**Part 1:**

**Locally all worked well, but got little confused and didn’t have time to make app connect with Mongo on GCP**

CI/CD pipeline was built using GitHub Actions for deploying a todo application to Google Cloud Run. The pipeline is configured to run tests, build Docker images, push them to Google Container Registry, and deploy them to Google Cloud Run.

**Pipeline Workflow**

1. **Trigger:** The pipeline is triggered on push events to the main or master branches.
2. **Jobs:**

**Job 1: Run Tests**

* + **Purpose:** Execute tests to ensure code quality and correctness.
  + **Steps:**
    1. **Checkout Repository:**
       - Uses actions/checkout@v4 to check out the repository.
    2. **Set Environment Variables:**
       - Sets the PORT environment variable to 3000.
    3. **Build and Test with Docker Compose:**
       - Builds and runs the tests using Docker Compose with the configuration defined in Task 1/app/compose.yaml.

**Job 2: Build and Push**

* + **Purpose:** Build Docker images for the app and mongo services and push them to Google Container Registry.
  + **Steps:**
    1. **Checkout Repository:**
       - Uses actions/checkout@v4 to check out the repository.
    2. **Set Environment Variables:**
       - Sets the PORT environment variable to 8080.
    3. **Authenticate to Google Cloud:**
       - Uses google-github-actions/auth@v1 to authenticate with Google Cloud using service account credentials stored in GitHub Secrets (GCP\_CREDENTIALS).
    4. **Set Up Google Cloud SDK:**
       - Uses google-github-actions/setup-gcloud@v1 to set up the Google Cloud SDK.
    5. **Authenticate Docker with GCP:**
       - Configures Docker to use Google Cloud credentials for pushing images.
    6. **Build and Push app Service Docker Image:**
       - Builds the Docker image for the app service, tags it, and pushes it to Google Container Registry.
    7. **Build and Push mongo Service Docker Image:**
       - Pulls the latest MongoDB image, tags it, and pushes it to Google Container Registry.

**Job 3: Deploy to Google Cloud Run**

* + **Purpose:** Deploy the app and mongo services to Google Cloud Run.
  + **Steps:**
    1. **Checkout Repository:**
       - Uses actions/checkout@v4 to check out the repository.
    2. **Set Environment Variables:**
       - Sets the PORT environment variable to 8080.
    3. **Authenticate to Google Cloud:**
       - Uses google-github-actions/auth@v1 to authenticate with Google Cloud.
    4. **Set Up Google Cloud SDK:**
       - Uses google-github-actions/setup-gcloud@v1 to set up the Google Cloud SDK.
    5. **Deploy mongo Service to Cloud Run:**
       - Deploys the mongo Docker image to Google Cloud Run, exposing it on port 27017.
    6. **Deploy app Service to Cloud Run:**
       - Deploys the app Docker image to Google Cloud Run, exposing it on port 8080.

**Part 2:**

**Was fully focused on making deploy work on GCP, so I was lack of time to configure HTTPS for secure communication and perform static code analysis.**

Security measures:

* Service account credentials (GCP\_CREDENTIALS) and other sensitive data (e.g., REGION, GCP\_PROJECT\_ID, REPOSITORY) are stored securely in GitHub Secrets.
* Port number and MongoDB URI were stored as environment variables.