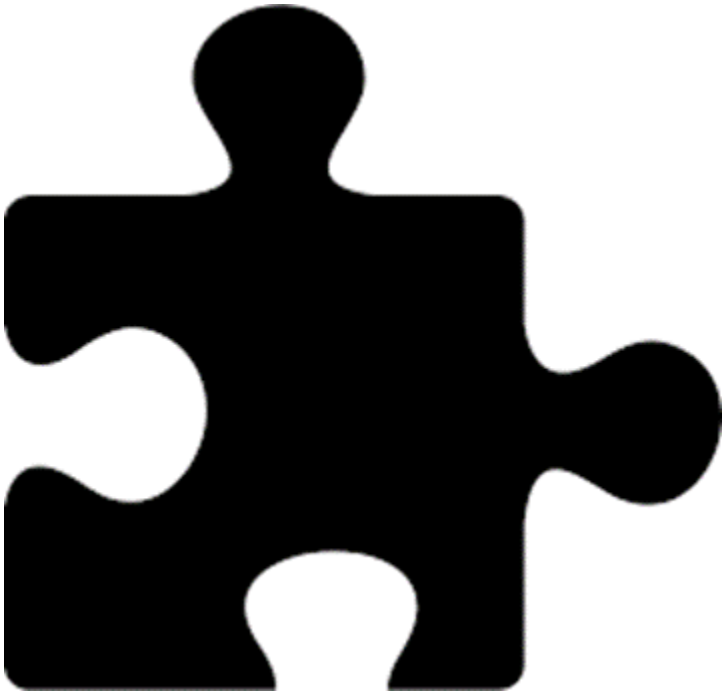
System Design



PriceHawk

February 5, 2023

Table of Contents

System Architecture

High Level Description

The goal of the project is to build a chrome extension.

The Chrome extension's user interface would be built using React, which is a JavaScript library for building user interfaces. The React frontend is responsible for handling the user interactions with the extension and making API requests to the Django backend. The team decided to use TypeScript which builds on JavaScript to aid with the development. This makes up our client tier.

As for our business layer, it consists of the rest API, which is built using Django, which is a Python web framework. The API is responsible for serving client requests from the frontend. It also acts as the middleman connecting to the data tier as many requests to the API will require querying and storing data to the database. Additionally in the business layer, we use Sendgrid to handle sending emails and perform validation on our behalf.

PostgreSQL is used as the main database management system in the backend. PostgreSQL fits perfectly with Django as it provides SQL querying and is supported by Django's ORM (Object Relational Mapping). ORM allows the developers to define programmatic relationships between Django models and the data in the database. Docker is used to run the PostgreSQL image. Docker is a good choice as it provides fast deployment and persistence via volumes. This makes up our data tier.

For error handling in the business layer, we use Django to return validation, permission, and server errors.

CRC Models

User

- Responsibilities
    - Handles the standard register + signin flow.
    - Handles password resets.
    - Handles guest user migrations.

- Collaborators
    - Emailer
    - Item
    - Built in Django user model
    - REST framework's auth controller.

Item

- Responsibilities
    - Handles CRUD operations for items which the user wants to track.
    - Calls the notifier when the price is updated.

- Collaborators
    - Notifier
    - Recommender
    - User

Oauth

- Responsibilities
    - Handles the oauth register + signin flow.

- Collaborators
    - Built in Django user model.
    - REST framework's auth controller.

Scraper

- Responsibilities
    - Handles scraping a tracked item, and extracting the price from its URL.

- Collaborators
    - Item
    - Scheduler

Scheduler

- Responsibilities
    - Handles initializing the Scraper on some schedule.

- Collaborators
    - Scraper

Notifier

- Responsibilities
    - Handles notifying the user when a tracked item's price changes.

- Collaborators
    - Emailer
    - Item

Emailer

- Responsibilities
    - Handles emailing the user when one of their tracked items' price changes.
    - Only notifies if the user wants to be notified.

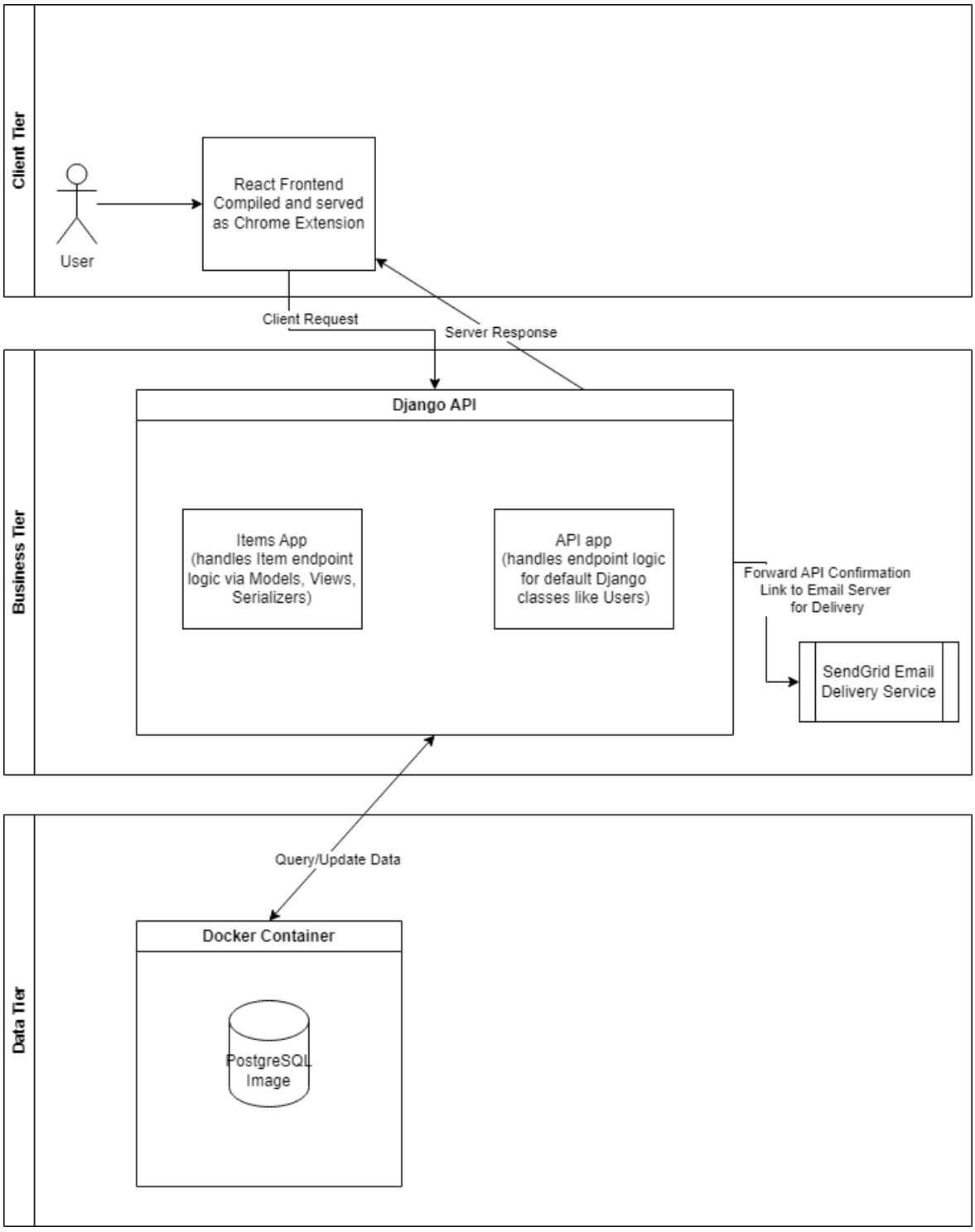
- Collaborators
    - Notifier
    - User

Recommender

- Responsibilities
    - Handles finding similar tracked items.

- Collaborators
    - Item

Architecture Diagram



Source for Three-Tier: https://mcs.utm.utoronto.ca/~rosenbl6/csc409/22f/lectures/architecture/