PAW-CHEF

Mehul Aneja
mehulaneja123@gmail.com
12 July 2024

About Paw-Chef

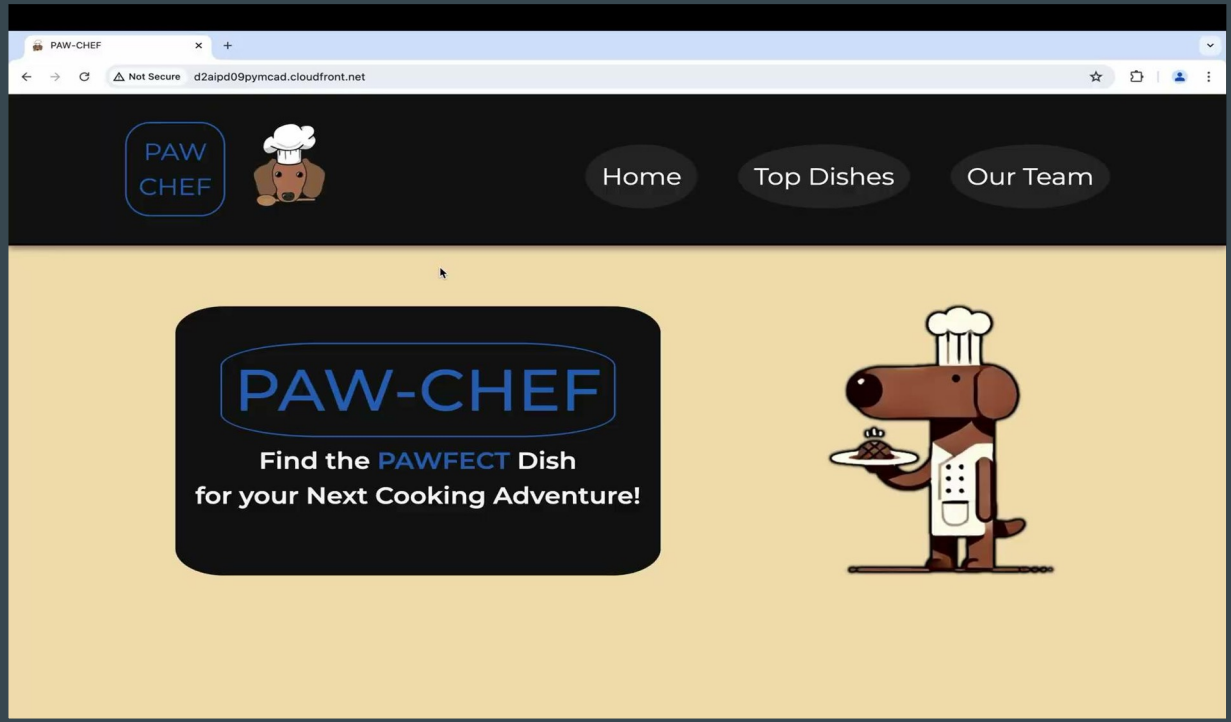
Paw-Chef is a cooking platform where users can view and share their favorite dishes

Project Links

Github Repo - https://github.com/Mehul-Aneja/paw-chef

Note – This was a project for me to able to explore and learn AWS architecture. As such, the project is no longer available there as it was a financial expense

Website Demo



https://drive.google.com/file/d/1KWCinGRS47T2JHgAUuj0KJd95KpBbRGU/view

Frontend

The frontend of Paw-Chef is built using **React**
Styling is managed through plain **CSS**
Bootstrap for some icons such as the delete button on the dish cards
React-router-dom for dynamic routing
React-loading-skeleton for a loading skeleton for card components when data is being fetched

Talking about **components**, each individual dish card is a separate component - same with each individual team card. The navbar is also another component.

Please see the Github repo for more details.

Frontend interacts with the backend by using the **Fetch API** for making requests.

Backend

The backend of Paw-Chef is built using **Python & Django**

Pillow for image processing. Backend stores these images in a media folder.

Project contains two apps - ‘Dishes’ and ‘Team’ with each app having its separate functionality for handling views, URLs, and models. For more details refer to the Github repo please.

Backend provides a RESTful API for the frontend to interact with to perform CRUD operations.

GET at **/dishes/api/dishes/** and **/team/api/team/** for retrieving data.
/dishes/health/ for health check
DELETE at **/dishes/api/dishes/delete/${id}** for removing a dish + more
POST using the add image form data at **/dishes/api/dishes/** + more

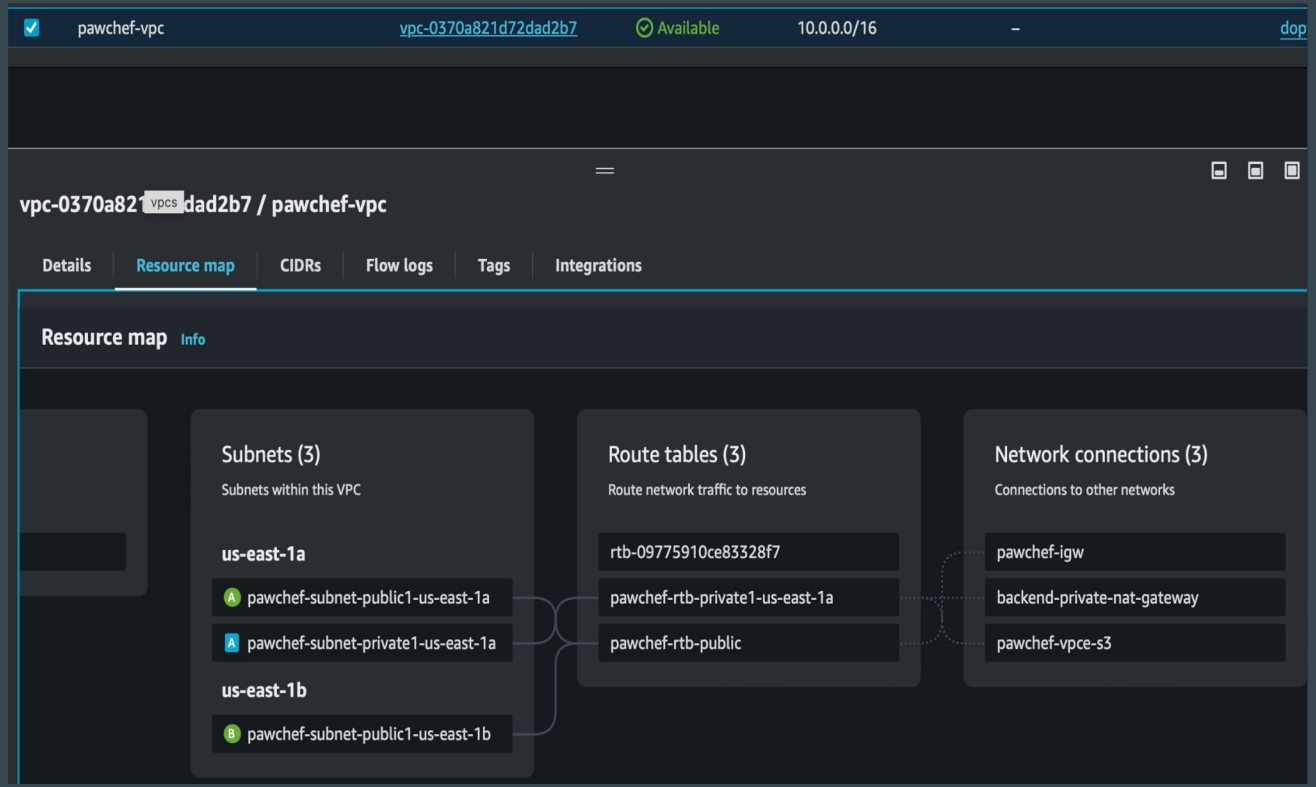
AWS Deployment and Architecture

1) CUSTOM VPC

Created a custom VPC for isolating network environment.

Architecture Overview:

- 2 public subnets
- 1 private subnet
- Associated route tables
- Associated security groups
- Internet gateway
- NAT gateway
- S3 endpoint
- Application Load Balancer
- Cloudfront CDN



2) SUBNETS

- **2 Public Subnets** for the resources accessible from the internet, such as the EC2 Bastion Host used for accessing the private EC2 backend instance.

  Internet Gateway allows internet access for resources in public subnets

  NAT Gateway allows internet access for instances in the private subnet without allowing them to be directly exposed to the internet

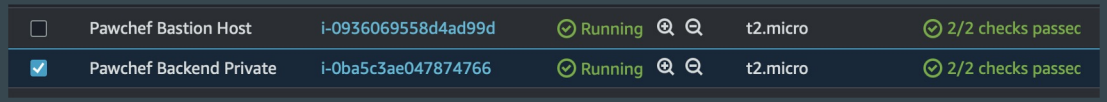
  Route table to route traffic through Internet Gateway

- **1 Private Subnet** for hosting the EC2 backend instance

  Route table to use NAT Gateway for outbound internet access

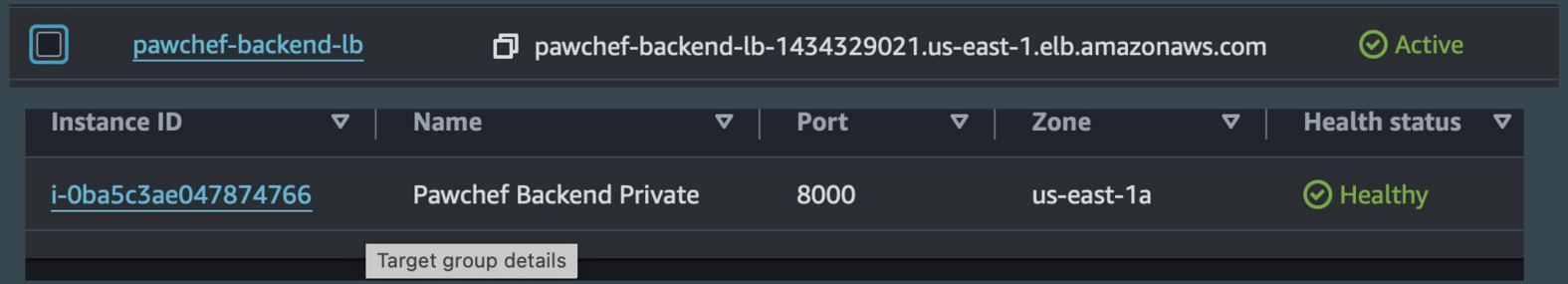
3) EC2

- **Pawchef Bastion Host** in the public subnet to securely connect to instance in private subnet. Associated security group allows SSH access only from known IPs for security.
- **Pawchef Backend Private** in the private subnet for running the Django application. It allows access from the Application Load Balancer and the Bastion Host ip address



4) APPLICATION LOAD BALANCER

- Listener routes traffic to target group containing the private backend instance as target
- Health check - status 200 and ‘OK’ for verifying validity



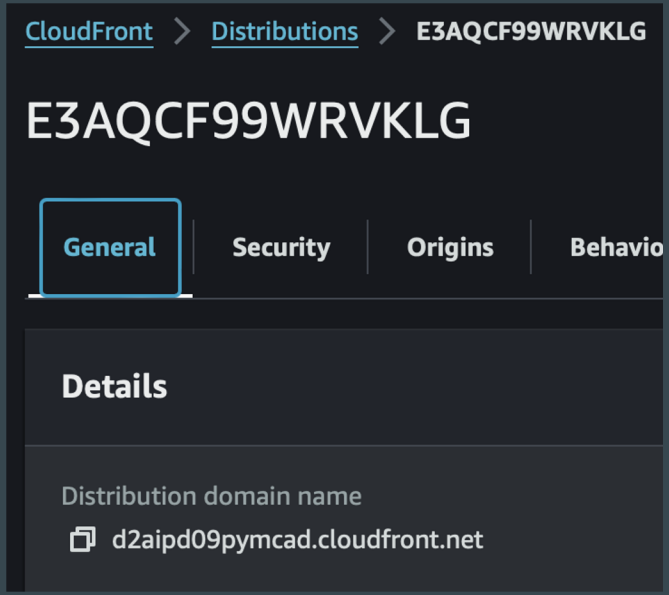
5) S3 BUCKET

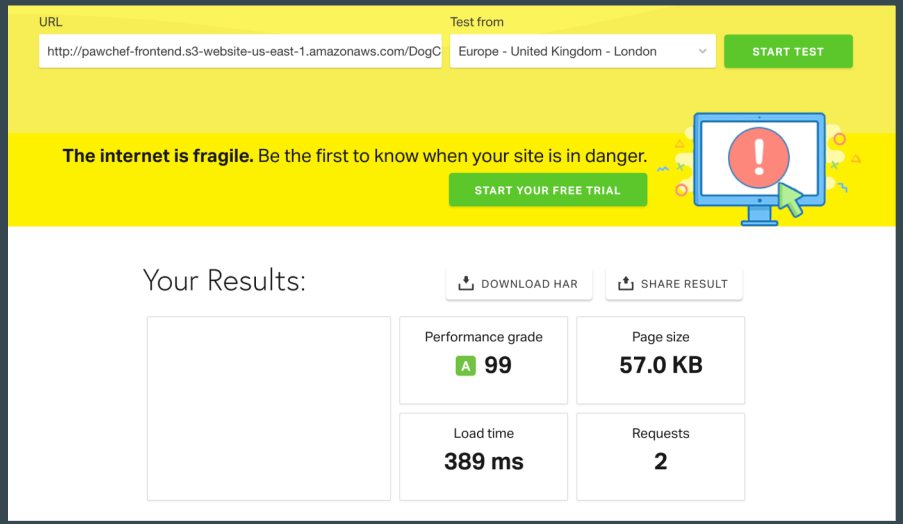
- Hosted the React frontend on an S3 bucket configured for static website hosting
- Uploaded contents of build directory after running ‘npm run build’ command on the frontend of the project

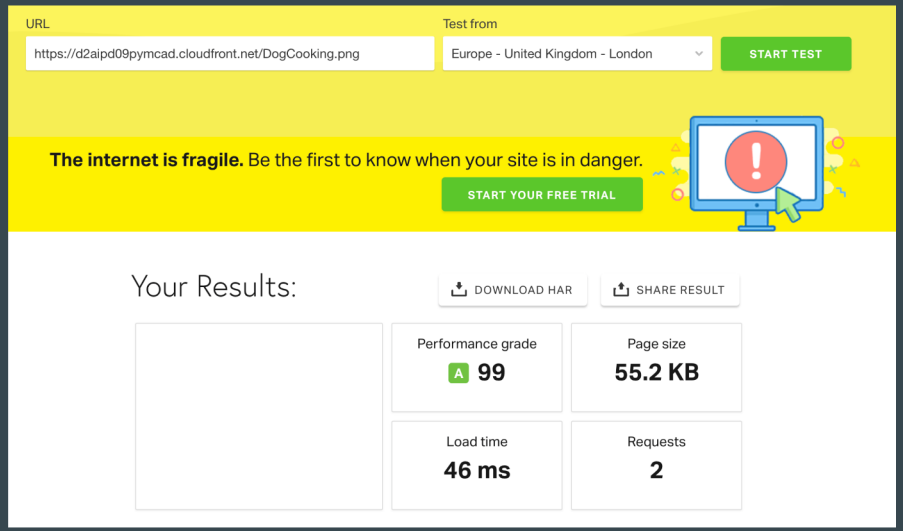
	Name ▲	AWS Region ▽	IAM Access Analyzer	Creation date
○	pawchef-frontend	US East (N. Virginia) us-east-1	View analyzer for us-east-1	July 11, 2024, 02:23:50 (UTC-04:00)

6) CLOUDFRONT CDN

- Created a Cloudfront distribution pointing to the S3 Bucket so that content from S3 Bucket can be cached and delivered from the CDN instead of the Bucket for faster loading times.







Future Work

- Add unit tests
- Code review, refactoring, and abstraction
- Additional core features
- Add user profiles and ability to sign up/login and save recipes
- Delete images from storage when a dish is deleted
- ...