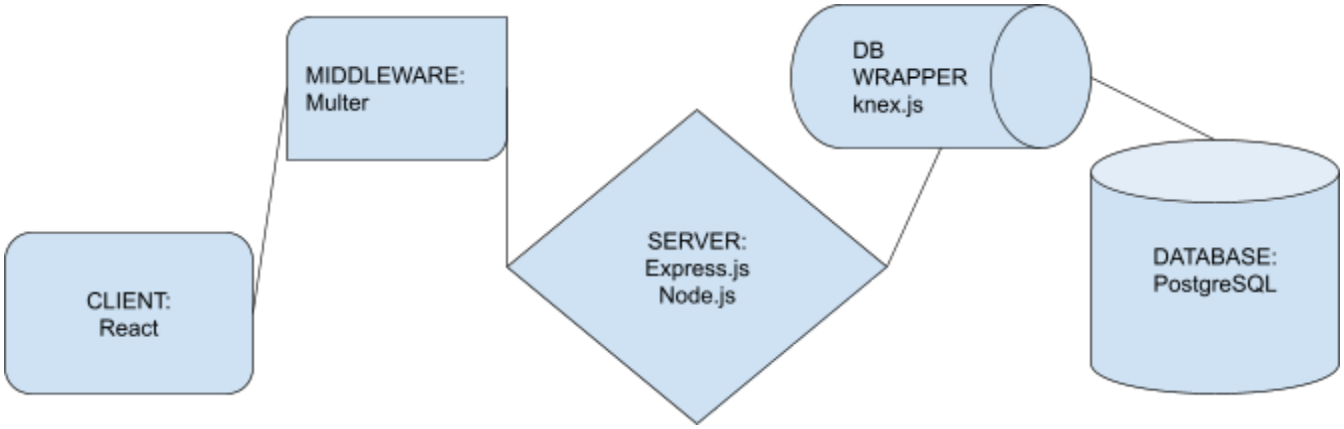
Leslie Ngo
Shopify Winter Backend Internship
System Design Overview



Front end:

- React

Back end:

- Node.js
- Express.js
    - Multer
    - Axios
    - fs
- PostgreSQL**
    - Knex.js

Testing

- Supertest
- Jest

Instructions

1. First, populate an instance of Postgres with the included .sql files in the root directory.
    a. Go to the db folder, and inside index.js, change the Knex connection object credentials if needed.
2. NPM install.
3. NPM start to run the app.

Client

- Features
  - Single and multiple file uploads.
  - Persistent data powered by Postgres.
  - Image search via join query through Knex.js.
  - Keyword population via Knex.js.

Express Server Endpoints

- GET
  - /get
    - Responds with urls to default photos, and any new photos added.
  - /keywords/:image
    - Responds with keywords associated with the image title provided.
  - /search/:keyword
    - Responds with urls to photos associated with the provided keyword.

- POST
  - /upload
    - If provided files, this endpoint will store the location and metadata of any images uploaded onto Postgres, and will copy the file to disk.
    - These files can be found in the uploads folder located in the root directory, and are accessed by the client via URL.
  - /addKeyword/:image/:word
    - Providing the specific image title and a string to this endpoint will allow the collection of keywords to be updated.

Schema: