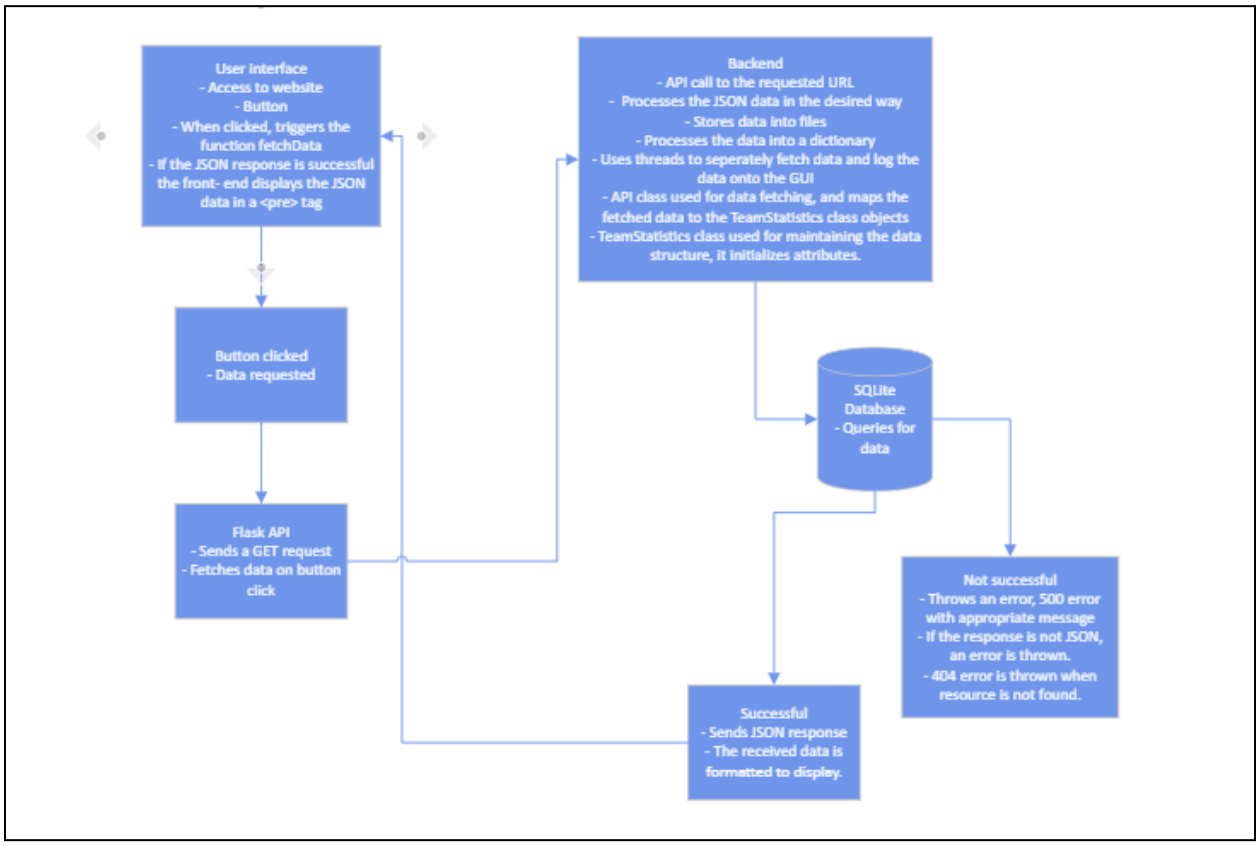
Name: Abhishek (Abi) Parekh
Project #4

Model:



Source code:
(App.js) Function fetchData is used for requesting the JSON data be sent when the button is pressed.

```
const handleFetchData = async () => {
   console.log("retrieving data...")
   try {
       let response = await fetch("http://127.0.0.1:5000/api/team_statistics");
       if (response.ok) {
           let data = await response.json();
           setTeamStats(data);
           setError(null);
       } else {
           let errorText = await response.text();
           console.error(`Unexpected Response: ${errorText}`);
           setError(`Unexpected response: ${errorText}`);
       }
   } catch (error) {
       console.error("There was an error fetching the data", error);
       setError(error.toString());
   }
```

```
};
```

(app.py) multi-threading

```
def communication_thread_function():
    api_call = API()
    team_stats = api_call.get_team_statistics()

    try:
        # Save to the database
        for stat in team_stats:
            team = Team(team_name=stat.team_name, team_number=stat.team_number,
team_code=stat.team_code, score=stat.score)
            db.session.add(team)
        db.session.commit()
    except Exception as e:
        print(f"Error saving to database: {e}")
        db.session.rollback()
    finally:
        db.session.close()

    global team_stats_cache
    team_stats_cache = [team_stat.add_to_tuple() for team_stat in team_stats]
    print("Data fetched:", team_stats_cache)


def presentation_thread_function():
    while not team_stats_cache:
        print("Presentation thread: Waiting for data from the communication thread...")
        threading.Event().wait(timeout=1)
    print("Presentation thread: Data ready for presentation!")
```

(App.css) For styling the button

```
.fetch-data-btn {
    background-color: blue;
    border: none;
    border-radius: 8px;
    padding: 20px 40px;
    font-size: 1.5em;
    cursor: pointer;
    transition: background-color 0.3s ease;
    color: white;
    text-decoration: none;
    text-align: center;
    margin: 0 auto;
    display: block;
}

.fetch-data-btn:hover {
 background-color: #4faed9;
```

```
}

.fetch-data-btn:active {
 background-color: #3f91b9;
}
```

(api.py) used for requesting JSON data

```
class API:
    def __init__(self):
        self.base_url = "https://sports.snoozle.net/search/nfl/searchHandler?"
        self.file_type = "inline"
        self.stat_type = "teamStats"
        self.season = 2020
        self.http_client = requests.Session()

    def get_team_statistics(self):
        team_stats_list = []

        for team_name in range(1, 33):  # iterate over the teams from 1-32
            url =
f"{self.base_url}fileType={self.file_type}&statType={self.stat_type}&season={self.seas
on}" \
                  f"&teamName={team_name}"
            response = self.http_client.get(url)
            response_json = response.json()
            print("Request URL:", url)
            print("Response Status Code:", response.status_code)
            print("Response Text:", response.text)  # Print the response content
            print("JSON Response:", response_json)
            print()
```

(teamDetail.py) used for initializing attributes

```
# team stats class
# Import into api.py

class TeamStatistics:
    def __init__(self, team_name, team_number, team_code, score):
        """
        Initialize TeamStatistics instance
        :param team_name: Name of the team
        :param team_number: Identifying team by their number
        :param team_code: The code identifying the team
        :param score: Score of the team
        """

        self.team_name = team_name
        self.team_number = team_number
        self.team_code = team_code
        self.score = score

    def __str__(self):
```

```
        # return a formatted string with the team attributes
        return f"Team Name: {self.team_name}, Team Code: {self.team_code}, Score:
{self.score}"
```

(app.py) Setting up the database

```
app.config['SQLALCHEMY_DATABASE_URI'] = 'sqlite:///teams.db'
app.config['SQLALCHEMY_TRACK_MODIFICATIONS'] = False

db = SQLAlchemy(app)


class Team(db.Model):
    id = db.Column(db.Integer, primary_key=True)
    team_name = db.Column(db.String(80), nullable=False)
    team_number = db.Column(db.String(80), nullable=False)
    team_code = db.Column(db.String(80), nullable=False)
    score = db.Column(db.String(80), nullable=False)


with app.app_context():
    db.create_all()
```

To run the program:

- In the terminal run the command *python app.py*
- Then in another terminal window run the command *cd json-gui*
- Then run the command *npm start*
- The site will be up and running