**DEMO PROJECT**

**AUTOMATION TESTING DOCUMENTATION OVERVIEW**

**OVERVIEW:** Automation Testing refers to the use of specialized tools and scripts to automatically execute tests on software applications.It is a key aspect of modern software development, particularly in continuous integration and continuous delivery (CI/CD) environments.

**ROLES AND RESPONSIBILITIES:**

Integrate automated tests into CI/CD pipelines and ensure smooth execution of automated tests in various environments.

**CI/CD Pipeline Integration:**

**Pipeline Overview:**

The CI/CD pipeline is configured to automatically run tests at various stages:

Our CI/CD pipeline includes the following stages:

1. Code Commit
2. Build
3. Test
4. Deploy

**Jenkins Job Configuration**

* **Job Name**: automated-tests
* **Build Triggers**: Poll SCM, GitHub webhook

Pipeline Script (Jenkinsfile):

pipeline {

agent any

stages {

stage('Clone') {

steps {

git 'https://github.com/SucharithaSathupalli01/Project-Development.git'

}

}

stage('config') {

steps {

bat '''

git config user.name "harish"

git config user.email "harishcloud1810@gmail.com"

'''

}

}

stage('steps') {

steps {

bat '''

git init

git status

git add .

'''

}

}

}

}

**Environment Management**

* Use separate environments for development, testing, staging, and production.
* Manage environments using Infrastructure as Code tools like Terraform and Ansible.

**Test Execution**

* Manual tests are reserved for exploratory testing.
* All other tests are automated and run in the CI/CD pipeline.

**Pipeline Execution**

* Tests are automatically executed as part of the CI/CD pipeline defined in the Jenkinsfile.

**Reporting and Monitoring**

Test Reports

* Unit Reports: Generated in build/reports/tests/test
* Cypress Reports: Generated in cypress/reports

#### **Monitoring Test Results**

Use monitoring tools like Grafana and Kibana to visualize and track test results over time.

**Maintenance and Scalability**

**Updating Tests**

Update tests as the application evolves, ensuring they remain relevant and effective.

**Refactoring**

Regularly refactor tests to improve readability, performance, and maintainability.

**Scalability**

Scale test automation to handle larger and more complex applications using parallel execution and cloud-based testing environments.

This documentation should be maintained and updated regularly to reflect any changes in the tools, processes.