- Repo is on github (duh)
- Models API on Render
    - Should only have 1-2 people on this
    - Cloud platform for a lot of things but we will use it for the API
    - Known to be easy to use
    - Free to use in small scale
        - Can be helpful for testing
    -
- Database is on Digital Ocean
    - Cloud infrastructure provider for our case it is used for the database
    - Data scientists on DO
- Prisma
    - Used for the schema
    - It is an object relational mapper (ORM)
        - Connects OOP with relational databases
        - Helps to interact with the data in a more OO structure rather than a query structure

PRISMA SCHEMA:

```
generator client {
  provider      = "prisma-client-js"
  binaryTargets = ["native", "rhel-openssl-1.0.x"]
}

datasource db {
  provider = "postgresql"
  url      = env("DATABASE_URL")
}

model Flex {
  id          Int       @id @default(autoincrement())
  name        String    @unique
  description String
  rarity      Float?
  uniqueId    String    @unique
  Recipe      Recipe[]
  UserFlex    UserFlex[]
}

model Recipe {
  id              Int      @id @default(autoincrement())
  flexUniqueId    String
  achievementKeys String[]
  rarity          Float?
  disabled        Boolean? @default(false)
  Flex            Flex     @relation(fields: [flexUniqueId], references: [uniqueId])
}


model User {
  id             Int           @id @default(autoincrement())
  state          String?
  totalGames     Int?
  processedGames Int?
  gameId         String        @unique
  source         String
  games          Json?
  notes          String?
  processedDate  DateTime?
  info           Json?
  clusterInfo    Json?
  country_code   String?
  persona_name   String?
  real_name      String?
  email          String?
  userCampaigns  userCampaign[]
}

model UserStat {
  id             Int     @id @default(autoincrement())
  gameId         String  @unique
  state          String?
  mostPlayedGame Json?
  genres         Json?
  notes          String?
}

model UserFlex {
  id             Int      @id @default(autoincrement())
  flexUniqueId   String
  recipe         Json?
  achievedDate   DateTime
  gameId         String
  recipeAchiKeys String[]
  Flex           Flex     @relation(fields: [flexUniqueId], references: [uniqueId])

  @@unique([flexUniqueId, gameId])
```

```
}

model Game {
  id              Int              @id @default(autoincrement())
  appid           String           @unique
  name            String
  metadata        Json?
  platform        String
  achievements    Json?
  category        String[]
  genre           String[]
  GameAchievement GameAchievement[]
}

model GameAchievement {
  id              Int              @id @default(autoincrement())
  apiname         String?
  unique_key      String           @unique
  name            String
  description     String?
  percent         Float?           @default(0.0)
  appid           String
  platform        String
  recipeId        Int?
  Game            Game             @relation(fields: [appid], references: [appid])
  UserAchievement UserAchievement[]
}

model UserAchievement {
  id              Int            @id @default(autoincrement())
  gameId          String
  unlocktime      Int
  playTimeInGame  Int
  unique_key      String
  GameAchievement GameAchievement @relation(fields: [unique_key], references:
[unique_key])

  @@unique([gameId, unique_key])
}

model Employer {
  id       Int       @id @default(autoincrement())
  name     String
  email    String
  createdAt DateTime   @default(now())
  updatedAt DateTime   @updatedAt
  uuid      String     @unique
  campaigns Campaign[]
}

model Campaign {
  id            Int           @id @default(autoincrement())
  name          String
  employerUuid  String
  Employer      Employer       @relation(fields: [employerUuid], references: [uuid])
  userCampaigns userCampaign[]
}

model userCampaign {
  id         Int      @id @default(autoincrement())
  userId     String
  campaignId Int
  Campaign   Campaign @relation(fields: [campaignId], references: [id])
  User       User     @relation(fields: [userId], references: [gameId])

  @@unique([userId, campaignId])
}
```