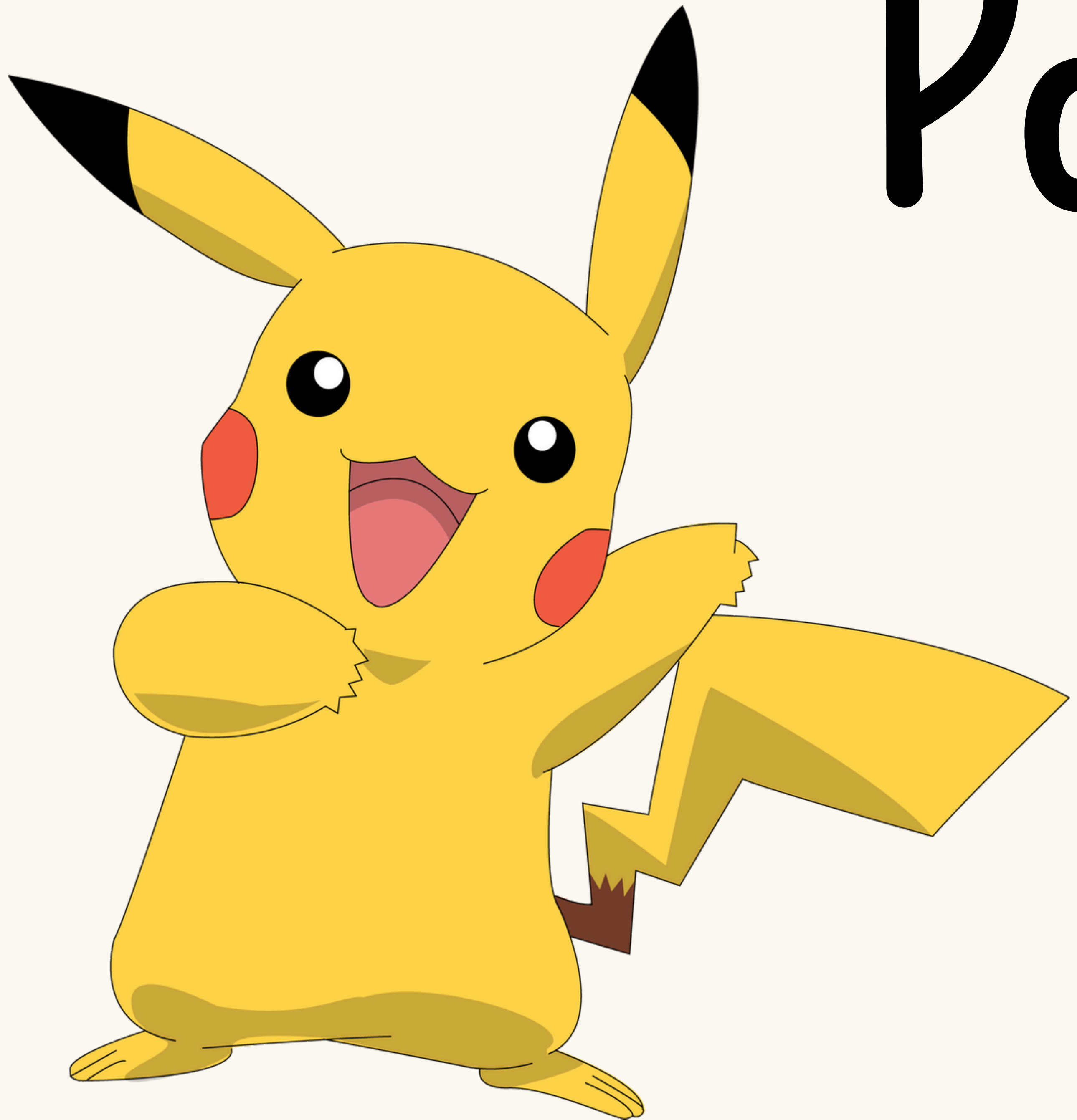
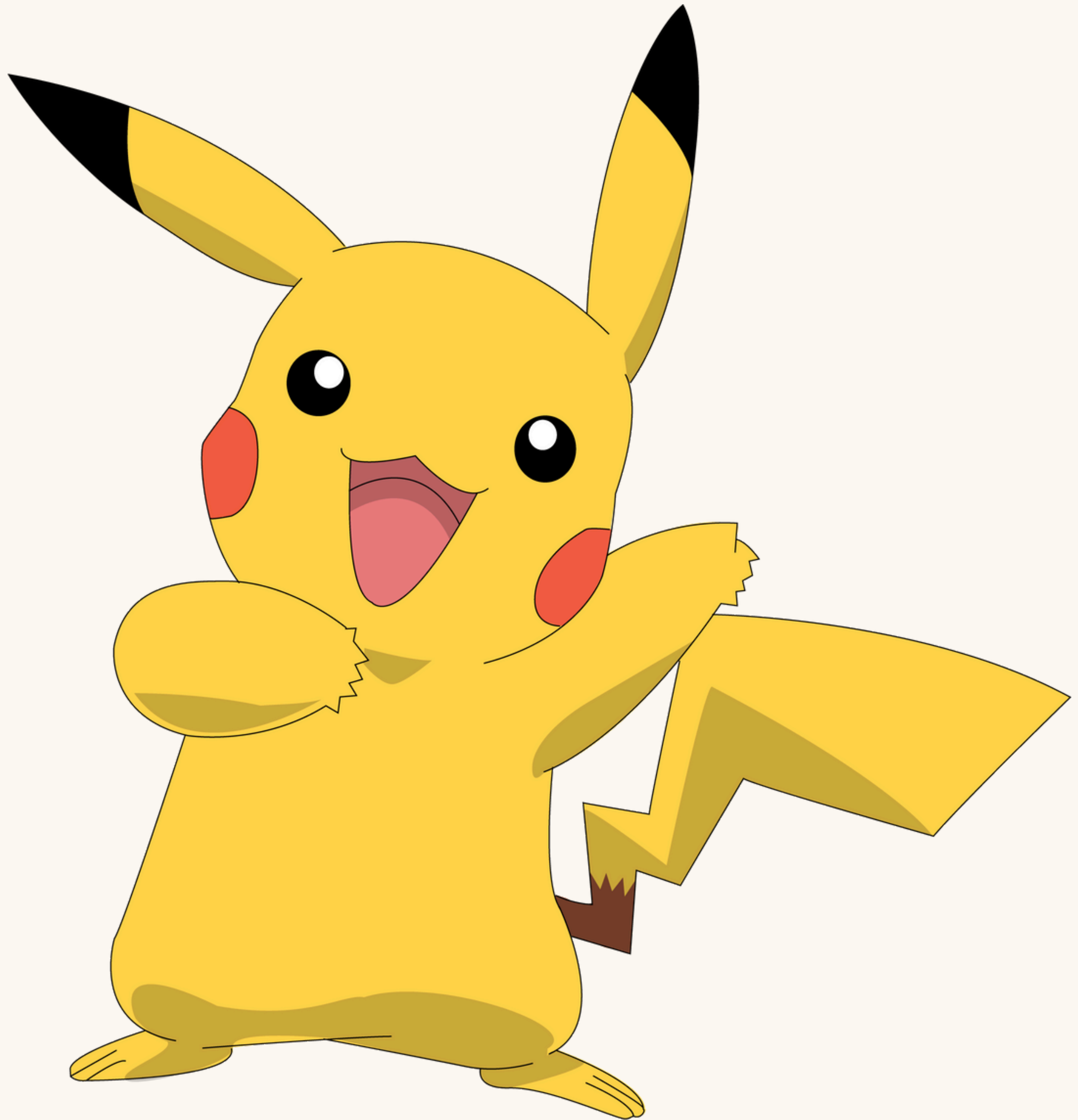
Nadine Jubeh

Pokémon App



aws

Agenda



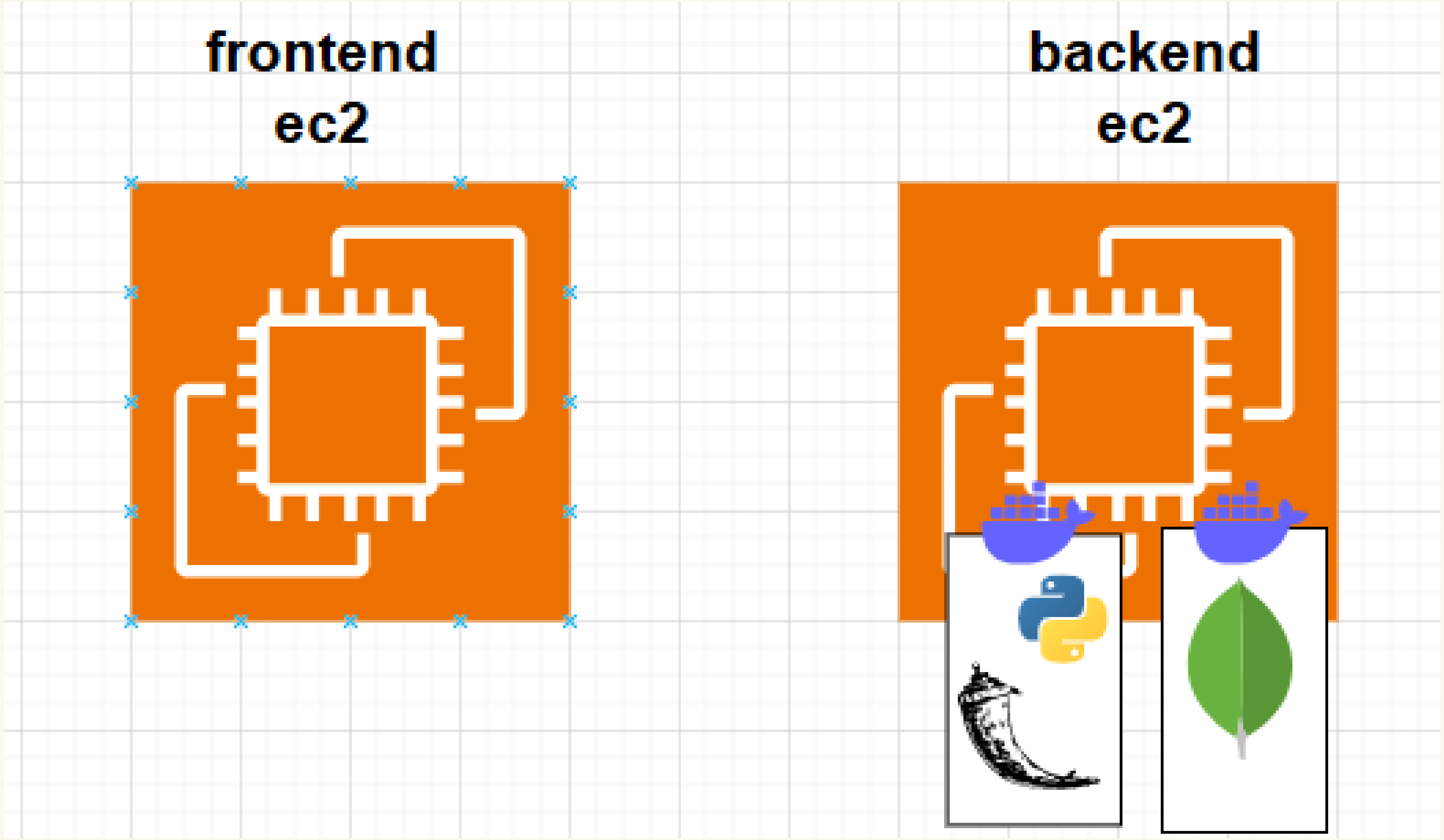
Description

The Pokémon Game is a cloud-based application where a Pokémon-themed CLI game interacts in real-time with a custom backend via a web API. This project integrates containerized services, web APIs, and infrastructure automation.

It runs across two EC2 instances on AWS:

- One for the PokeAPI game
- One for the Flask + MongoDB backend

Architecture



Technologies

Terraform

Infrastructure as Code tool for provisioning AWS resources including EC2, VPC, and security groups.

AWS EC2

Cloud computing service for hosting both frontend and backend servers with scalable instances.

MongoDB

NoSQL database for storing Pokémon data with flexible document schemas.

Ansible

Configuration management tool for automating server setup and application deployment.

Docker

Containerization platform for packaging the backend application with all dependencies.

Python

Programming language used for both the Flask backend API and the frontend gaming application.

flow

1 **Infrastructure Provisioning**

Terraform creates AWS resources including VPC, subnets, security groups, and EC2 instances configured with user data scripts for automatic setup on launch.

2 **Server Configuration**

Ansible configures frontend servers with required dependencies. (user data replacement)

flow

3 **Backend Deployment**

Docker containers with Flask API and MongoDB are deployed on the backend server.

4 **Frontend Execution**

Python frontend application connects to the backend API to fetch and display Pokémon data.

5 **User Interaction**

Users interact with the frontend application, which communicates with the backend for data operations.

Network Configuration

Setting up proper security groups and network rules to allow communication between frontend and backend while maintaining security.

Service Orchestration

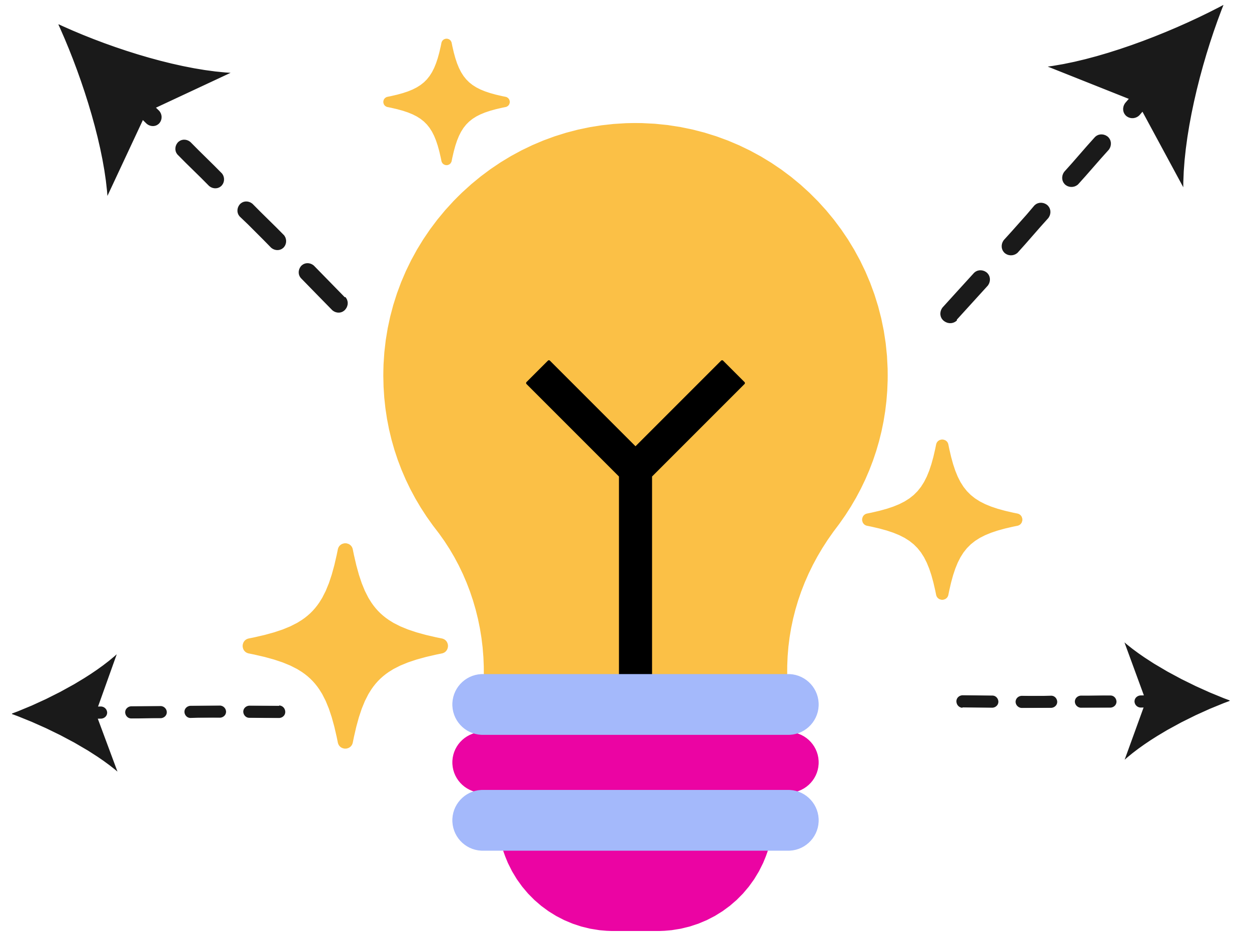
Coordinating the deployment sequence between Terraform infrastructure creation Ansible configuration, and application startup to ensure dependencies are properly resolved.

Challenges

WHAT HAVE I LEARNED

Containerization

Cloud Infrastructure Management



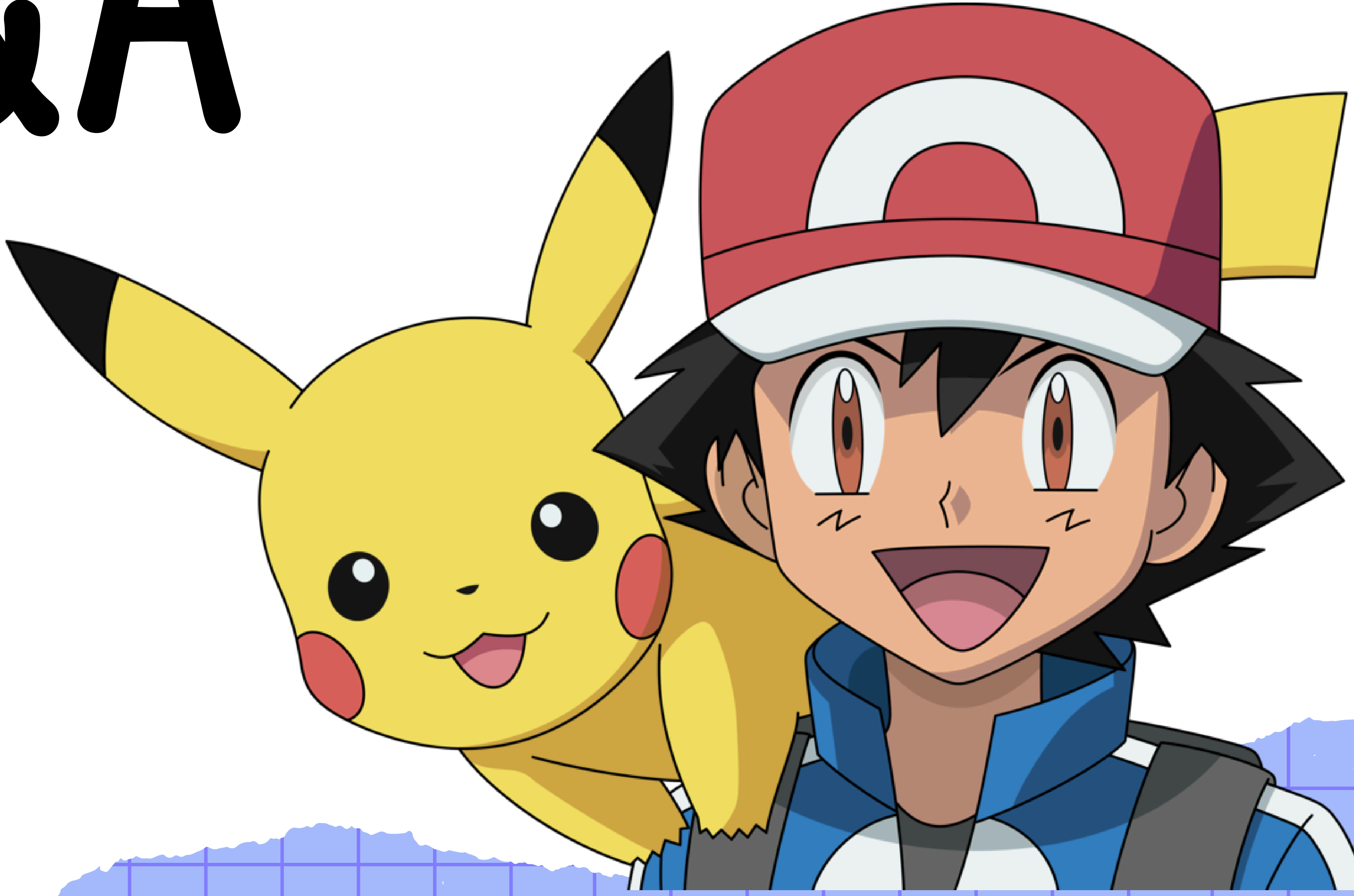
Flask

Ansible

Plans for the future

- go global
- IAM ROLES
- graphical frontend

Q&A



THANK YOU!