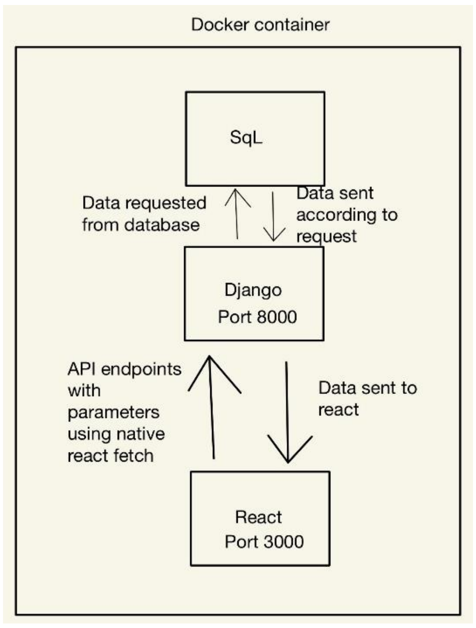
Building Systems for People Final Report

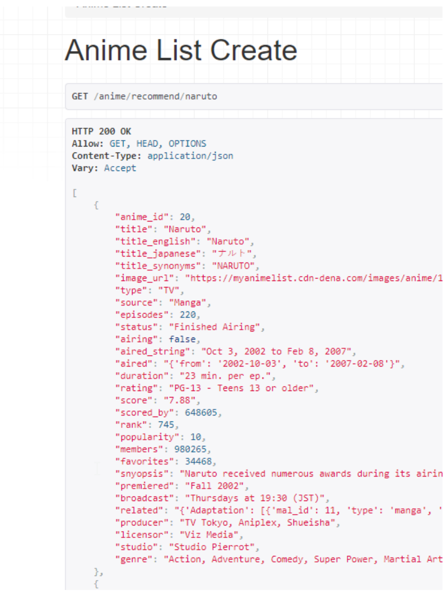
The full stack web application I chose to build uses a React + Django with the rest framework installed. The website's purpose is to allow users to search through a large database provided by the website for shows they like and follow. The reason for choosing the Django + React + SQL framework is because as opposed to other full stack frameworks like MERN is due to a variety of reasons.

Django is better suited for my website as opposed to the most popular web-stack MERN because it comes with many in-built functions such as user authentication, testing, and database management. Combining Django with the rest framework also allows you to send high quality JSON responses to your frontend via end points set up by default Django which returns serialised data. Django also uses SQL as its native database as opposed to MongoDB, and since I'm more used to using SQL as opposed to a cloud database which would cost a lot more since I store 15000 entries as opposed to being a lot cheaper to host on Django. As a result of not having to constantly query Mongo dB for data Django can also operate a lot faster as opposed to the cloud connection since it's all in one file structure. The reason I chose react as my front end is because it comes with many quality-of-life features and libraries such as Material-Ui which can help speed up dev-OPS a lot more which makes the program develop faster and get tested faster. Material UI also comes with many template components that you can use to speed up front end development which leads to any development company being able to develop a lot faster.

The way my software will communicate with each other is very simple, there will be a database hosted on my Django database which will be run in separate port from my React frontend. To fetch data, whenever a user searches a show, it will send a GET request to my backend API end point, which will then respond with all the data in an Array format which the frontend can then decrypt. This will be done using the default react fetch command. This fetch is more than enough as opposed to using a library like axios since it will do everything my program needs to do which is fetch data. This model is therefore will have a different frontend and backend running at the same time which stay in contact with each other through API requests. The whole project will be hosted on Github where I will implement continuous integration which will test my code as I commit to my repository, this will allow a high level of code quality as I will use PIP8 to test my code quality and will use coverage to ensure all of my code is properly tested.

Data management


Docker container
SqL
Data requested from database
Data sent according to request
Django Port 8000
API endpoints with parameters using native react fetch
Data sent to react
React Port 3000


Anime List Create

```
GET /anime/recommend/naruto
```

```
HTTP 200 OK
Allow: GET, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept

[
    {
        "anime_id": 20,
        "title": "Naruto",
        "title_english": "Naruto",
        "title_japanese": "ナルト",
        "title_synonyms": "NARUTO",
        "image_url": "https://myanimelist.cdn-dena.com/images/anime/1
        "type": "TV",
        "source": "Manga",
        "episodes": 220,
        "status": "Finished Airing",
        "airing": false,
        "aired_string": "Oct 3, 2002 to Feb 8, 2007",
        "aired": "{'from': '2002-10-03', 'to': '2007-02-08'}",
        "duration": "23 min. per ep.",
        "rating": "PG-13 - Teens 13 or older",
        "score": "7.88",
        "scored_by": 648605,
        "rank": 745,
        "popularity": 10,
        "members": 980265,
        "favorites": 34468,
        "snyopsis": "Naruto received numerous awards during its airin
        "premiered": "Fall 2002",
        "broadcast": "Thursdays at 19:30 (JST)",
        "related": "{'Adaptation': [{'mal_id': 11, 'type': 'manga', '
        "producer": "TV Tokyo, Aniplex, Shueisha",
        "licensor": "Viz Media",
        "studio": "Studio Pierrot",
        "genre": "Action, Adventure, Comedy, Super Power, Martial Art
    },
    {
```

APIS,

STORAGE,

AUTHENTICATION,

CONTAINERS, HOW YOU NEED BOTH THE SERVER TO RUN AND THE FRONTEND

CONTINIOUS INTEGRATION

MAINTENANCE AND OBSERABILITY

SECURITY