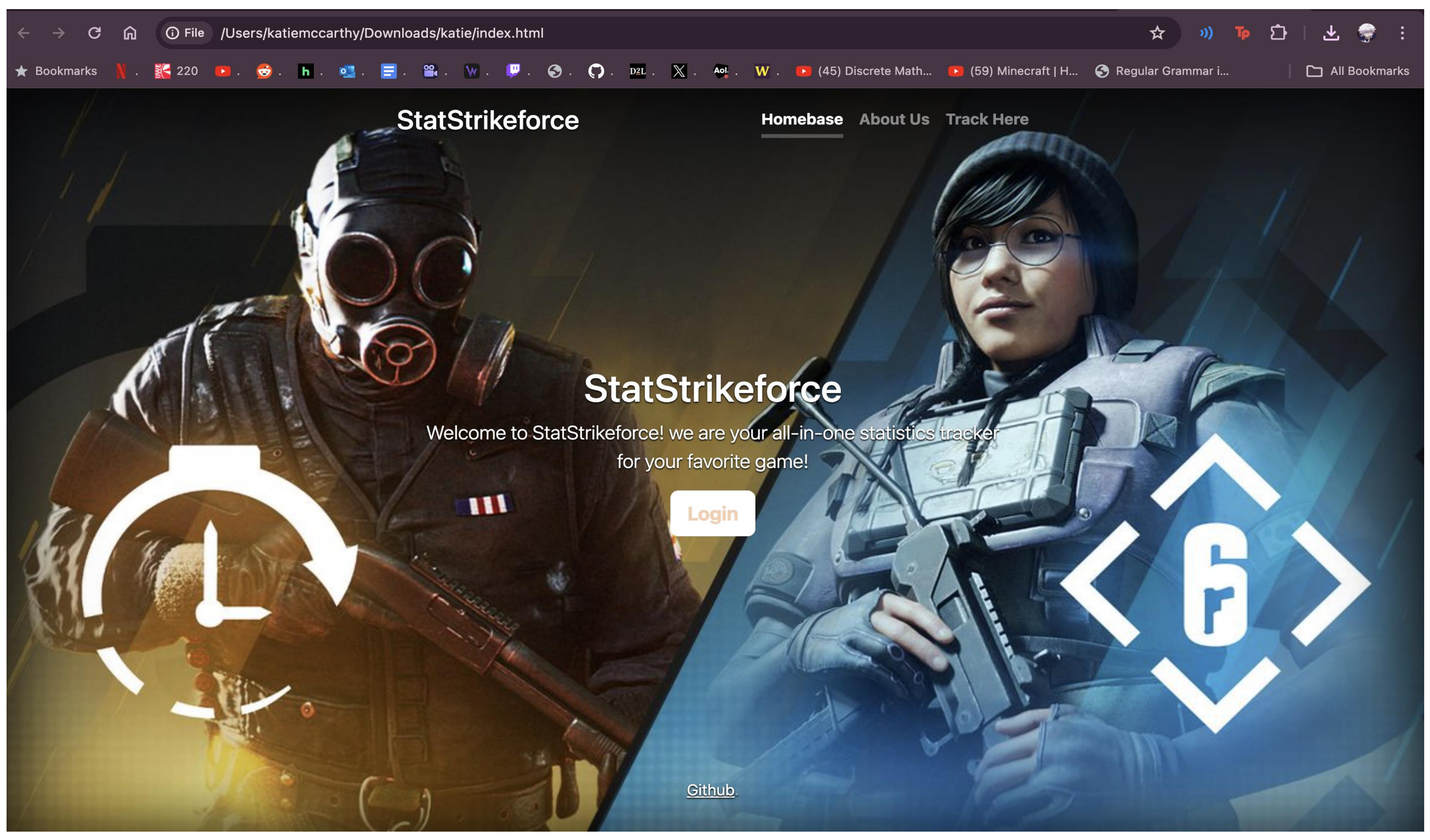
STATSTRIKEFORCE

Katherine McCarthy, Maxwell Mendenhall Ryan Sayre, Tobyn Sitar

GitHub

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.

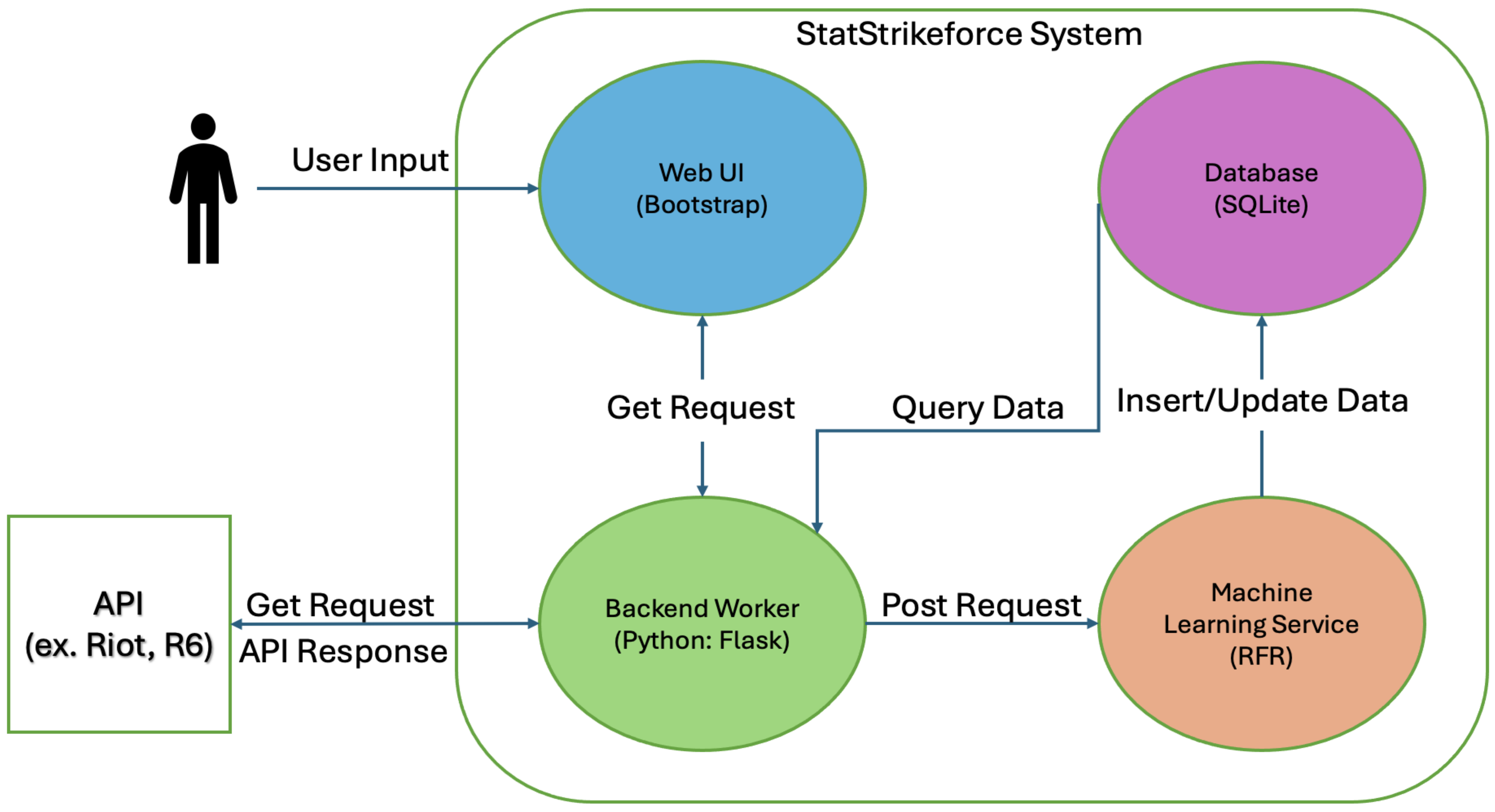

/Users/katiemccarthy/Downloads/katie/index.html
StatStrikeforce
Homebase
About Us
Track Here
StatStrikeforce
Welcome to StatStrikeforce! we are your all-in-one statistics tracker for your favorite game!
Login
Github

Web UI

For the Web UI, we plan on creating a homepage for users to login with their account information for their R6 account, leading them to a personalized page with all their statistics. The website features three sections, a Homepage, About Us page, and a Track Here page. We used Bootstrap and its libraries to form our vision for the user interface. The website features a sleek, aesthetic design to house all its capabilities.

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.



UserID		Predictions
(P) UserID: varchar	Kills: int	P(Win): float
platform: varchar	Deaths: int	P(K/D): float
Playerlevel: int	Assists: int	P(ΔRP): int
Totaltime(hrs): int	K/D: float	Timestamp: int
Wins: int	Headshot%: float	
Losses: int	Rank: varchar	
Win%: float	RP: int	
Matches: int	Tophero: varchar	

Database Design

- For our database design, we opted for SQLite, due to its space-saving abilities and compatibility with our backend through its own Python libraries to store and reference data.
  - UserID is used to store the game statistics, login credentials and basic profile information. Each user choosing a 'UserID' to be used as the primary key.
  - Predictions table stores predictions generated by the machine learning service. New predictions will be generated when data is deemed stale through Timestamp.

Cloud Deployment

- Our CI/CD pipeline begins with 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 said images from DockerHub and run them in pods within our experiment nodes. GitHub will also push a deployment update to Kubernetes directly.
- Once the application is deployed, we can test functionality and plan for future builds.