Test Automation Mentoring Program: Advanced

CI/CD. HOME TASK

Tech stack: .NET, Java, JavaScript

TABLE OF CONTENTS

INTRODUCTION

You will be working with Report Portal application deployed locally on your workstation during all the mentoring program.

The scope of this task is to work with CI tools in order to build CI process. The goals of this module are:

- Give a possibility to set up CI
- Learn and apply different clean code practices
- Practice with CI tool/Sonar
- Practice with code review

NOTE

Tasks here do not contain definitions of done. Your Mentor will define exact definitions individually for each task.

MARKS

Points	Mark	Description
1	0.5	Unsatisfactory
2	1	Unsatisfactory
3	1.5	Unsatisfactory
4	2	Satisfactorily
5	2.5	Satisfactorily
6	3	Satisfactorily
7	3.5	Good
8	4	Good
9	4.5	Excellent
10	5	Excellent

Home task is DONE if all mandatory tasks are completed and more than (>=) 4 points received.
Mentor can take away points if Mentee's code quality does not meet his/her requirements.

DEADLINE

It's expected that Mentee completes a task in 1 week. The deadline can be adjusted in case of:

1. Vacation
2. Illness
3. Business trip
4. Exceptional case

Before adjusting the deadline please notify and confirm with Coordinator about changes.

TASKS

1. CI

Complexity	Points	Necessity
Intermediate	2	Task is Mandatory

TASK

1. Set up CI tool (or use local installation or using your EPAM virtual machine). Choose CI tool from below:
    a. Jenkins*
    b. Bamboo
    c. TeamCity
2. Set up a pipeline that will do the following:

a. Build project and execute tests - after each commit to Master branch
b. Build project and execute tests - once per day using the latest source files
c. Publish test results (display pass/fail numbers in pipeline)

***NOTE: EPAM GitLab cannot be set up for work with Jenkins yet. So if you use these tools, please consider 2 options:**

1. **Use just Schedule as a trigger in task #2**
2. **Change VCS to GitHub. If you use GitHub please do not store sensitive info there (e.g. RP credentials).**

DEFENITIONS OF DONE

CI is configured for your project. From now you should use CI tool for each commit into code repository.

2. QUALITY GATE

Complexity	Points	Necessity
Intermediate	2	Task is Optional

TASK

Configure quality gate. Use at least below outputs:

1. Build output
2. SonarQube scanner output

You may add as many quality outputs as you wish.

DEFENITIONS OF DONE

Quality gate configured.

3. CONFIGURE YOUR TEST RUNNER JOB TO EXECUTE API TESTS IN DOCKER CONTAINER

Complexity	Points	Necessity
Intermediate	2	Task is Mandatory

TASK

1. Install docker on your test job runner
2. Create image with API tests
3. Execute image and publish results

DEFINITION OF DONE

CI/CD pipeline functioning using Docker Containers and run API tests.

4. IMPLEMENT IAC

Complexity	Points	Necessity
Advanced	2	Task is Mandatory

TASK

1. Create separate folder in your repository

2. Export your CI/CD pipeline configuration as a single file. Use .yaml if possible as a standard representation.
3. Reconfigure your CI/CD tool to use config file from your repository for every operation. You can decide about branches using and synch in with your branching strategy.

DEFINITION OF DONE

CI/CD pipeline use config file from your repository for execution.

5. TAF INTEGRATIONS

Complexity	Points	Necessity
Intermediate	2	Task is Mandatory

TASK

1. Messenger's integration – configure Slack/Teams integration to send Start/Finish run messages for any test run
2. Jira/ADO integration - implement integration with Jira/ADO to change test cases status for all UI tests according to test run results.
3. Sauce Labs integration - run your UI test (configured for Web RP testing) using Sauce Labs/Browserstack for at least 2 browsers
4. Reporting integration - implement integration with Report Portal to see the UI test run details. Include also Logging.

DEFENITIONS OF DONE

1. Appropriate classes are created for TAF integration with Jira and Slack
2. Test cases in Jira are getting updated from TAF according to automated test's run
3. Start/Finish run messages reported to the Slack from TAF
4. Appropriate classes are created for TAF integration with Sauce Labs
5. Test are being run in Sauce Labs
6. Your tests output is available in RP. You can also see logs and filter them according to level.

Ver.	Description of Change	Author	Date
1.0	Initial Version	Yury Karpinski	14-Oct-2020
1.1	Updated: Structure	Yury Karpinski	24-Jan-2020
1.2	Added note about Jenkins	Yury Karpinski	24-Jan-2021
1.3	Renewed text	Yury Karpinski	26-March-2022
2.0	Merge with Docker and TAF Integrations	Yury Karpinski	27-July-2022