strategy models.

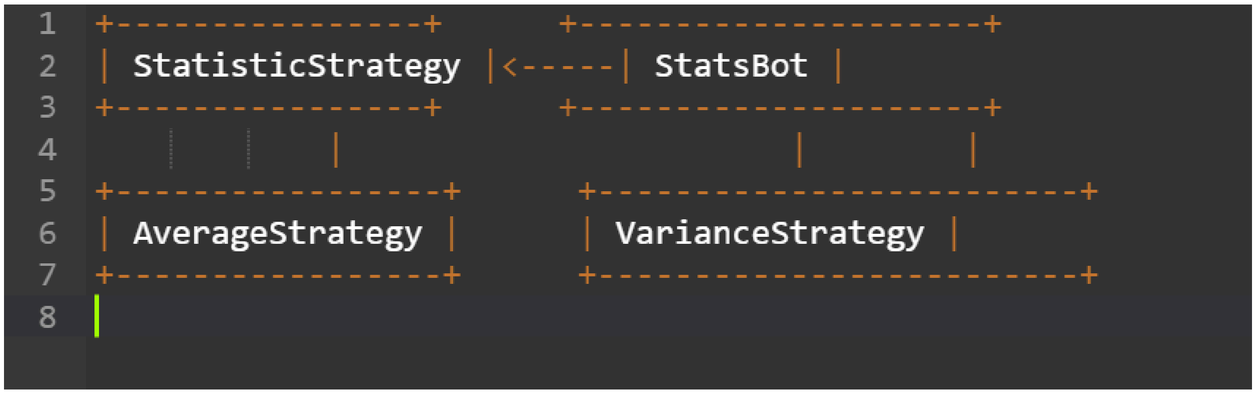


```java
// Interface for statistical calculation strategies
interface StatisticStrategy {
    double calculateStatistic(double[] data);
}

// Concrete implementation of StatisticStrategy for average calculation
class AverageStrategy implements StatisticStrategy {
    @Override
    public double calculateStatistic(double[] data) {
        // Calculate average
        double sum = 0;
        for (double value : data) {
            sum += value;
        }
        return sum / data.length;
    }
}

// Concrete implementation of StatisticStrategy for variance calculation
class VarianceStrategy implements StatisticStrategy {
    @Override
    public double calculateStatistic(double[] data) {
        // Calculate variance
        double mean = new AverageStrategy().calculateStatistic(data);
        double sumSquaredDifferences = 0;
        for (double value : data) {
            sumSquaredDifferences += Math.pow(value - mean, 2);
        }
        return sumSquaredDifferences / data.length;
    }
```



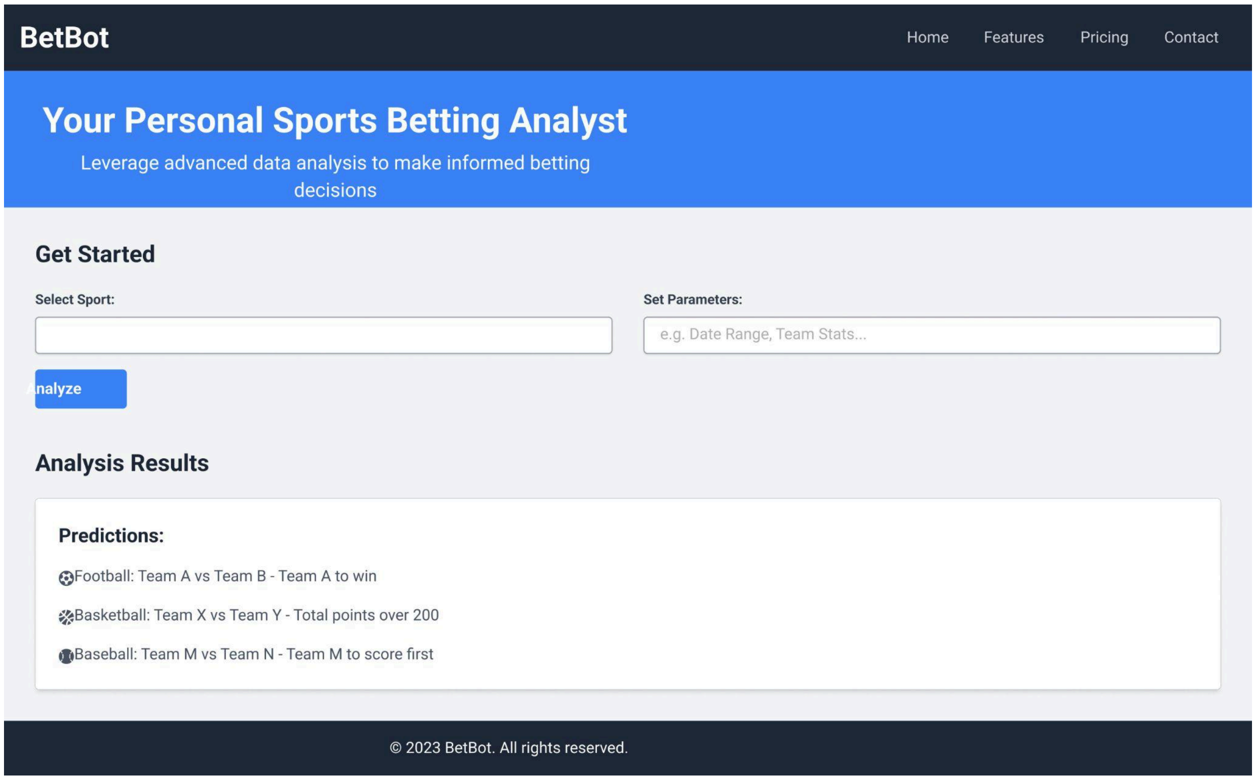
```java
class StatsBot {
    private StatisticStrategy statisticStrategy;

    public StatsBot(StatisticStrategy statisticStrategy) {
        this.statisticStrategy = statisticStrategy;
    }
}

// this method returns the data using the current strategy
//this could be finding the average mean or variance
public double calculateStatistic(double[] data) {
        return statisticStrategy.calculateStatistic(data);
    }



public void main(){
    // Create the stats bot with an initial strategy this could be average or variance
    StatsBot statsBot = new StatsBot(new AverageStrategy());

    // Calculate average of data
    double[] data = { 10, 20, 30, 40, 50 };
    double average = statsBot.calculateStatistic(data);

    // Switch to variance calculation strategy
    statsBot.setStatisticStrategy(new VarianceStrategy());

    // Calculate variance of data
    double variance = statsBot.calculateStatistic(data);
}
```

Class Diagram:



```
+----------------+      +---------------------+
| StatisticStrategy |<-----| StatsBot |
+----------------+      +---------------------+
            |                       |        |
+-----------------+      +-------------------------+
| AverageStrategy |      | VarianceStrategy |
+-----------------+      +-------------------------+
```

3. Design Sketch



Your Personal Sports Betting Analyst

Leverage advanced data analysis to make informed betting decisions

Get Started

Select Sport:

Set Parameters: e.g. Date Range, Team Stats...

Analysis Results

Predictions:

Football: Team A vs Team B - Team A to win

Basketball: Team X vs Team Y - Total points over 200

Baseball: Team M vs Team N - Team M to score first

© 2023 BetBot. All rights reserved.

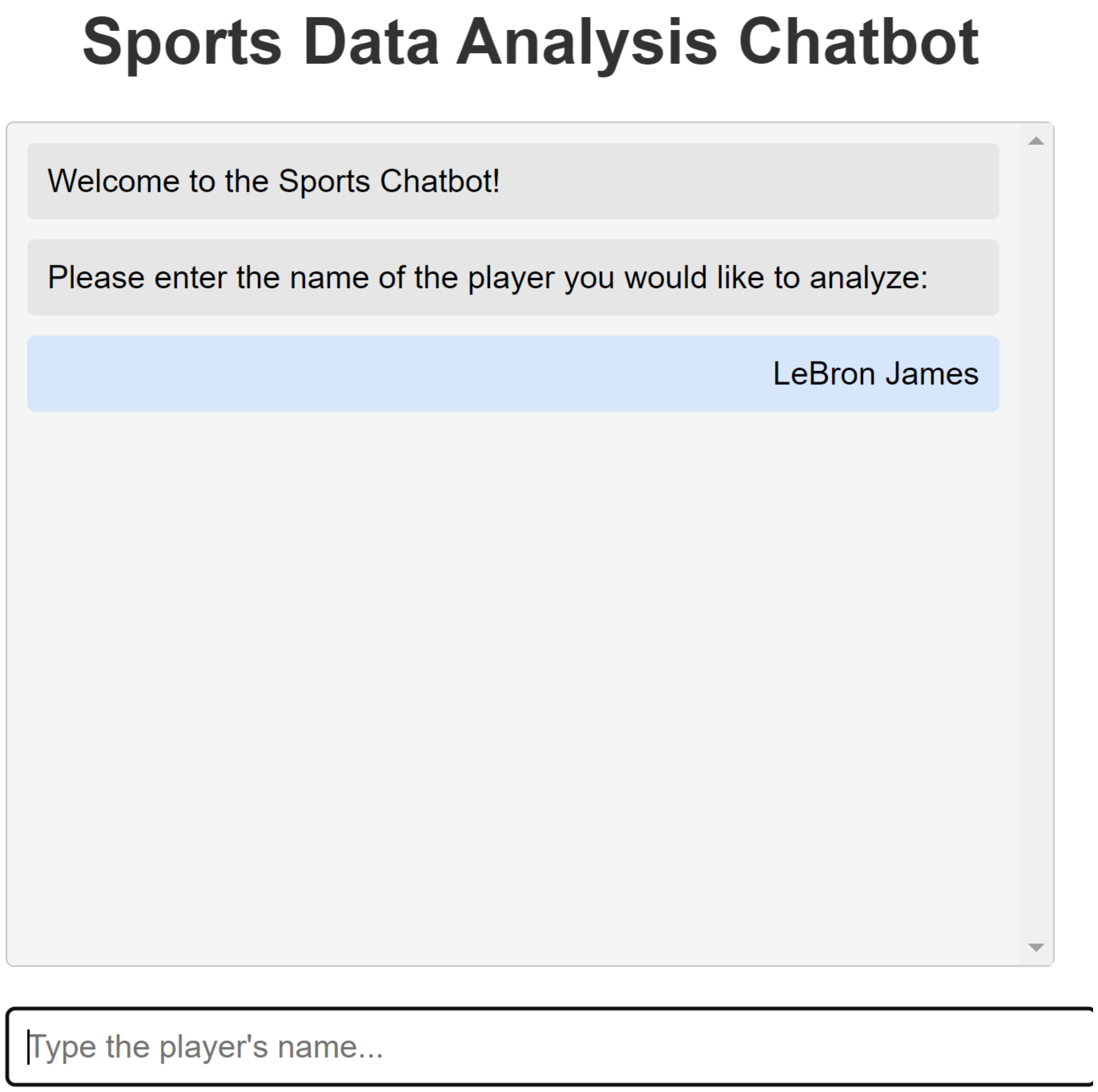
The design for our bot, as depicted in the provided sketch, focuses clarity and ease of use to enhance the user experience. The interface is streamlined, with a simple navigation bar that includes essential links such as Home, Features, Pricing, and Contact, ensuring users can easily find more information. The main interaction area is divided into clear, distinct sections for starting an analysis and viewing results. Input fields for 'Select Sport' and 'Set Parameters' are designed to be intuitive, inviting users to begin their analysis with minimal clicks or typing required. The 'Analyze' button is prominently placed to prompt immediate action. Below, the

'Analysis Results' are cleanly organized, providing straightforward predictions for various sports, which helps users quickly interpret the output. Overall, the interface prioritizes functionality with a minimalistic approach to avoid overwhelming the user with too many options or excessive information at once.

Process Deliverable

Prototyping: submit a prototype of your system

Visual Prototype:

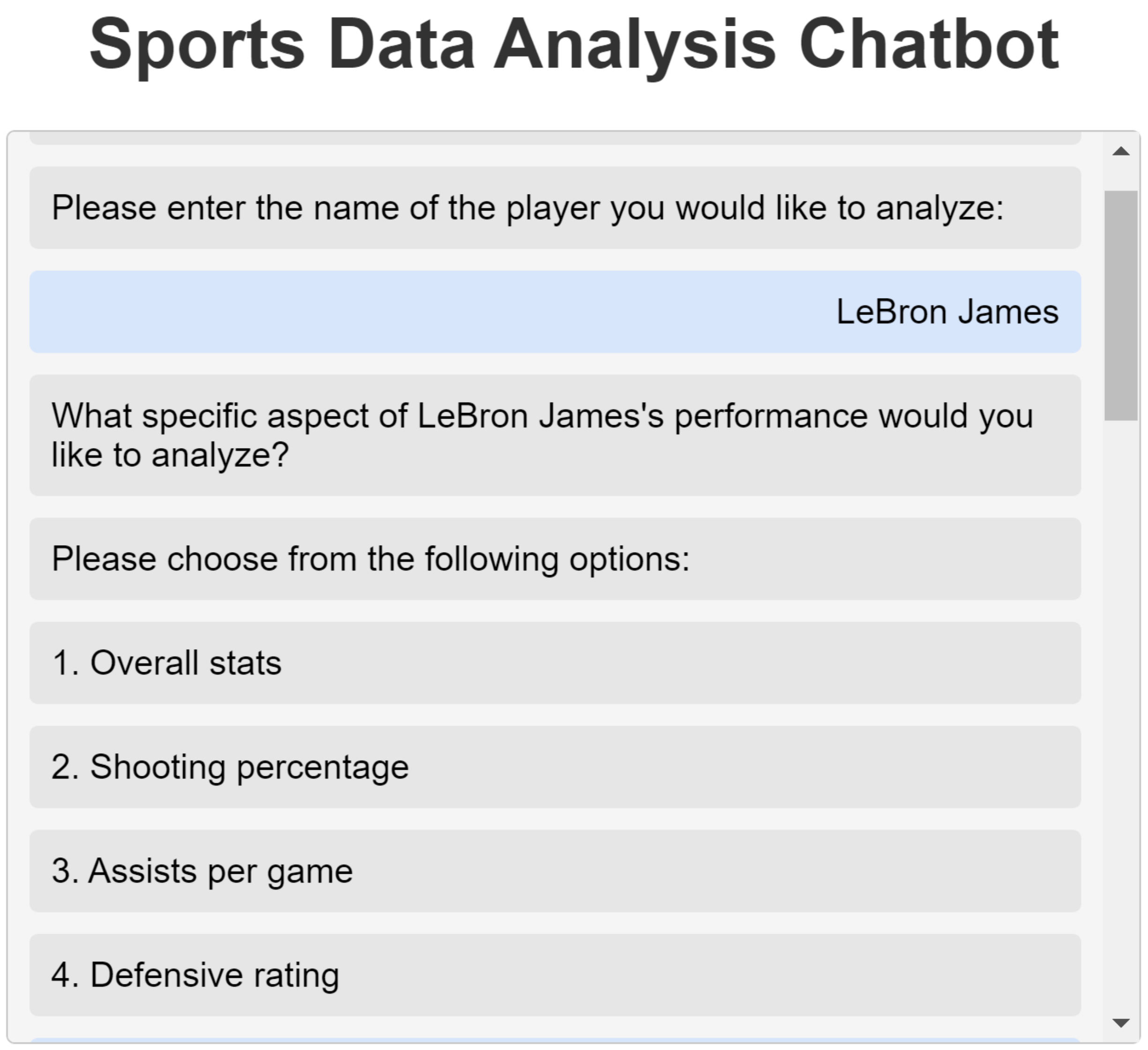


Sports Data Analysis Chatbot

Welcome to the Sports Chatbot!

Please enter the name of the player you would like to analyze:

LeBron James



Sports Data Analysis Chatbot

Please enter the name of the player you would like to analyze:

LeBron James

What specific aspect of LeBron James's performance would you like to analyze?

Please choose from the following options:

1. Overall stats

2. Shooting percentage

3. Assists per game

4. Defensive rating

This is an example of how the user would interact with the chatbot. In this example the chatbot prompts the user to input a player's name and then prompts them to enter what aspect of their game they would like to see. We will try and implement functionality in this way but we want to imagine how a user would go through our program to see if it is something that could actually be used. Visualizing our project like this will also help us to make tweaks before we start trying to implement something that is flawed. This section is also different to the prediction part of our program.

Kanban: a list of prioritized tasks from your task management system:

1. Define the Pipelining Stages:
    - Identify and clearly outline the stages required for data processing in the bet bot system. Each stage should represent a specific task that contributes to the overall prediction-making process.

2. Implement the Strategy Pattern:
    - Develop the algorithm classes for different prediction strategies.
    - Code the Strategy interface and the concrete strategy classes.
    - Ensure that algorithms are encapsulated and interchangeable without affecting the client.

3. Interface and User Experience Design:
    - Finalize the UI/UX design for the main interaction area, including the sport selection and parameter setting.
    - Implement the 'Analyze' button with event handling to trigger the analysis process.

4. Develop Analysis and Prediction Engine:
    - Create the backend service that will process the input parameters and generate predictions.
    - Ensure that the prediction engine is modular to allow for easy updates to the algorithms used for predictions.

5. Results Display Logic:
    - Code the logic for displaying the analysis results in an organized and readable format.
    - Implement the functionality to handle different types of sports and their respective data visualization needs.

- Karan Reddy
- Aidan
- Aneesh
- Soham