BITTIGER

BITTIGER

WeChat Bot House Finder

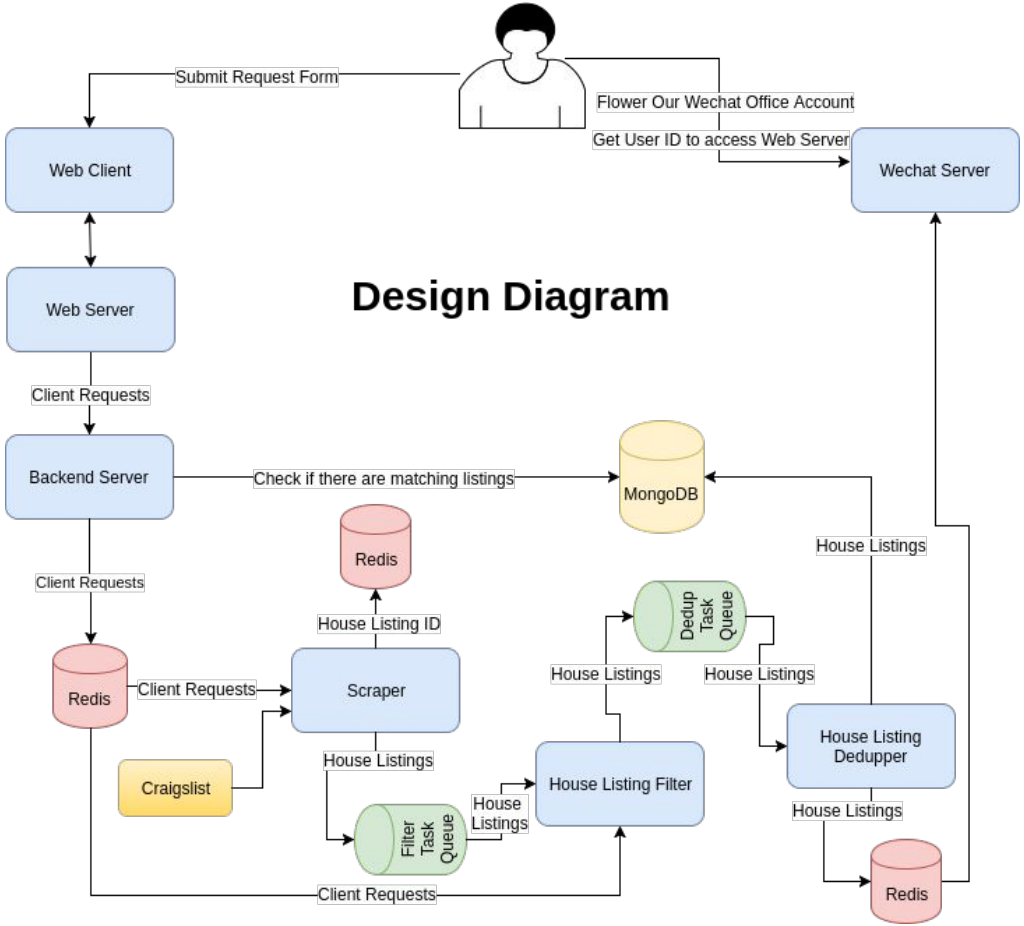
Linguan Yang

Outline

- Architecture
- Demo
- Conclusion

Architecture

- Front-End Web Server
- Back-End Server
- Data Pipeline
- WeChat Server



Architecture: Front-End Web Server

UI

- Home Page
- Request Form Page
- History Page
- Detail Page

API

- GET: /home/userId/:userId
- GET: /requestForm/userId/:userId
- POST: /request/userId/:userId
- GET: /history/userId/:userId
- GET: /requestDetail/userId/:uerId/requestId/:requestId
- DELET: /requestDetail/userId/:uerId/requestId/:requestId
- POST: /requestDetail/userId/:uerId/requestId/:requestId

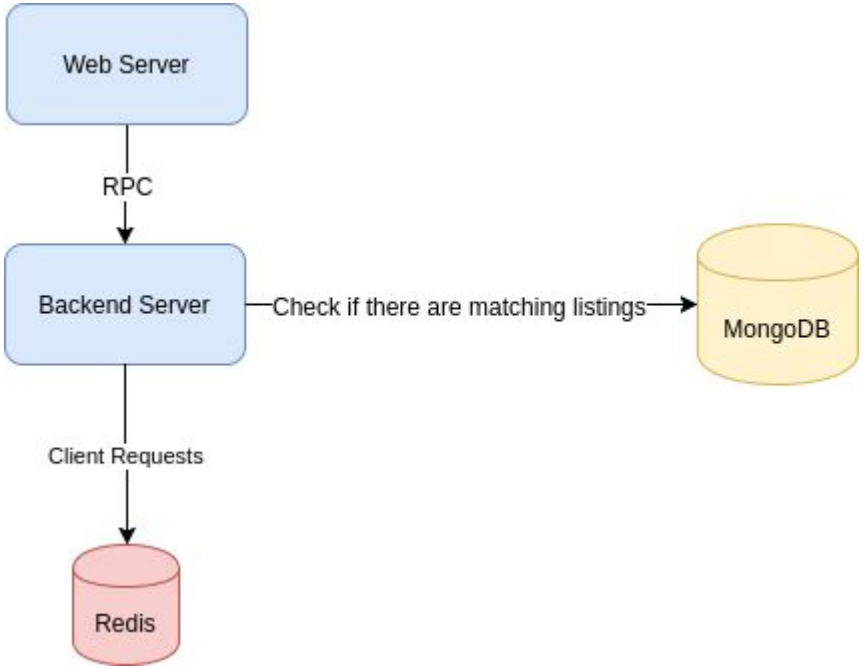
Architecture: Front-End Web Server

Technology

- React
- Node.js
- JavaScript

Architecture: Back-End Server

- Web Server communiates with Back-End server by RPC
- Store client request form into Redis
- Handler other requests from clients (request history, detail)



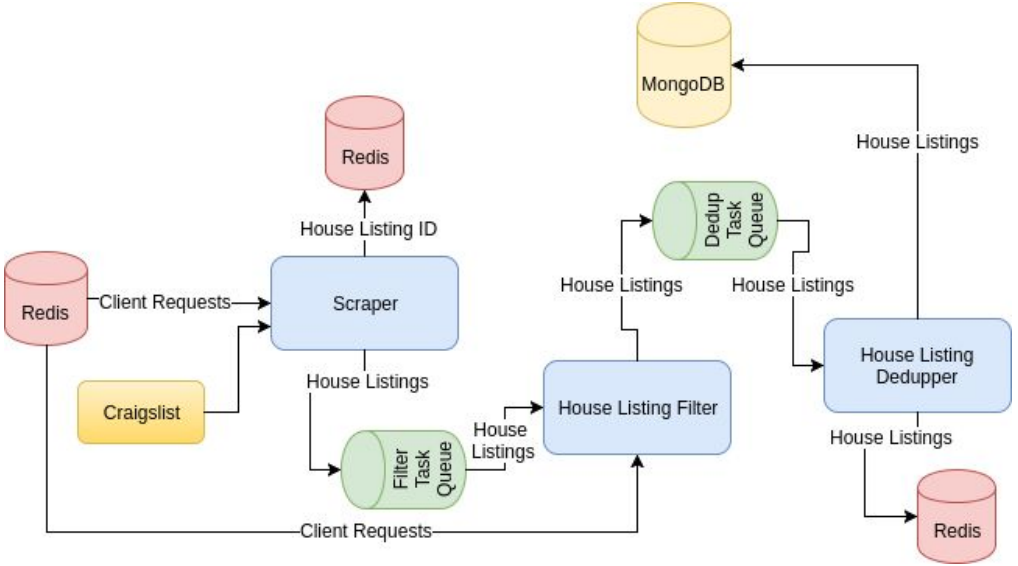
Architecture: Back-End Server

Technology

- RPC
- Python

Architecture: Data Pipeline

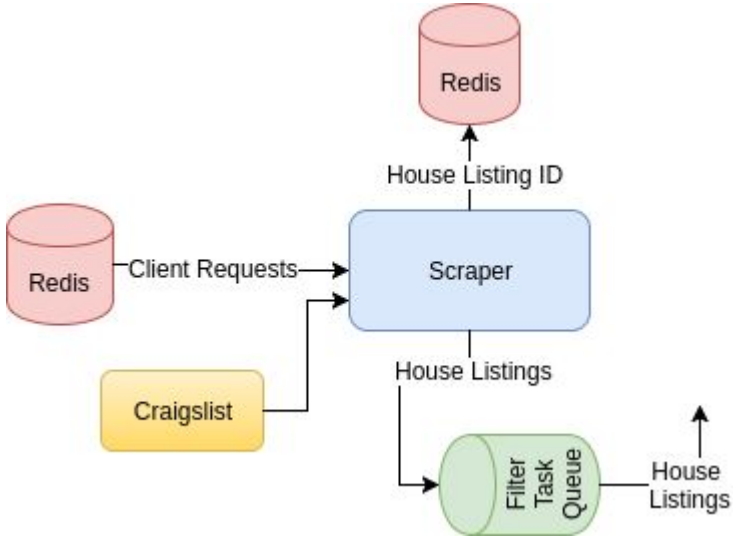
- Scraper
- Filter
- Deduper



Architecture: Data Pipeline

Scraper

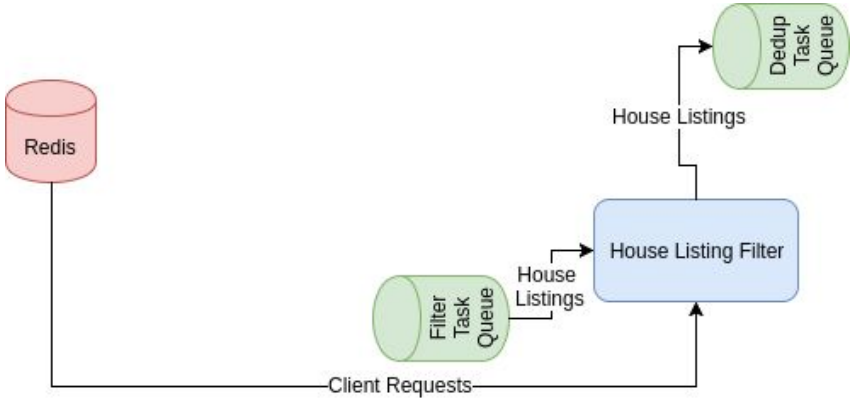
- Read client request from Redis
- Use python-craigslist package to fetch the house listing
- Filter by posting ID.
- Store the listing into Filter Task Queue



Architecture: Data Pipeline

Filter

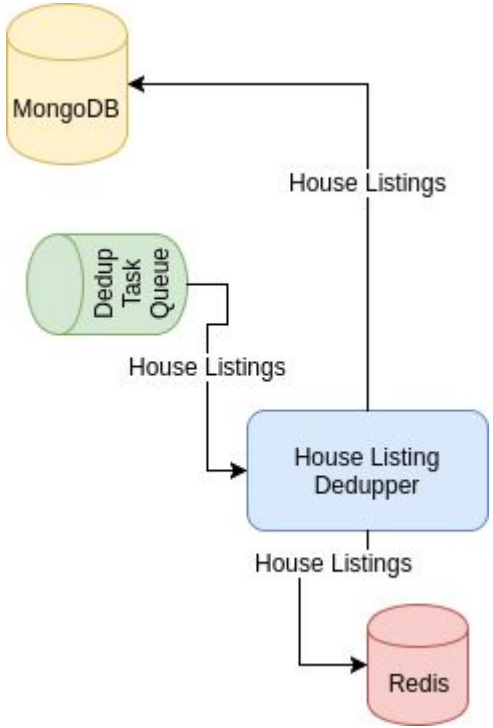
- Fetch house listing from Queue
- Fetch client's request from Redis based on client ID
- Filter listings by the time takes from the house to work place using Google Maps API
- Store the listings into Deduper Task Queue



Architecture: Data Pipeline

Deduper

- Fetch house listings from Queue
- Fetch image url using BeatifulSoup
- Filter listings based on the hash of image url
- Store the listings into MongoDB and Redis
- In Redis, the listings expire in 10 days



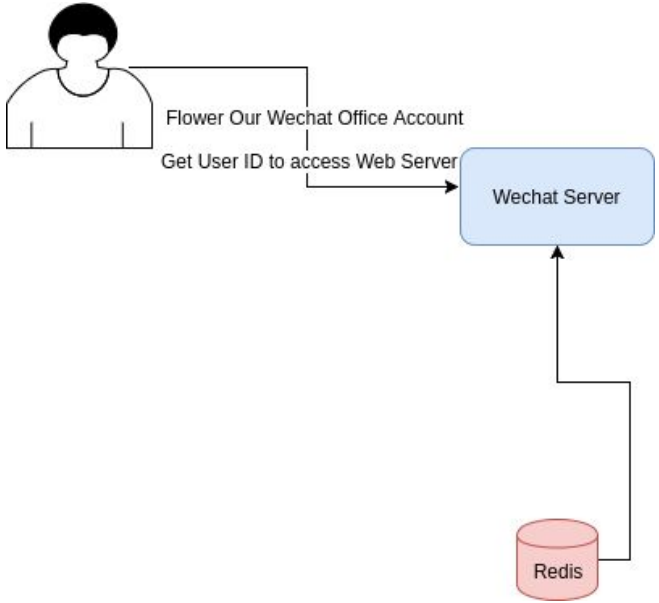
Architecture: Data Pipeline

Technology

- RabbitMQ
- BeatifulSoup
- Python

Architecture: WeChat Server

- Handler messages, subscribe and unsubscribe events
- Fetch new house listings for clients



Architecture: WeChat Server

Technology

- Flask
- Python

Architecture: Deployment

- This system is deployed on Vultr cloud
- Nginx
- Docker

Architecture: Future Work

- Avoid scrape redundant data
- Use distributed system to scale
- Use MongoDB replication groups to replicate data to provide fault tolerance
- Use another application, such as Slack, WhatsApp, instead of WeChat to achieve notification services
- Use machine learning to predict rental

DEMO

Conclusion

- End-to-end design and implement the system
- Front-end framework: Angular, Ract
- Back-end: Flask
- Depolyment: Nginx, Docker