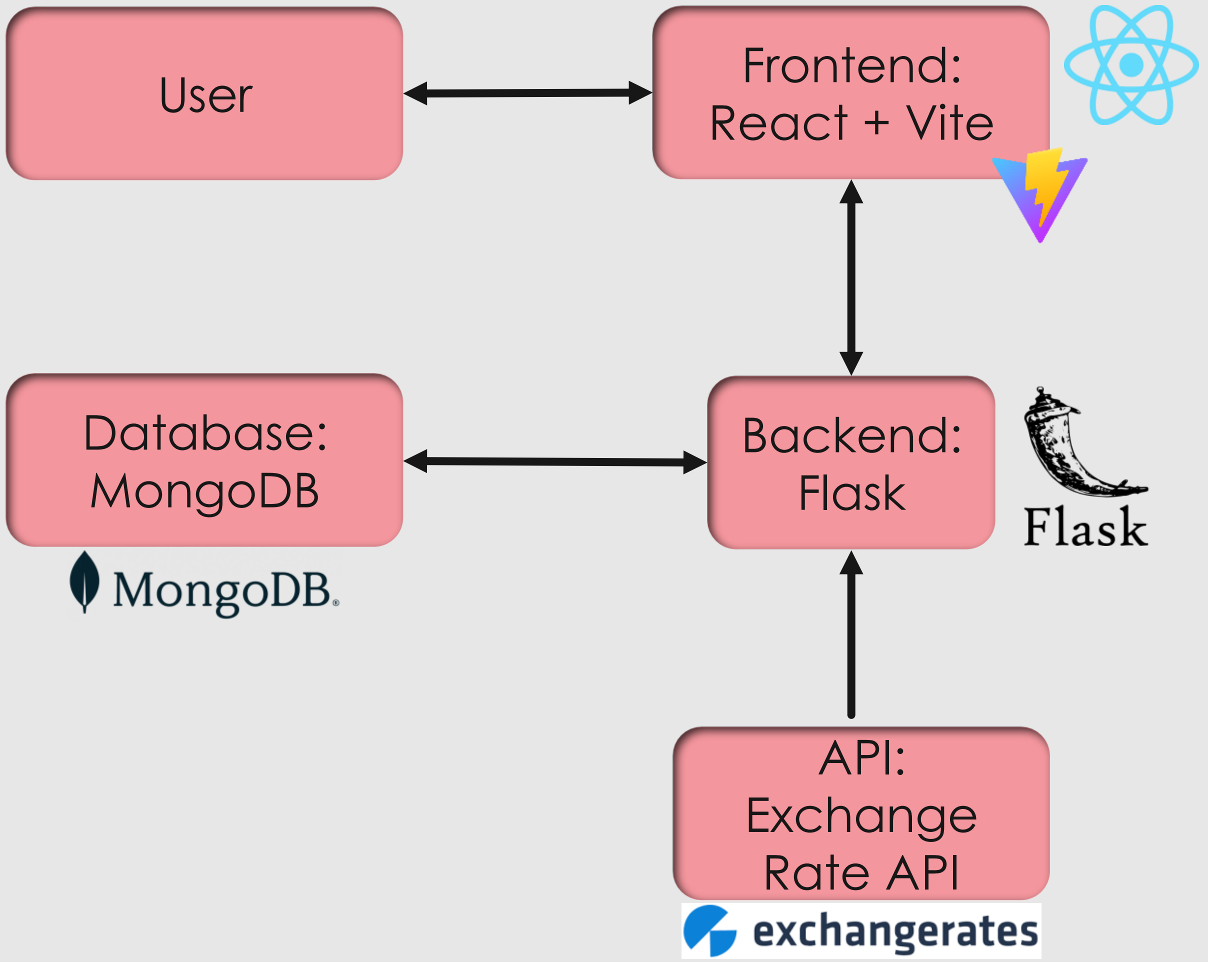
Travel Tracker

Olivia Folsom, Tasmia Iqbal, Panhapich Leang, Olivia Tarsillo

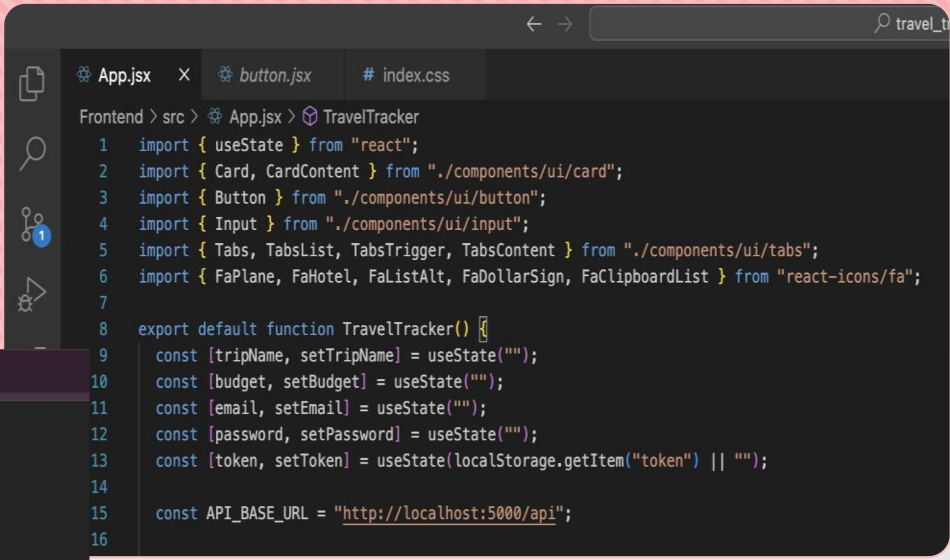
- Travel Tracker is a vacation planning web application
- Users can manage the logistics of their trips:
  - Budget
  - Flight information
  - Stay information
  - Itinerary
  - Notes
- Currency converter for out-of-country trips



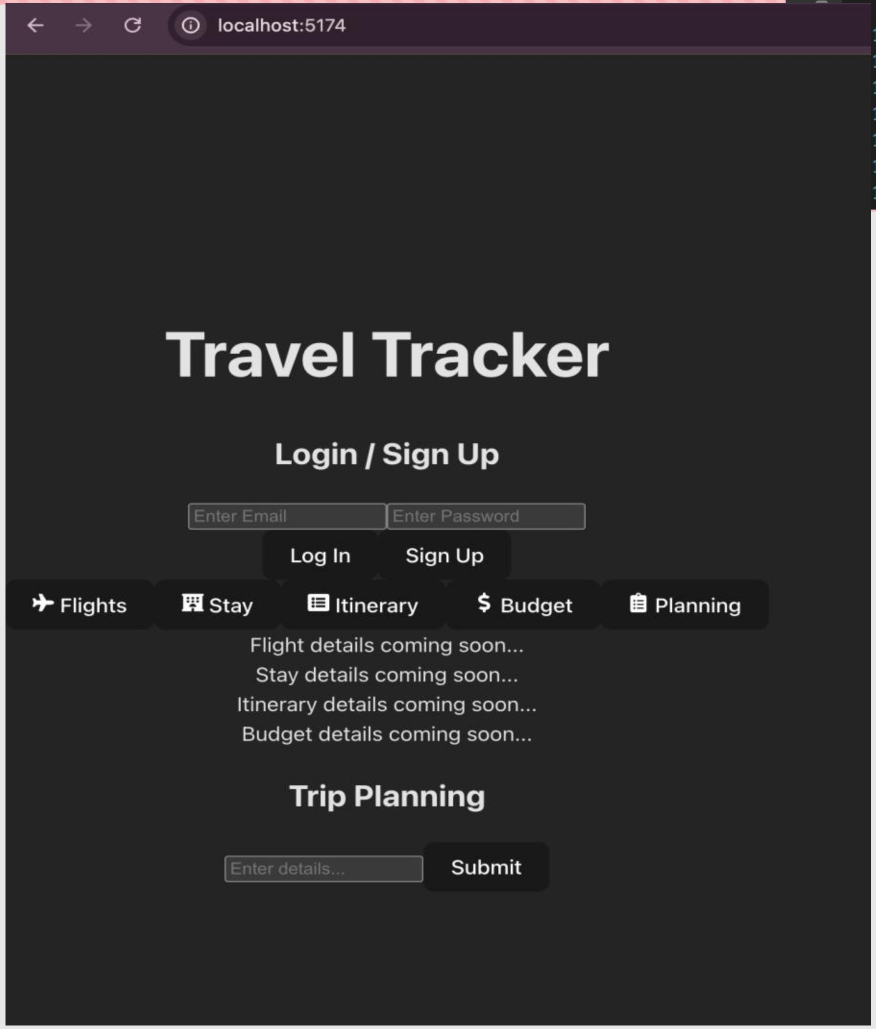
Implementation Plan - Frontend

Technologies that we currently used for frontend:

- React (for frontend framework & library)
- ShadCN UI and Tailwind CSS (for styling)
- React Icons – FontAwesome (for icons)
- React Hooks –useState (for state management)
- Fetch API (for HTTP requests)
- Local Storage (authentication)
- Vite (build tool to run React)
- Docker for containerization



```
import { useState } from "react";
import { Card, CardContent } from "./components/ui/card";
import { Button } from "./components/ui/button";
import { Input } from "./components/ui/input";
import { Tabs, TabsList, TabsTrigger, TabsContent } from "./components/ui/tabs";
import { FaPlane, FaHotel, FaListAlt, FaDollarSign, FaClipboardList } from "react-icons/fa";

export default function TravelTracker() {
  const [tripName, setTripName] = useState("");
  const [budget, setBudget] = useState("");
  const [email, setEmail] = useState("");
  const [password, setPassword] = useState("");
  const [token, setToken] = useState(localStorage.getItem("token") || "");

  const API_BASE_URL = "http://localhost:5000/api";

```



Travel Tracker

Login / Sign Up

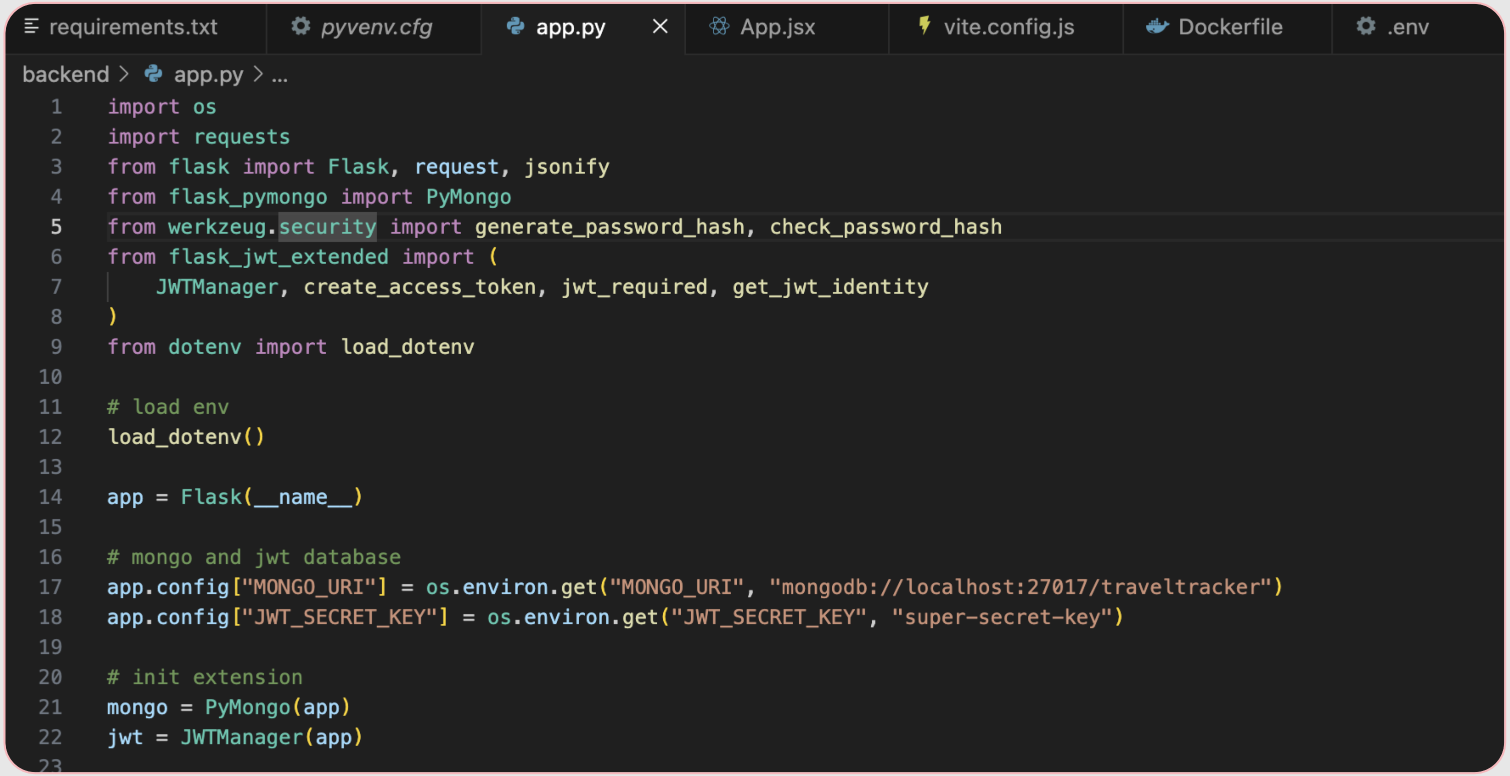
Flight details coming soon...
Stay details coming soon...
Itinerary details coming soon...
Budget details coming soon...

Trip Planning

Implementation Plan - Backend

What have we been using?

- Uses Python/Flask
- MongoDB as database
- JSON Web Tokens
- Libraries: Flask-PyMongo, flask-jwt-extended, Requests, python-dotenv, werkzeug
- Docker for containerization
- ExchangeRate-API



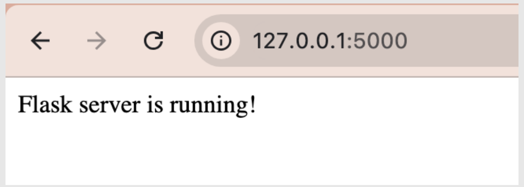
```python
import os
import requests
from flask import Flask, request, jsonify
from flask_pymongo import PyMongo
from werkzeug.security import generate_password_hash, check_password_hash
from flask_jwt_extended import (
    JWTManager, create_access_token, jwt_required, get_jwt_identity
)
from dotenv import load_dotenv

# load env
load_dotenv()

app = Flask(__name__)

# mongo and jwt database
app.config["MONGO_URI"] = os.environ.get("MONGO_URI", "mongodb://localhost:27017/traveltracker")
app.config["JWT_SECRET_KEY"] = os.environ.get("JWT_SECRET_KEY", "super-secret-key")

# init extension
mongo = PyMongo(app)
jwt = JWTManager(app)
```

Implementation Plan - Backend

- MongoDB:1 DB
  - 2 Collections: Users and Trips
- Test data
- Backend uses HTTP POST and GET
- Adding a PUT method and update functionality to the DB

How have we progressed?

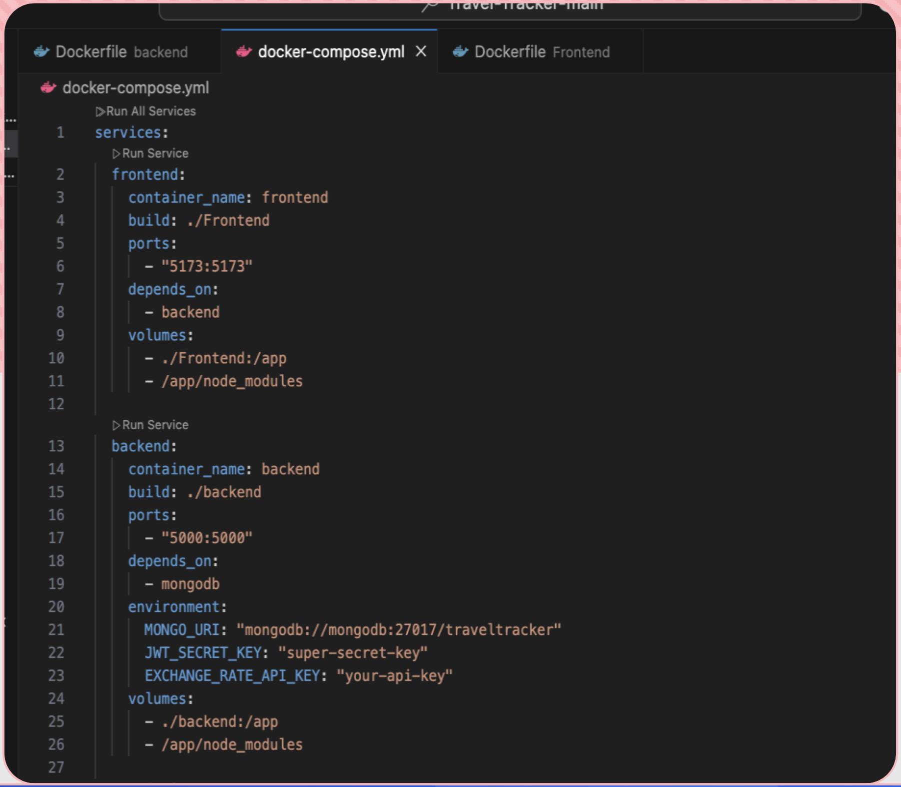
- Working Server!
- Functions for signing up and login
- Functions for saving trip info
- Functions for converting trip info
- Working Database.
- Finetuned what we want in backend
- Separating login & active page



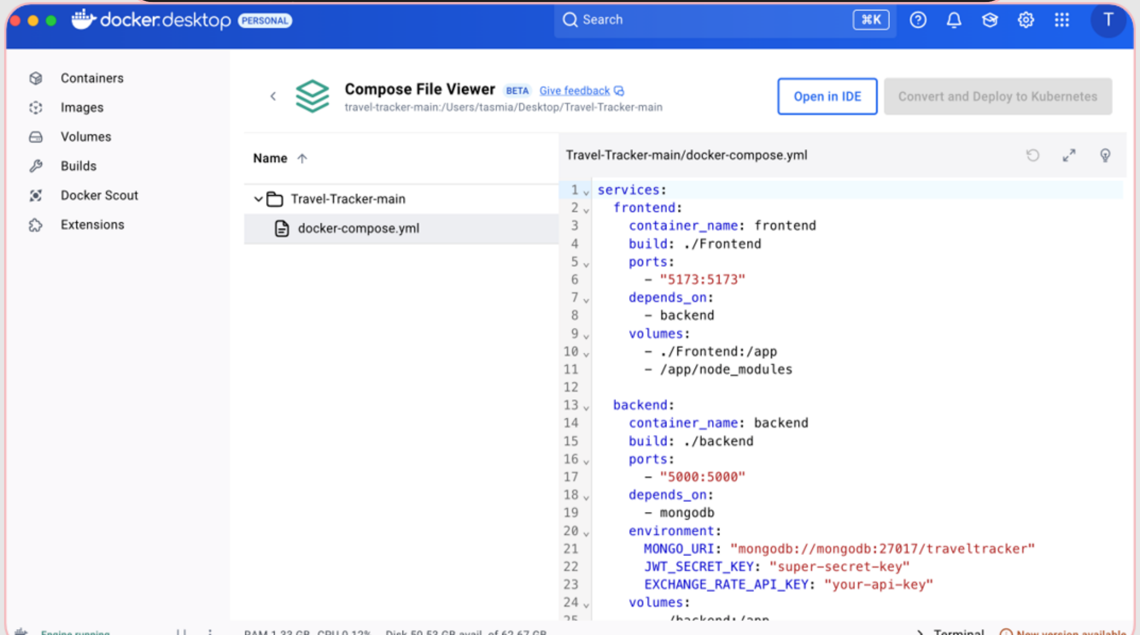
Flask server is running!

Docker Setup

- In Docker setup, each component (frontend and backend) has its own Docker file.
- Docker Compose managing multi-container Docker applications by defining services in a docker-compose.yml file.
- **Frontend**: Runs the frontend application, exposes port 5173, and depends on the backend.
- **Backend**: Runs the backend API, exposes port 5000, depends on MongoDB, and uses environment variables to configure MongoDB and API keys.
- **MongoDB**: Runs the MongoDB database, stores data persistently using a mounted volume, and exposes port 27017.



```yaml
services:
  frontend:
    container_name: frontend
    build: ./Frontend
    ports:
      - "5173:5173"
    depends_on:
      - backend
    volumes:
      - ./Frontend:/app
      - /app/node_modules

  backend:
    container_name: backend
    build: ./backend
    ports:
      - "5000:5000"
    depends_on:
      - mongodb
    environment:
      MONGO_URI: "mongodb://mongodb:27017/traveltracker"
      JWT_SECRET_KEY: "super-secret-key"
      EXCHANGE_RATE_API_KEY: "your-api-key"
    volumes:
      - ./backend:/app
      - /app/node_modules

```


docker desktop PERSONAL
Search
⌘K
T
Containers
Images
Volumes
Builds
Docker Scout
Extensions
Compose File Viewer BETA Give feedback
travel-tracker-main:/Users/tasmia/Desktop/Travel-Tracker-main
Open in IDE
Convert and Deploy to Kubernetes
Name
Travel-Tracker-main
docker-compose.yml
Travel-Tracker-main/docker-compose.yml

```yaml
services:
  frontend:
    container_name: frontend
    build: ./Frontend
    ports:
      - "5173:5173"
    depends_on:
      - backend
    volumes:
      - ./Frontend:/app
      - /app/node_modules

  backend:
    container_name: backend
    build: ./backend
    ports:
      - "5000:5000"
    depends_on:
      - mongodb
    environment:
      MONGO_URI: "mongodb://mongodb:27017/traveltracker"
      JWT_SECRET_KEY: "super-secret-key"
      EXCHANGE_RATE_API_KEY: "your-api-key"
    volumes:
```