# STATSTRIKEFORCE

Katherine McCarthy, Maxwell Mendenhall, Ryan Sayre

PROJECT OVERVIEW

StatStrikeforce is a cloud-base web application, designed to reinvent the way you track your favorite video game statistics. The application allows for users to track their headshot percentage, Kill/Death ratio, win rate, and many other useful statistics with ease. StatStrikeforce gives players a competitive edge through its prediction feature, which uses machine learning to predict a player's performance and probability of a win in future matches based on recent match performance.

Commit Code

GitHub

Deployment Update
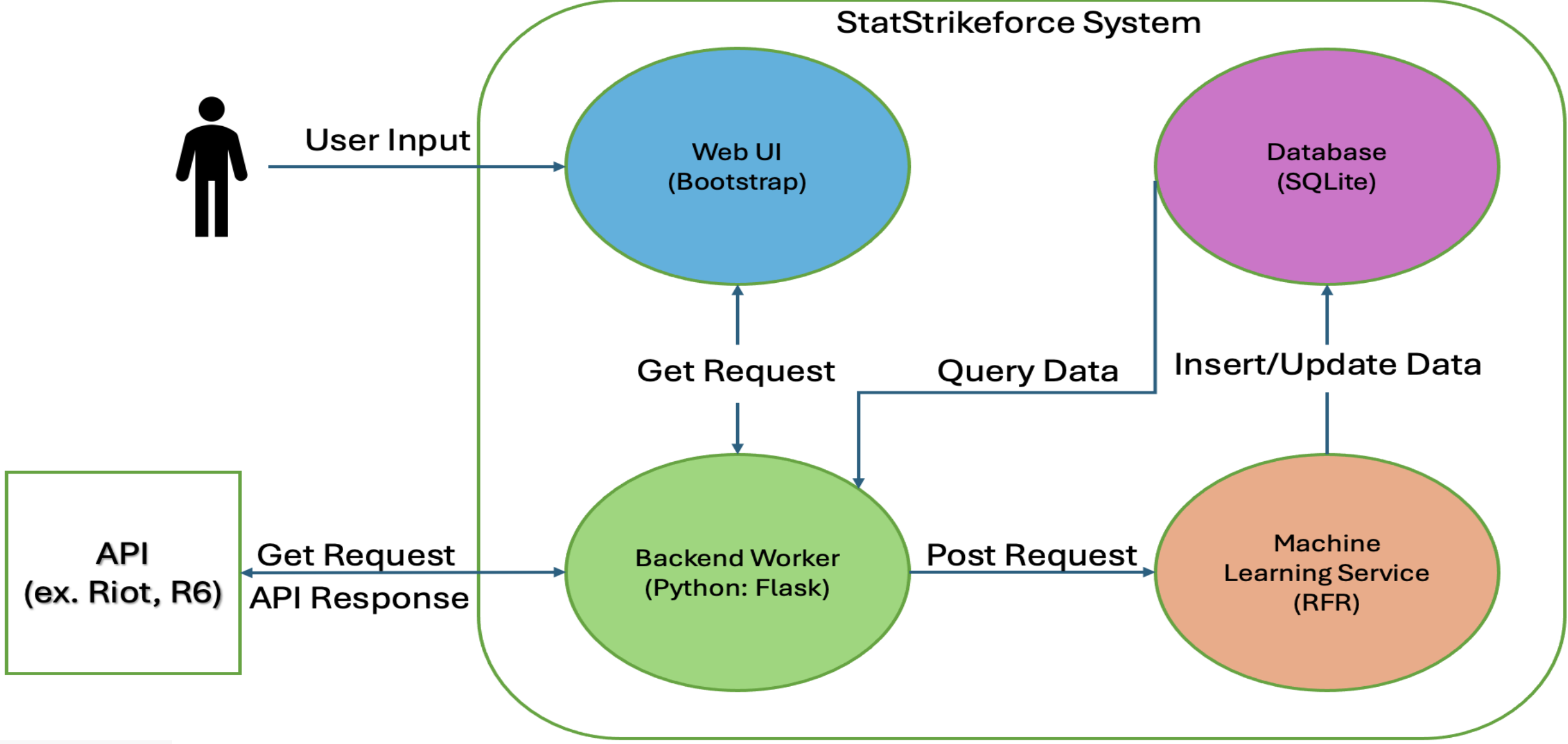
Build Images

Pull Images

docker
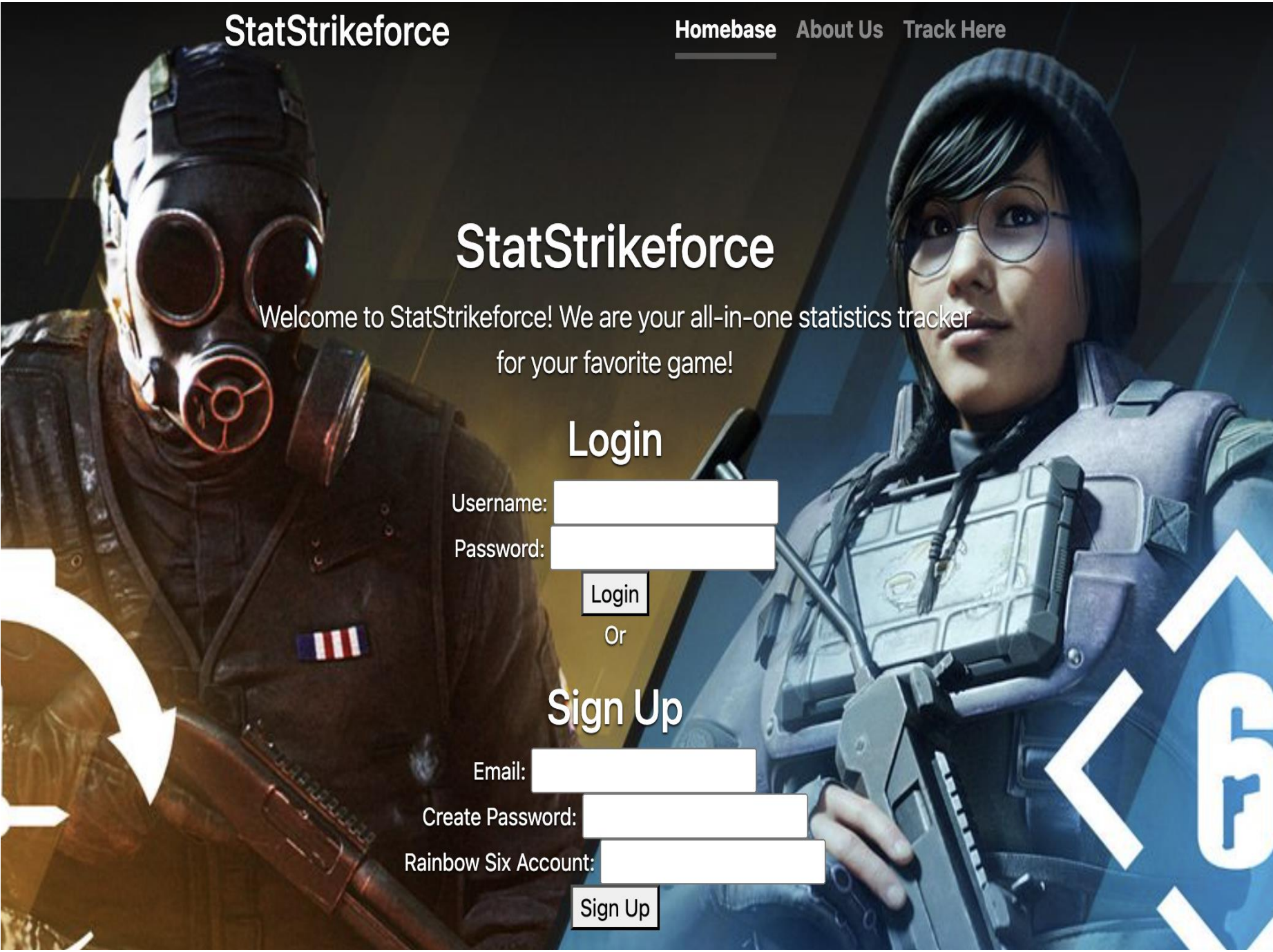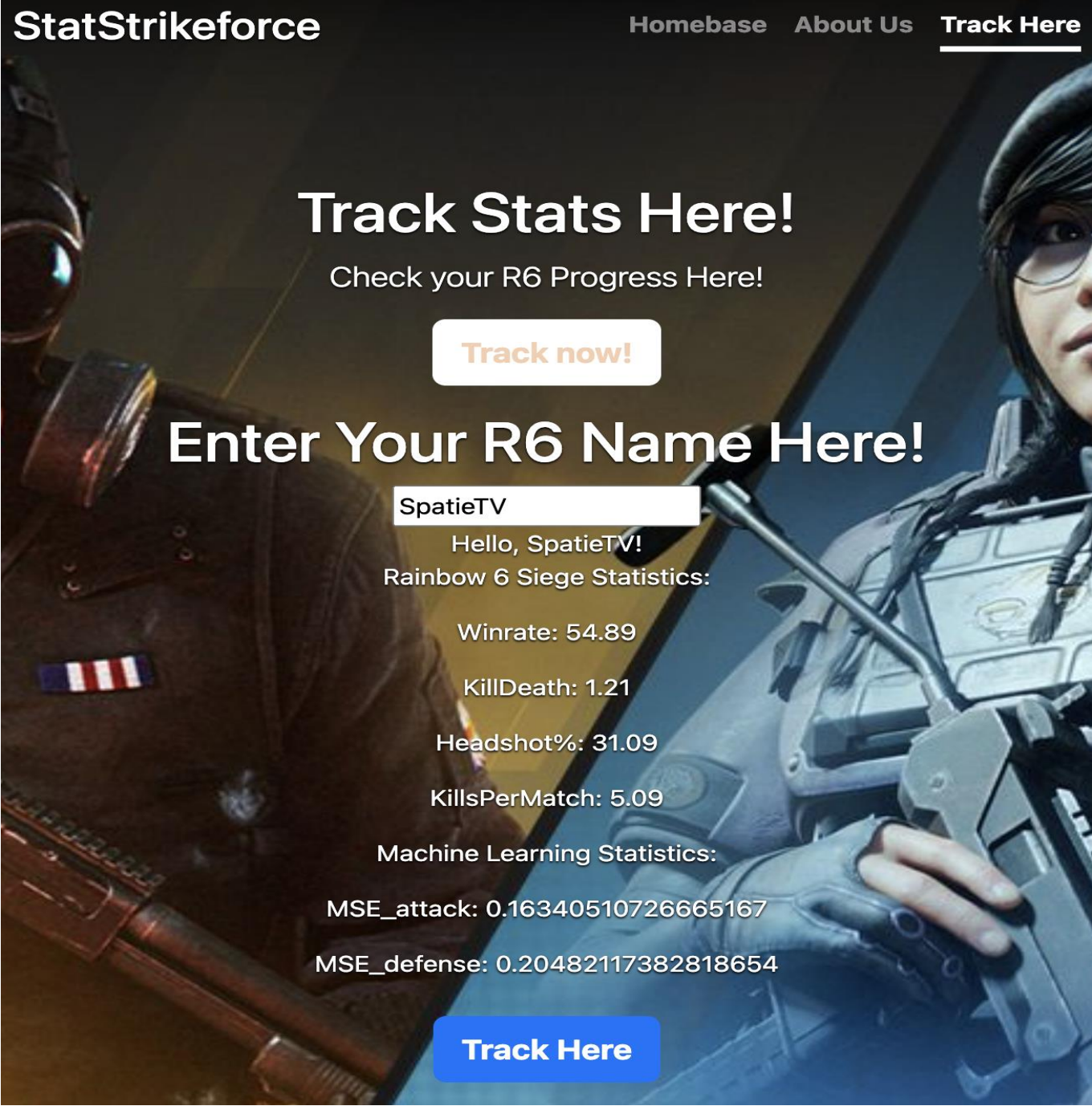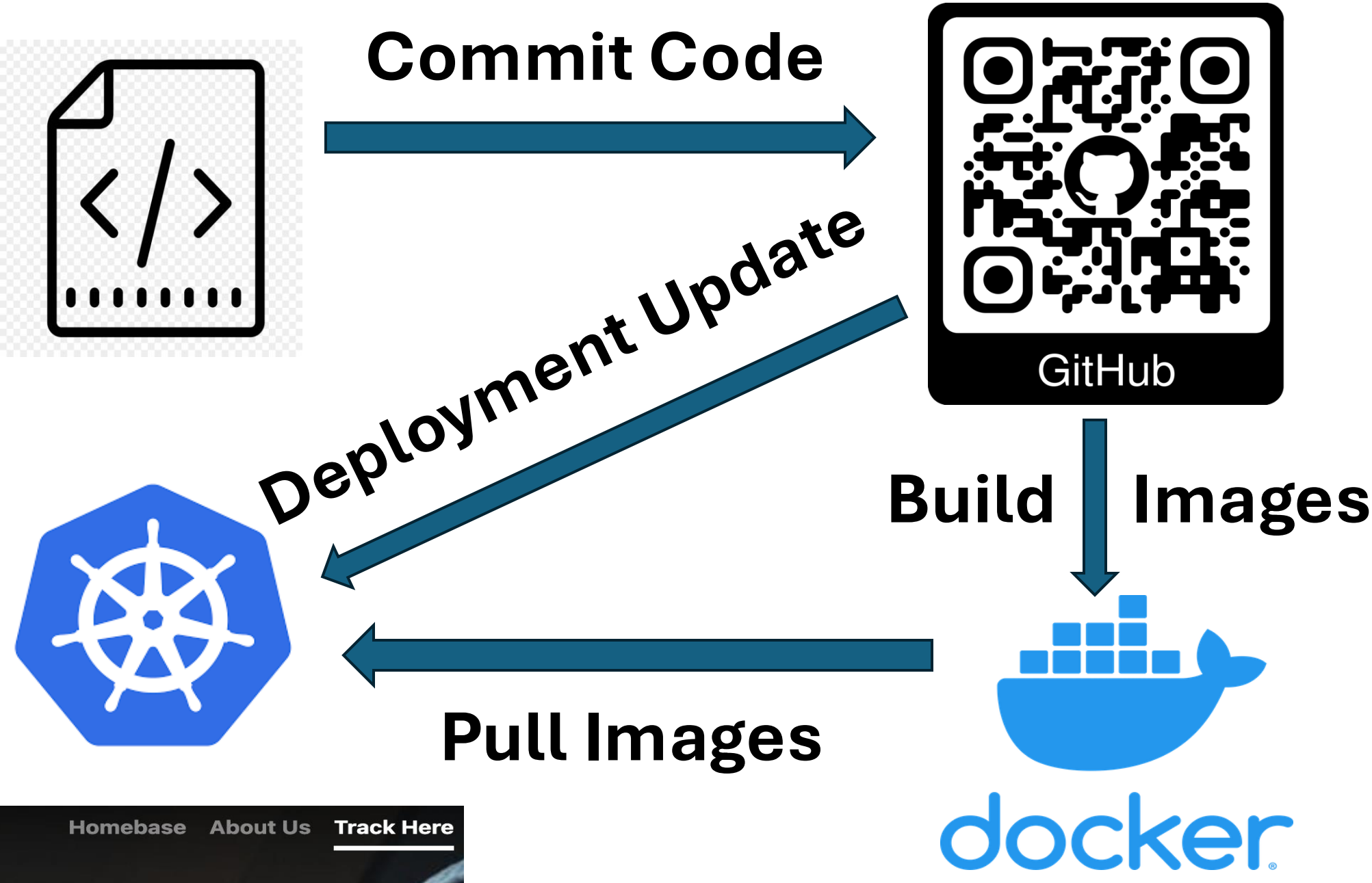
Web UI

The website features three sections:

- **Homebase:** Create or Login to account
- **About Us:** Team member Descriptions
- **Track Here:** Auto search for player stats

- Uses Bootstrap, HTML, CSS, and JavaScript

Login Page

Stats Page

Concept Architecture

1. The backend worker will communicate with the front end via GET requests.
2. Then backend will communicate with the Rainbow Six API to get user info.
3. POST requests will be sent to the Machine Learning service.
4. ML service will then query the data into the database, with the primary key being UserID.
5. Each UserID will have corresponding prediction values made with the user data from the API.

StatStrikeforce System

Web UI (Bootstrap) — User Input

Database (SQLite)

Get Request — Query Data — Insert/Update Data

API (ex. Riot, R6) — Get Request / API Response

Backend Worker (Python: Flask) — Post Request — Machine Learning Service (RFR)

Database Design

```
schema.sql

Users > ryansayre > Downloads > schema.sql
1   CREATE TABLE user
2   (
3       id              INTEGER PRIMARY KEY AUTOINCREMENT,
4       username        TEXT UNIQUE NOT NULL,
5       password_hash   TEXT        NOT NULL,
6       r6_user_id      TEXT,
7       created_at      DATETIME DEFAULT CURRENT_TIMESTAMP
8   );
9
10  CREATE TABLE user_stats
11  (
12      id              INTEGER PRIMARY KEY AUTOINCREMENT,
13      user_id         TEXT UNIQUE NOT NULL,
14      mse_attack REAL,
15      mse_defend REAL,
16      FOREIGN KEY (user_id) REFERENCES user (id)
17  );
18
```

- For our database design, we chose SQLite due to its space-saving capabilities and its compatibility with our backend through Python libraries, enabling efficient data storage and retrieval.
  - User Stats Table: Linked to the user via '**user_id**', this table tracks game statistics. It uses the '**user_id**' to connect user profiles performance from Rainbow Six Siege to their metrics in '**mse_attack**' and '**mse_defend**'
  - User Table: This table is for our login system. It stores the '**id**' as the primary key along with '**username**', '**password_hash**' for secure authentication, and the '**r6_user_id**'. The '**id**' is unique to every user and auto-increments with new entries.

Cloud Deployment

- Source code written and tested locally, before pushing to our project's GitHub repository.
- GitHub will then trigger the build process to have Docker push updated images.
- Kubernetes will pull images from DockerHub and run them in pods.
- GitHub will also push a deployment update to Kubernetes directly.
- Once the application is deployed, we can test functionality, monitor in dashboard, and plan for future builds.

CPU Usage

Memory Usage

Deployments

Name	Images	Labels	Pods	Created ↑	
webui	rsayre14/csc468:webui	role: app	2 / 2	46 minutes ago	
machinelearning	rsayre14/csc468:machinelearning	role: app	2 / 2	46 minutes ago	
backend	rsayre14/csc468:backend	role: app	2 / 2	46 minutes ago	