Part 1 - Typescript Backend

1. Write a backend service using NodeJS/Typescript and Express
    a. The backend should expose the following HTTPpaths:
        i. /health
            GET: returns 200 OK if the service is healthy
        ii. /ready
            GET: returns 200 OK if the service is ready
        iii. /candidate (manage/create/retrieve candidates records from postgresql)
            GET: See all info about a candidate
            POST: Create/Update a candidate's details or status
            DELETE: Delete a candidate
2. Build the service as a docker image
3. Create a docker-compose file that spin up the backend service and a postgres database
4. Run some queries and make sure it works as expected

Part 2 - Pulumi AWS Typescript

1. In the same github repository as the backend service, create a pulumi project - through which we'll create all of therequired infrastructure in AWS and deploy our backend
2. There should be 2 environments spinned up by the pulumi code- dev, and prod.
   Each environment will have its own stack.
   Branches will affect the dev environment, while master will affect the prod environment.
3. Each environment will have the following resources:
    a. VPC (2 public subnets, 2 private subnets, 1 NAT Gateway,1 Availability Zone), RDS (Smallest Instance Type), ECS Service (Build & Deploy the backend's Docker image on Fargate), ELB (Pointing at the ECS), Cloudwatch Dashboard (CPU & Memory Utilization only).