Solar System CI/CD Pipeline Project

# Project Overview

The Solar System NodeJS Application is a full-stack project built with Node.js, HTML, and MongoDB. It displays data about the solar system and its planets, allowing users to interact with the application via a browser.

# GitHub Actions and Benefits

GitHub Actions is a CI/CD tool that automates workflows directly from GitHub repositories. By defining workflows in YAML files, it facilitates testing, building, and deploying code. The key benefits include:  
- Automation: Streamlines repetitive tasks such as testing and deployment.  
- Integration: Allows seamless integration with external services such as AWS, Docker, etc.  
- Scalability: Can handle complex workflows with multiple jobs and dependencies.  
GitHub Actions enhances collaboration, boosts productivity, and reduces the likelihood of errors in software delivery.

# CI/CD Overview

This project uses GitHub Actions for CI/CD:

* **CI**: Ensures code is tested packaged before deployment.
* **CD**: Automates deployment to AWS using Terraform, and Ansible.

# CI Pipeline (Continuous Integration)

This CI pipeline is responsible for automatically testing and validating code changes to ensure that new commits do not introduce issues.

1. Triggering Events The pipeline is triggered on two events:

* Push events to any branch.
* Pull requests to the main branch.

**Jobs:**

* **Job 1: Build**
  + Runs on: Ubuntu-latest.
  + Steps:
    1. Checkout the repository: Downloads the latest version of the repository.
    2. Set up Node.js: Installs the Node.js version specified in the project.
    3. Install dependencies: Runs npm install to install project dependencies.
    4. Run tests: Executes npm test to run the unit tests.
    5. Run linting using npm run lint to check code formatting issues.
* **Job 2: Static Code Analysis**
  + Runs on: Ubuntu-latest.
  + Steps:
    1. Checkout the repository.
    2. Install dependencies.
    3. Run static analysis to identify potential issues.
* **Job 3: Build and Push Docker Image**
  + Runs on: Ubuntu-latest.
  + Steps:
    1. Checkout the repository.
    2. Login to Docker Hub.
    3. Build Docker image using docker build.
    4. Tag and push the image to Docker Hub.

**CD Pipeline (Continuous Deployment)**

The CD pipeline automates the deployment of the Solar System application after passing the CI checks.

* **Job 1: Provisioning infrastructure using terraform**
  + Runs on: Ubuntu-latest.
  + Steps:
    1. Checkout the repository.
    2. Set up Terraform to manage AWS infrastructure.
    3. Terraform init and verify.
    4. Apply Terraform Plan.
    5. Attach Elastic IP to ensure access to the application.
* **Job 2: Installation and Configuration using Ansible**
  + Runs on: Ubuntu-latest.
  + Steps:
    1. Checkout the repository.
    2. Set up Ansible.
    3. Configure webserver, Prometheus, Grafana. And Kubernetes
    4. Run Ansible Playbook to configure to deploy the Node.js dockerized application to staging and production environment.
    5. Attach Elastic IP to ensure access to the application.

**Project Creators**

* Ahmad Nabil Mahmoud Zin El-Din
* Mohamed EL-Mahdy
* Ahmed Hisham
* Mohamed Alaa