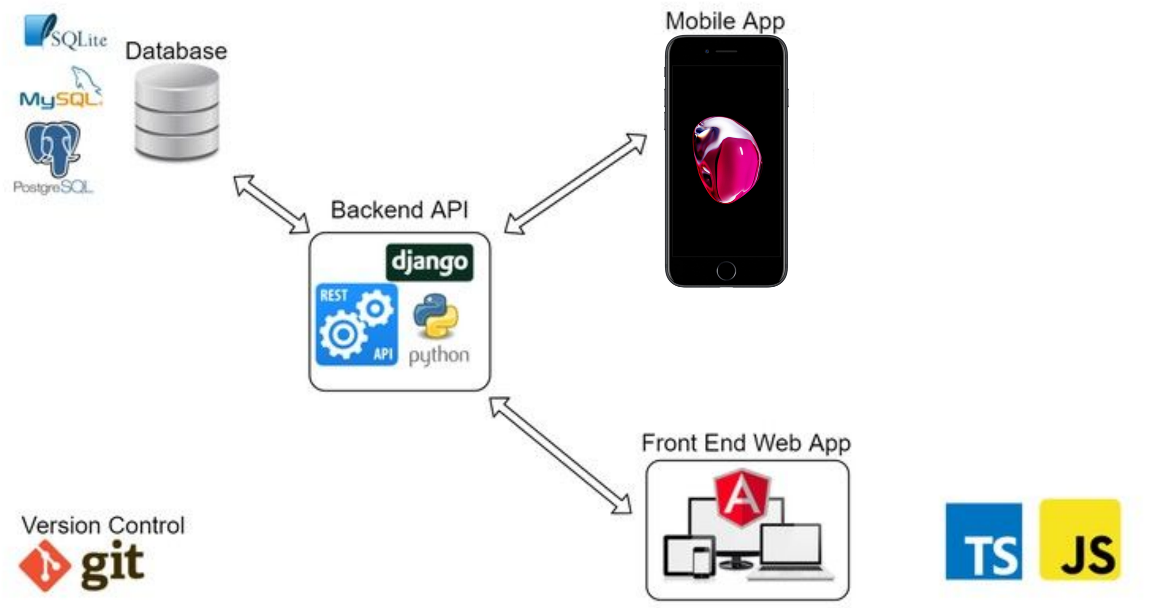
Lesson 5

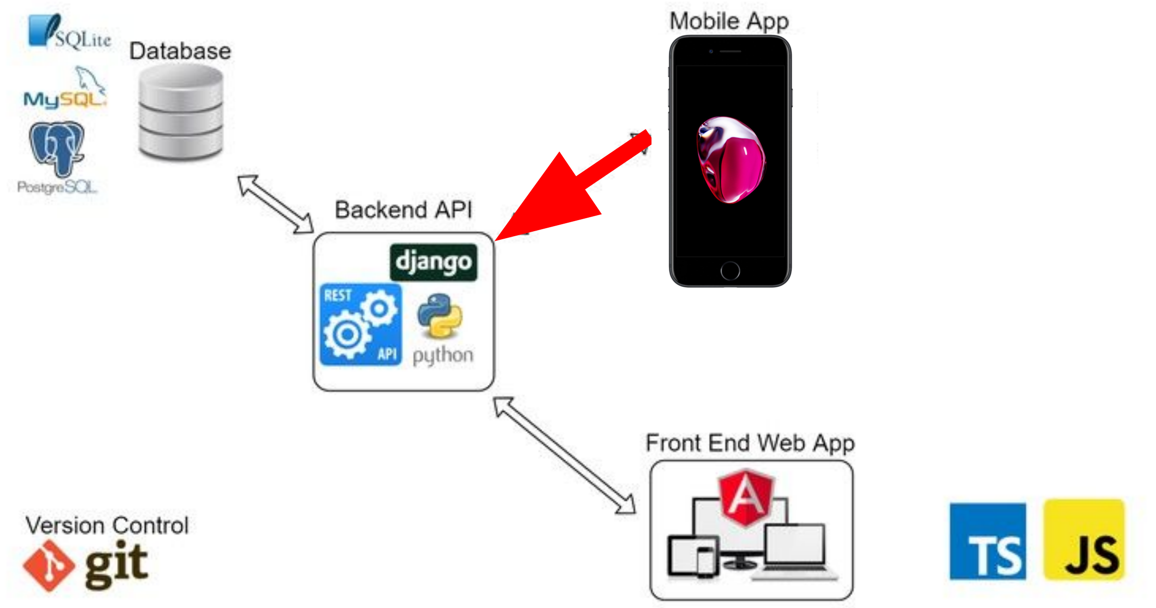
Parts of the stack



Request from start to finish

- Let’s say someone is trying to see their Instagram feed on their iPhone
- How would the whole process go?

Parts of the stack

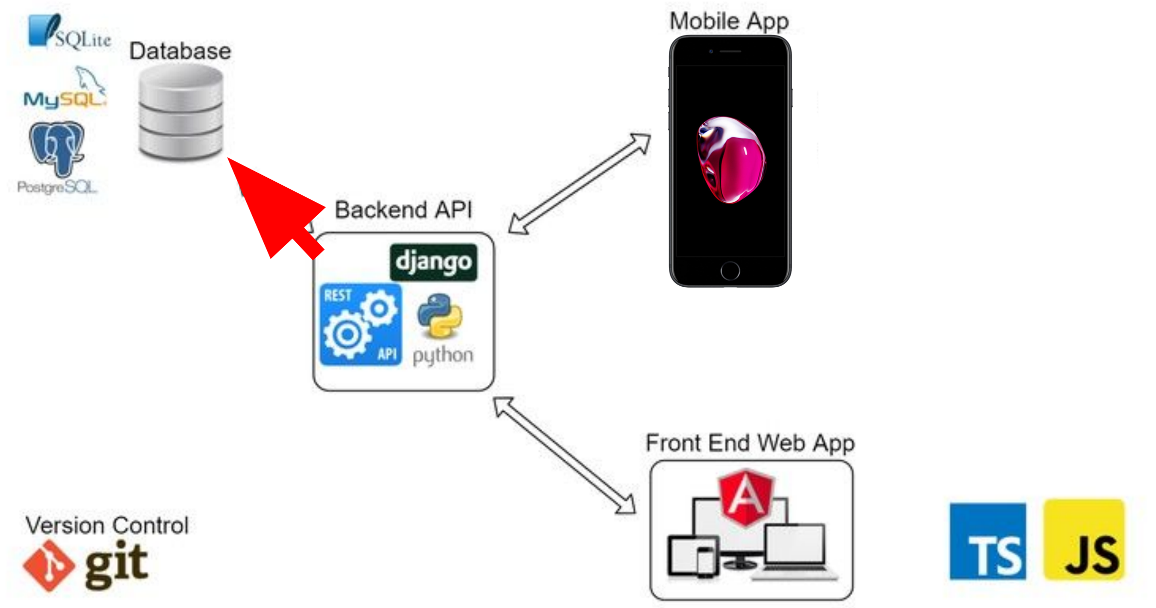


Client sends a request

HTTP **GET** request

Route would probably be something like **/feed**

Parts of the stack

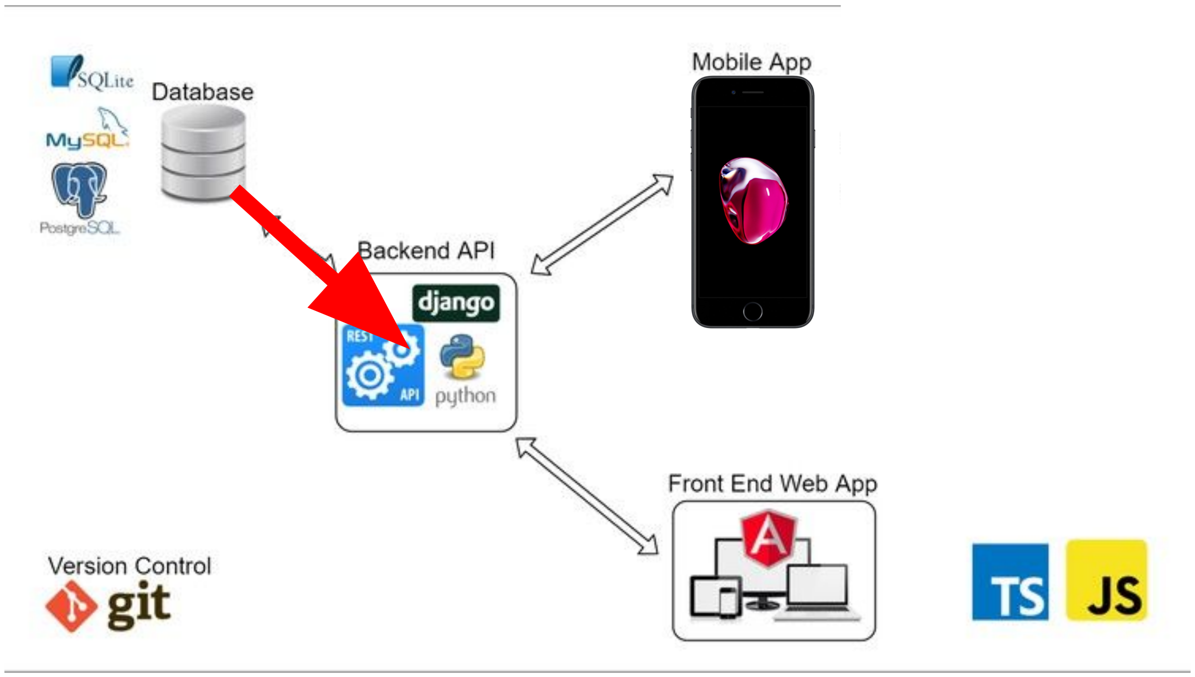


Server contacts the database

Issues a **query** to the database

- Asks to **find** the images that belong on the user's news feed
- Tells the database the criteria to determine what goes on the user's feed

Parts of the stack



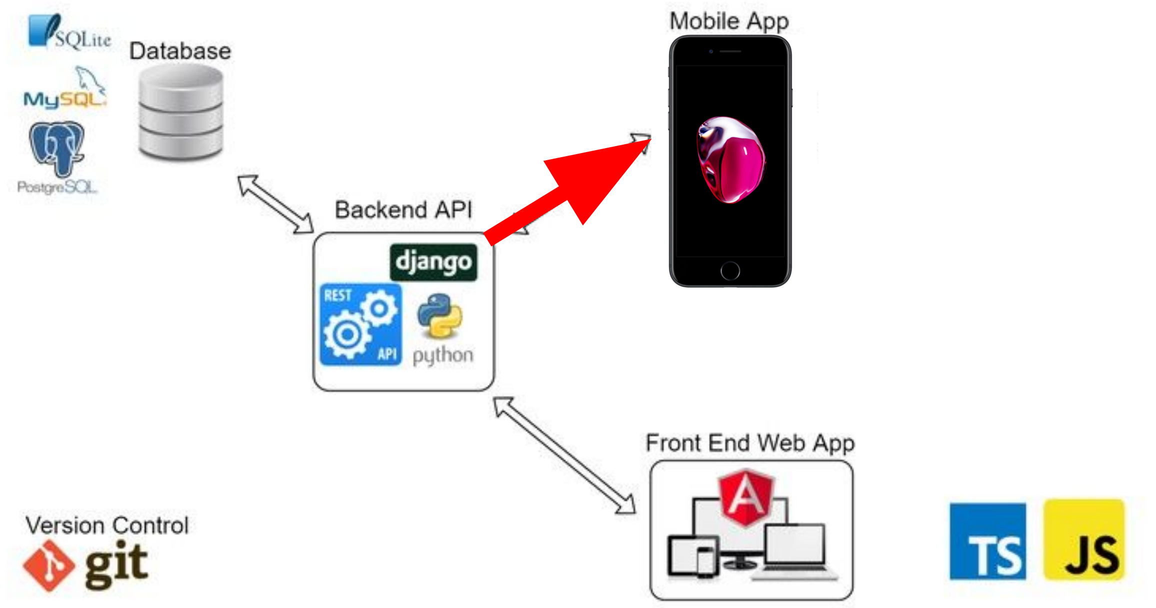
Database responds to server

Sends all the **results** of the **query** the server issued

They will be in the form of a list

The **results** depend on how the database is structured

Parts of the stack



Response to client

It would be a **list** of **objects**

What would each object look like?

Response to client

It would be a **list** of **objects**

What would each object look like?

```
{
        “username”: <person who posted it>,
        “Image”: <link to image>,
        “likes”: <number of likes>
        ...
}
```

Last time

Started the **database** using the **mongod** command (or some variation of it)

Then ran **mongo**

What do each of these do? How do they work with our Flask server?

Mongod - start a database

- Database is **ready** to **accept connections**
- In other words, the **database server** is running
    - You can **insert** stuff into it
    - You can **remove** stuff from it
    - You can **retrieve** stuff from it
    - You can **modify** stuff in it

Using the database

How do we send those retrieve/insert/remove/modify commands?

A few options

- Through the MongoDB shell (**mongo** starts the shell so you communicate with the database)
- Through some software like MongoDB Compass (free download)
- Through Python, Java, or any other major language

Pymongo

Python library containing tools for working with MongoDB

Uses:

- Inserting documents, querying for documents

Installation:

```
$ python -m pip install pymongo
```

Inserting a single document

- To insert a document into a collection, we can use the insert_one() method:

SYNTAX:

```
[collection].insert_one({your document})
```

Getting a Single Document

- GET a single document from the database using find_one

SYNTAX:

[collection].find_one({your document})

Designing our collections

Restaurants Collection. Each document looks like:

```
{
        “name”: “Olive Garden”,
        “address”: {
                “street”: “50 Main St.”,
                “state”: “NJ”,
                “zip”: “08901”
        },
        “items”: [
                {
                        “name”: “pasta”,
                        “price”: 12
                },
                {
                        “name”: “salad”,
                        “price”: 9
                }
        ]
}
```

Live Coding

- Code POST and GET for
    - /restaurants