



# COURTCONNECT

The gamechanger for tennis players.

# Developer Team: 12-03

**Tom Riley**

**Front-end**

**Trish Le**

**Front-end**

**Rodrigo Molerio**

**Front-end**

**Caleb Lehman**

**Back-end**

**Mitch Kubina**

**Back-end**

**Jake Carroll**

**Back-end**



# Project Description:

**Vision Statement:** For tennis players who want to find a court and competitors to play against. CourtConnect is a booking system and social network that allows easy and efficient access to play. Unlike other court reservation systems, our product connects the tennis community.

**Description:** Courtconnect allows users to view and make reservations at local tennis courts and their respective parks in the area, making match control easier than ever.

Each user creates their own unique profile, with different descriptors such as skill level and location to allow connectivity with other players. They can create reservations with the option of allowing others to join, as well as being able to join other's reservations to find new players in the area to play with.

# Tools Used:

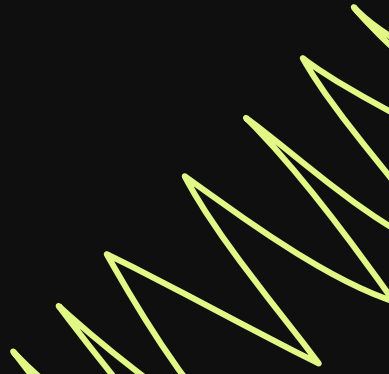
Ranked out of 5 stars

<u>Version Control: GitHub Repository</u>	3: Allowed efficient sharing of code, had a few problems with push conflicts
<u>GitHub Project Board</u>	5: Very easy to use with nice display of all our assigned tasks
<u>Figma</u>	5: Wireframe tool with easy editing tools, allowed great design
<u>DataBase: PostgreSQL</u>	4.5: Allowed easy and effective way to store and access data
<u>Visual Code Studio</u>	5: Very user friendly IDE
<u>NodeJS</u>	4: Easy to use application server
<u>Deployment: Microsoft Azure</u>	4: Quick deployment and fast way to create virtual machines
<u>HTML and EJS</u>	4: Allowed easy page design and development
<u>Bootstrap</u>	2.376: A little bit hard to create unique designs

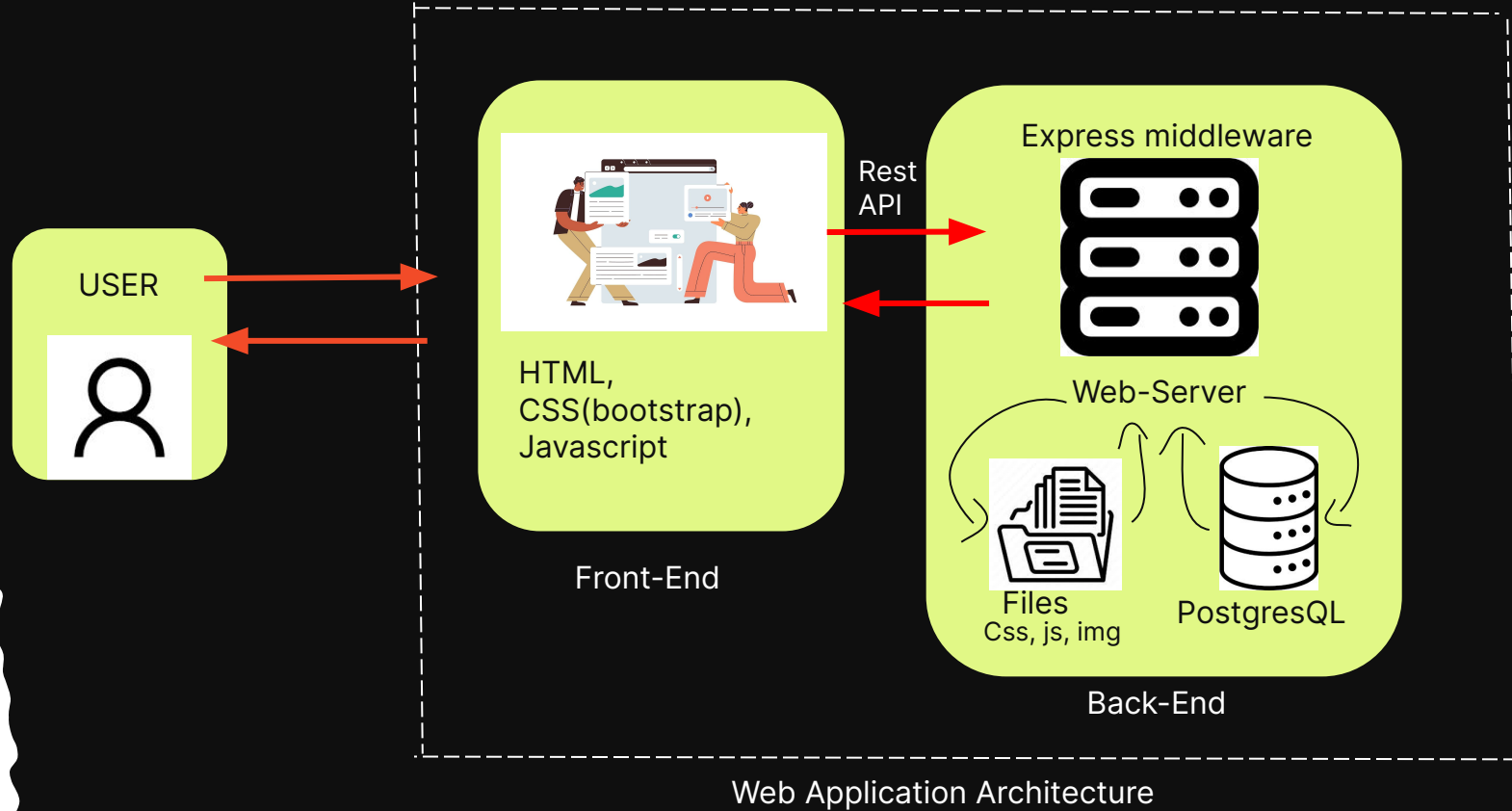


# Architecture

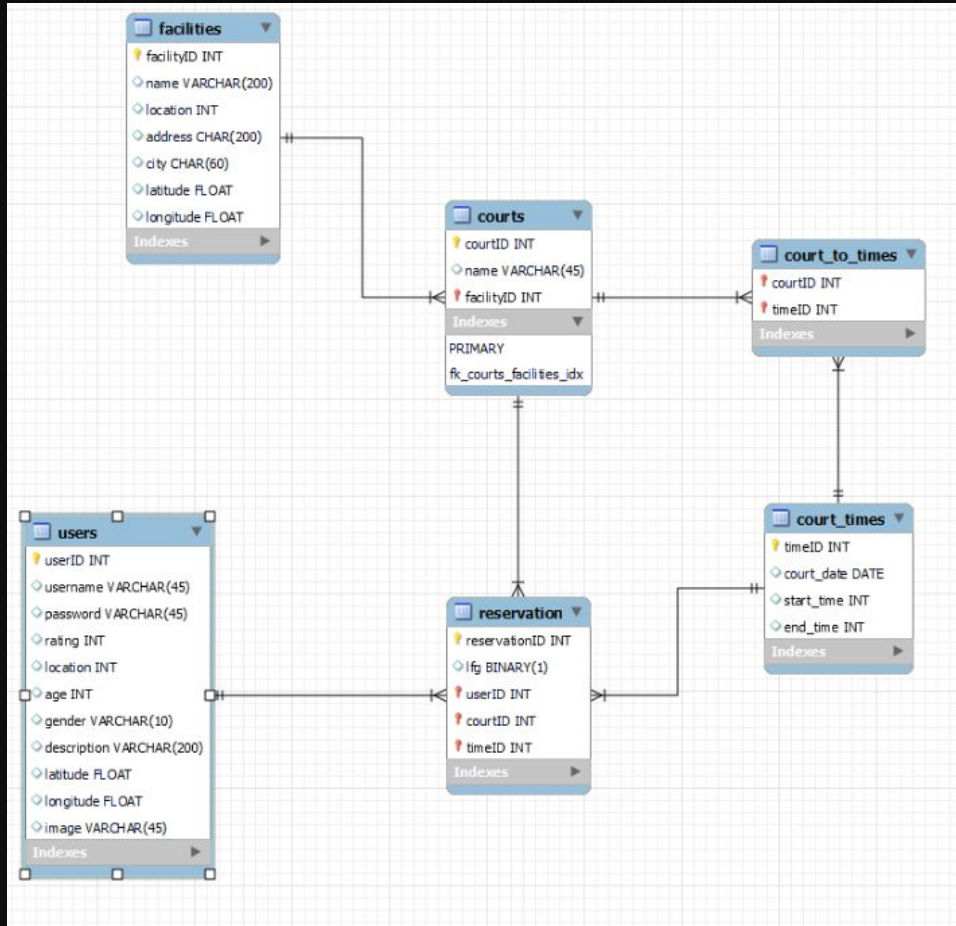
Here's a few diagrams to  
depict the structure



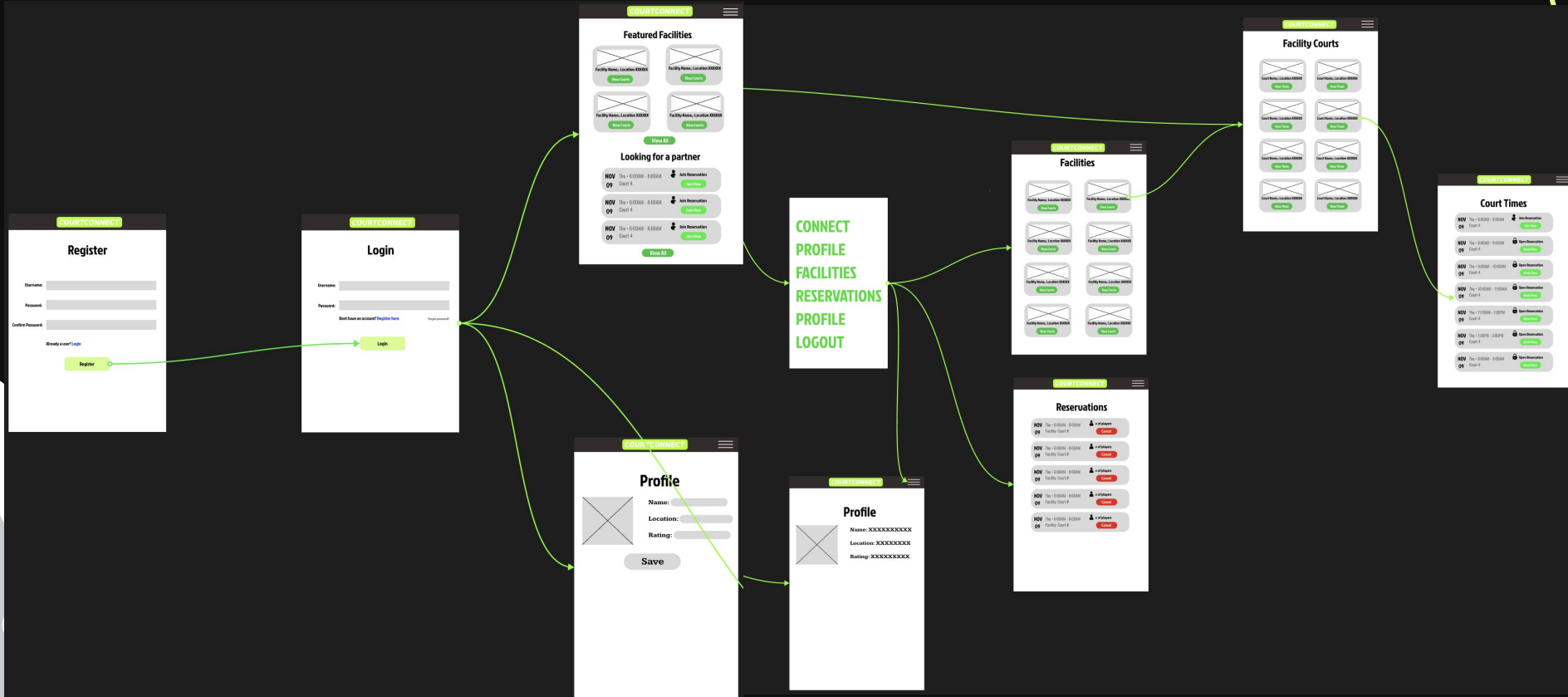
# Architecture Diagram



# Database Structure



# Data Flow:





# CHALLENGES FACED:

## Github Collaboration

There were several instances where there were merge conflicts, bugs with pushing code onto the repo at the beginning, which we ended up solving and started using branches and pull requests

## Deadlines

Time crept up on us, and we found that we were crunched on time which resulted in a lot of cramming and rushing to finish our parts

## Communication

It was a little hard to keep track of everything at the beginning in terms of planning out. But, we made a discord and held team meetings to assign roles and documented

## Debugging

There were a lot of instances where after merging, people's codes would break the repo, but we all collaborated to fix everything



**Ready for the demo?**

**[LINK](#)**