

CONTINUOUS AND AGILE SOFTWARE ENGINEERING

Coursework

- Nikolaos Polykandriotis
- Ioannis Thanos

21/05/2021

Table of Contents

Prefix and Scope of the Document	3
First Enhancement	3
1-3. Git Clone, Import Project & Switch Branch	3
4-7. Extra Shapes Creation – Pull Request	4
8. Jenkins Pipeline Setup.....	5
9-10. Screenshots.....	6
Second Enhancement.....	8
11. Database Integration.....	8
12-18. View Statistics & Reset Statistics.....	8
19-20. Webhooks – Screenshots	9



Prefix and Scope of the Document

This section provides a high-level view of the coursework

NewWebApp is a project that aims to present its readers geometric shapes in a web application and ultimately collect and provide data about visits to individual geometric shape pages and generate a report with usage statistics.

This document aims to explain to its readers the steps (functional & technical) that were followed and the respective methodologies that were applied / incorporated into the respective codes. In addition, this document is aiming to hand over, with the help of screenshots and tool outputs throughout the development of the project, the deployment of the application along with some conclusions and issues that we faced during its creation.

After becoming familiar with this document, a reader will be able to:

- Understand the use of build tools and use Maven for expressing dependencies and automating the software building process.
- Understand source control management and use Git and Github for maintaining code bases.
- Understand Continuous Integration and Deployment and use Jenkins to support CI/CD pipelines.
- Understand containers and use Docker for fast deployment of applications.
- Apply automated deployment as part of the CI/CD pipeline to a remote virtual machine.

Finally, below is the link to our GitHub repository “NewWebApp.git”:

<https://github.com/Nikpolik/NewWebApp>



First Enhancement

1-3. Git Clone, Import Project & Switch Branch

First step, instead of cloning the repository (“https://github.com/oefremidis/NewWebApp.git”) and then push it into our own remote, we decided to use “Fork”. Fork will create a copy of the repository in our Github account so that we can make changes to the project. Below are the steps:

1. Go to the desired GitHub repository
2. Click the button “Fork” at the top right corner
3. A new repository with the exact same name will be created into our repository

We then imported the project to IntelliJ using the "Get from Version Control" option.

For Git operations we used the command line.

There are a few ways you can create new branches in Git, with many of them differing in how your branch is created from the main branch, whether it be from your current branch, a different branch, a tag, etc.

The most common way to create a new branch is the following:

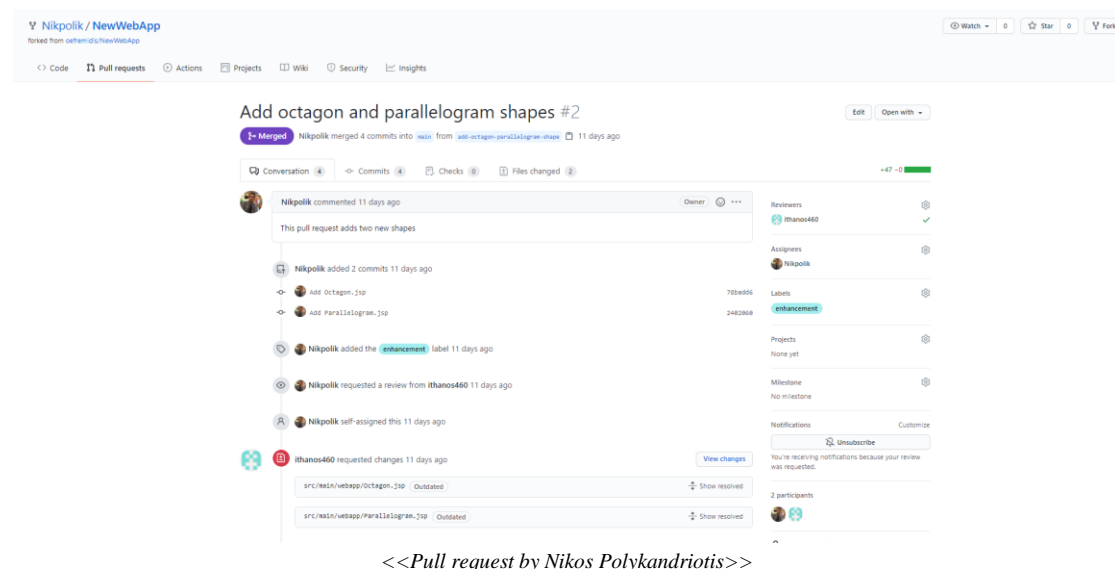
1. “\$ git checkout -b <branch-name>”
or
2. “\$ git branch <branch-name>”
“\$ git checkout new-branch”

4-7. Extra Shapes Creation – Pull Request

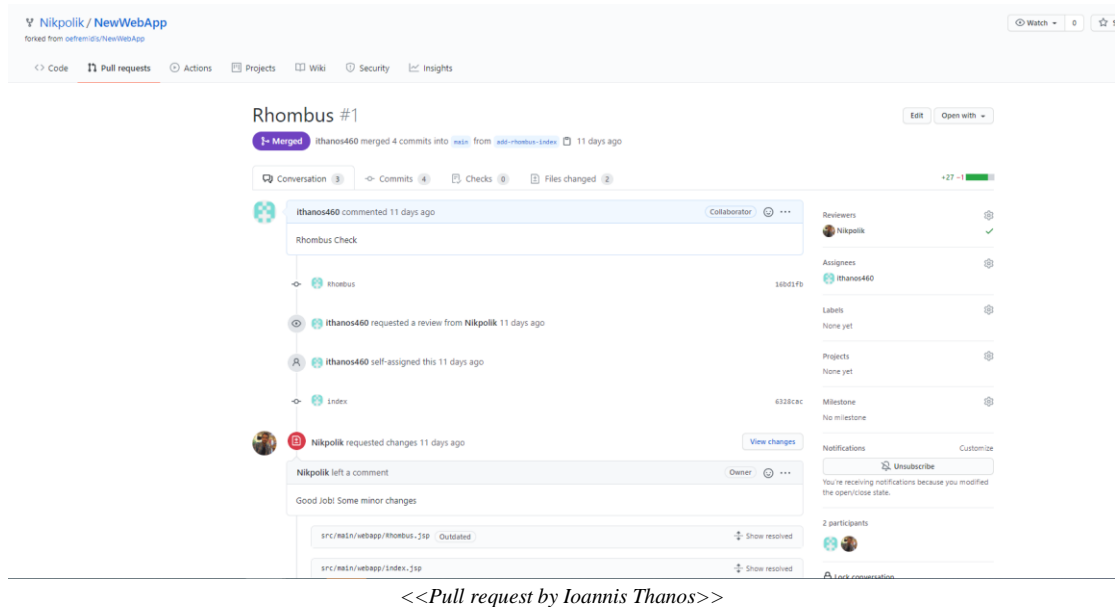
Each member of the team created the corresponding shapes in his local branch, committed them and finally pushed them to GitHub.

After that, each member opened a pull request and waited the review from the other.

Below are some screenshots from the aforementioned actions on GitHub:



<<Pull request by Nikos Polykandriotis>>



8. Jenkins Pipeline Setup

For creating our pipeline script there were two choices. Either use the built in editor of Jenkins or create our own script in our repository. We decided the second was better since we could use our project with any Jenkins instance. The steps we needed to take in order to deploy our application were :

- 1) Package our application in a war file
- 2) Create a docker image that runs tomcat and contains our site
- 3) Deploy the app by running the image we just created

The first two steps were easy since we just had to execute a set of predefined commands from maven and docker. By far the hardest part was deploying our application. We decided to deploy it in the same machine that Jenkins was running. To achieve this we had to find a way to stop a previously running container in order to restart the new one. After some research we found that it was possible by running the “`docker ps -a -q`” and saving its output in environment variable. This then allowed us to have a stage that run conditionally only when running containers were present (Stop Instances Stage in our Jenkinsfile). We also leveraged environment variables to simplify our script by defining the image name and our credentials (hardcoding them would be a security risk). After some failed attempts we managed to have the pipeline working.

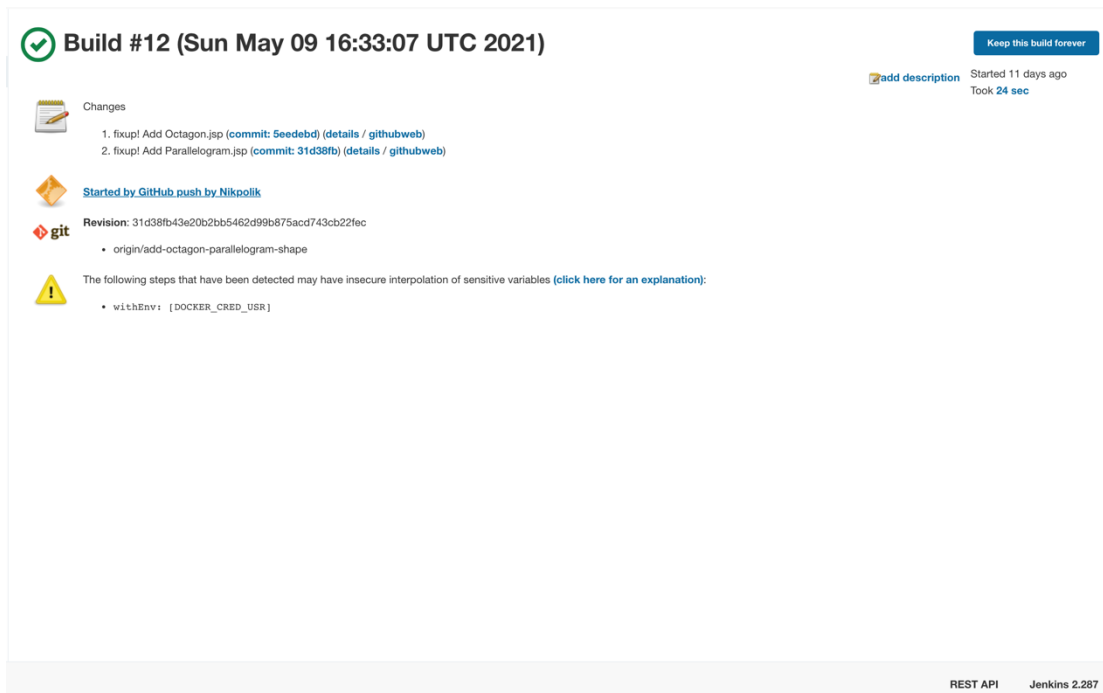
9-10. Screenshots

You can also view the output of Jenkin execution for this build in the below attached “Console_Output_Jenkins_shapes.txt” file:



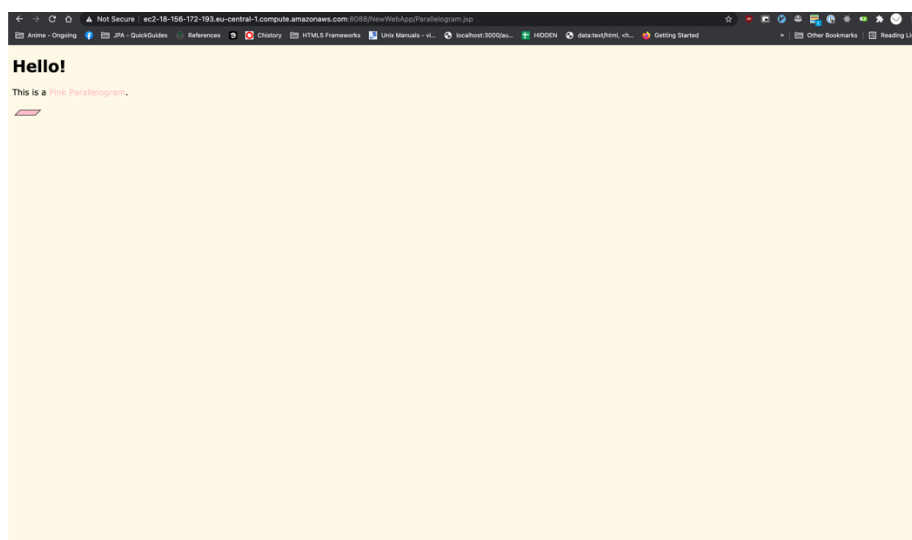
Console_Output_Jenkins_shapes.txt

Latest Jenkinsfile Link : “<https://github.com/Nikpolik/NewWebApp/blob/main/Jenkinsfile>”

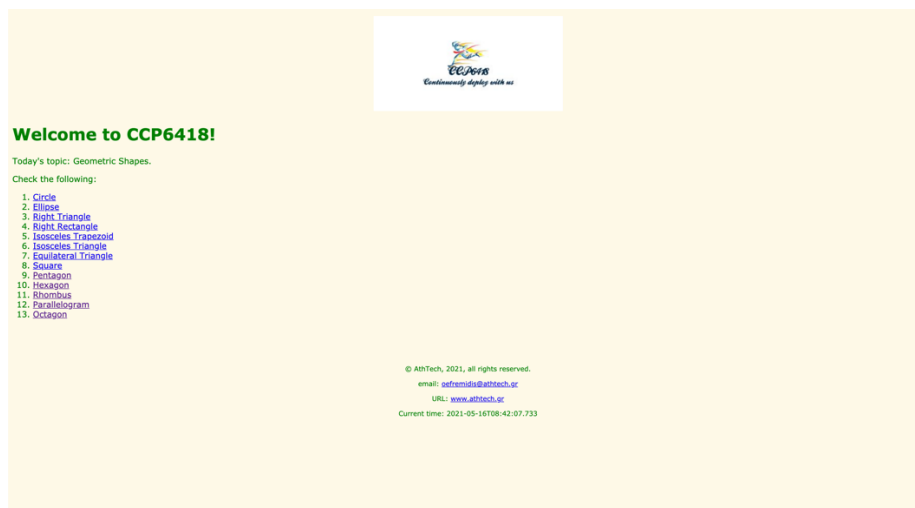


The screenshot shows the Jenkins Build #12 interface for a build completed on Sun May 09 16:33:07 UTC 2021. The build status is successful, indicated by a green checkmark. The interface includes a 'Keep this build forever' button and an 'add description' link. The 'Changes' section lists two commits: 'fixup! Add Octagon.jsp' and 'fixup! Add Parallelogram.jsp'. The 'Started by GitHub push by Nikpolik' section shows the build was triggered by a GitHub push. The 'Revision' section displays the commit hash '31d38fb43e20b2bb5462d99b875acd743cb22fec' and the branch 'origin/add-octagon-parallelogram-shape'. A warning icon indicates that the build steps may have insecure interpolation of sensitive variables, with a link to 'click here for an explanation'. The 'withEnv' section shows the environment variable 'DOCKER_CRED_USER'.

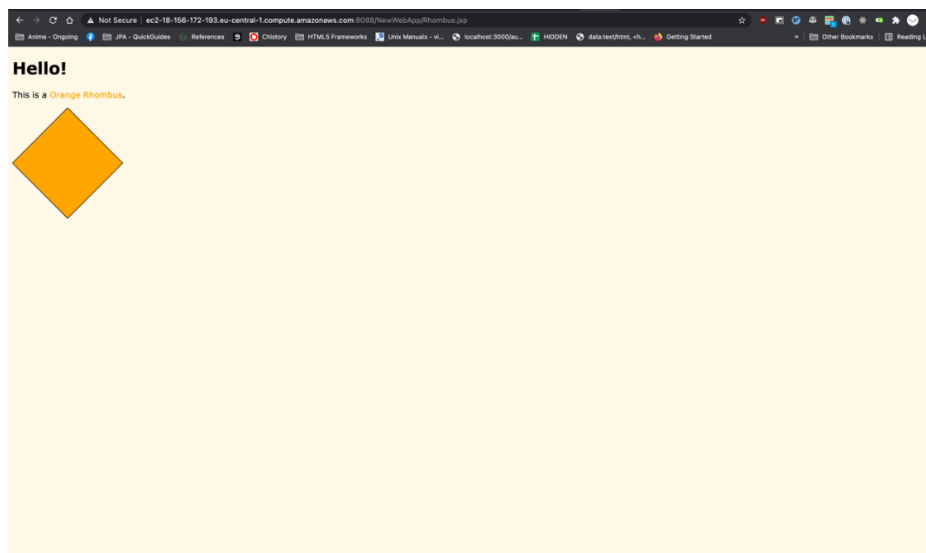
<<Jenkins execution after GitHub merge>>



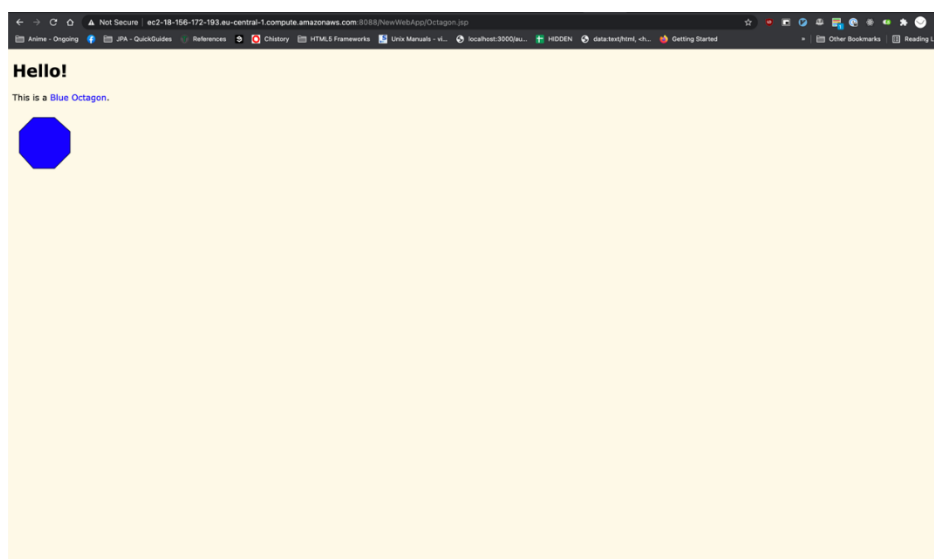
<<Page - Parallelogram>>



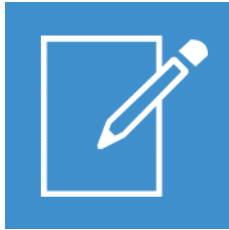
<<Page - Index>>



<<Page - Rhombus>>



<<Page - Octagon>>



Second Enhancement

11. Database Integration

In order to setup database integration for our project we had to add a maven dependency to our database driver. We searched on <https://mvnrepository.com> and found the corresponding driver in <https://mvnrepository.com/artifact/mysql/mysql-connector-java>. After that we put our new ShapeStatistics class in src/main/java so that maven can compile it and move it the proper directory where tomcat will have access to it.

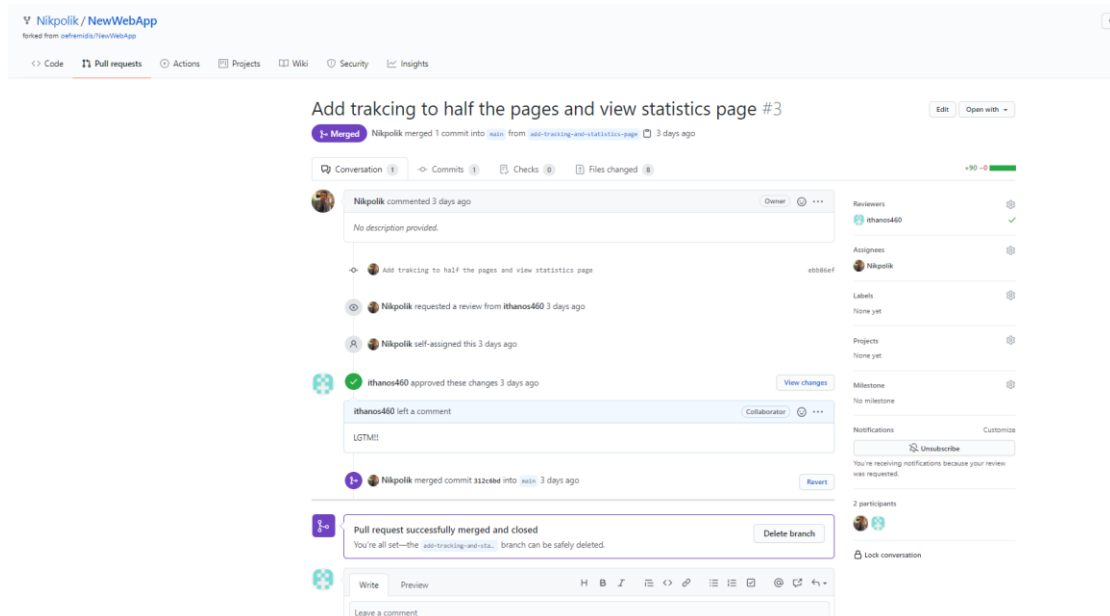
The second thing we had to do is update our deployment script to also start a database container. This at first seemed easy as the only thing we had to do was run an instance of https://hub.docker.com/_/mysql. However we found it quite difficult to pre seed our database with the proper tables. After reading the documentation we knew we had to put a *.sql script in /docker-entrypoint-initdb.d . Our first thought was to create a new docker volume that would map our /scripts folder with the corresponding directory inside the container. This proved quite hard since we had to set up proper permissions for docker to view the file as well as manage the volumes which had to be destroyed or reused so that they don't take up space on our vm. This made us decide to move in another direction. We created our own version of a mysql image with the seed files copied in the proper directory.

At this point we also realized that our command for stopping all containers was wrong and worked only for a single container. We had to update it work with both the web app container and the database container.

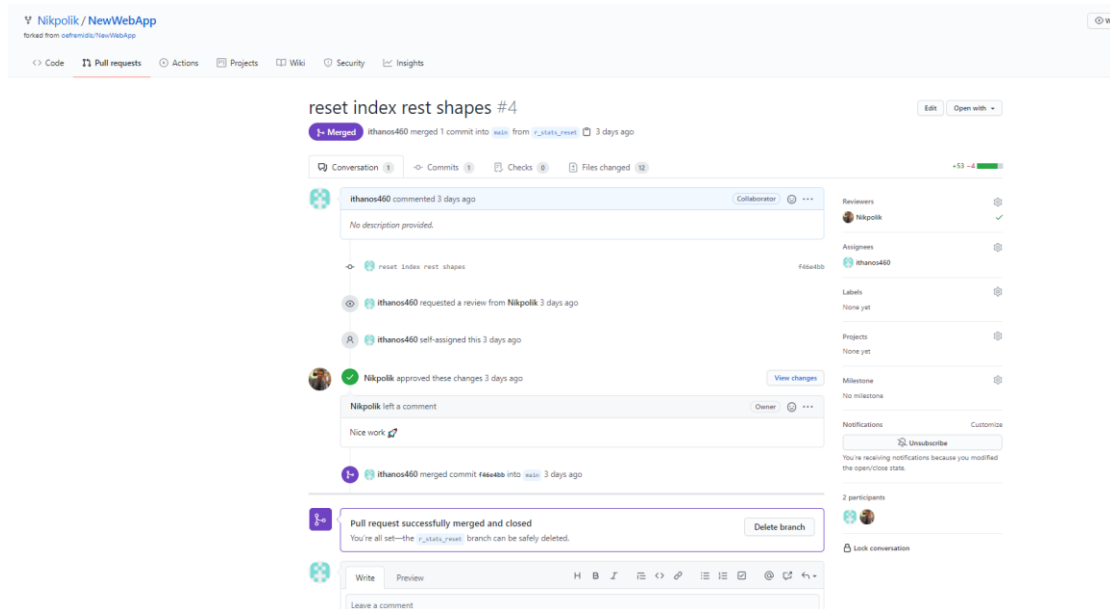
12-18. View Statistics & Reset Statistics

After finishing the database setup we switched to a new branch to develop the statistics tracking of each shape using the “git checkout -b <branch-name>” command.

Next steps were the same as in step “4-7. Extra Shapes Creation – Pull Request”, where we collaborated through GitHub to create View Statistics & Reset Statistics functionalities and finally merge the corresponding codes.



<<Pull request by Nikos Polykandriotis>>



<<Pull request by Ioannis Thanos>>

19-20. Webhooks – Screenshots

You can also view the output of Jenkin execution for this build in the below attached “Console_Output_Jenkins_database.txt” file:



Console_Output_Jenki
ns_database.txt

General Build Triggers Advanced Project Options Pipeline

Description

[Plain text] [Preview](#)

☐ Discard old builds

☐ Do not allow concurrent builds

☐ Do not allow the pipeline to resume if the master restarts

☒ GitHub project

Project url

<https://github.com/Nikopolis/NewWebApp.git>

[Advanced...](#)

☐ Pipeline speed/durability override

☐ Preserve stashes from completed builds

☐ This project is parameterized

☐ Throttle builds

Build Triggers

☐ Build after other projects are built

☐ Build periodically

☒ GitHub hook trigger for GITSCM polling

☒ Poll SCM

Schedule

H * * * * *

[Save](#) [Apply](#)

May 16, 2021 8:22:33 AM UTC; would next run at Sunday, May 16, 2021 9:22:33 AM UTC.

[Ignore post-commit hooks](#)

<<Jenkins Project Configuration_1>>

Dashboard > NewWebApp >

General Build Triggers Advanced Project Options Pipeline

Pipeline

Definition

Pipeline script from SCM

SCM

Git

Repositories

Repository URL

<https://github.com/Nikopolis/NewWebApp.git>

Credentials

- none - [Add](#)

[Advanced...](#)

[Add Repository](#)

Branches to build

Branch Specifier (blank for 'any')

/*

[Add Branch](#)

Repository browser

(Auto)

Additional Behaviours

[Add](#)

[Save](#) [Apply](#)

<<Jenkins Project Configuration_2>>

Build #75 (Tue May 18 18:05:32 UTC 2021)

Keep this build forever

add description

Started 2 days 15 hr ago
Took 27 sec



Changes

1. Fix link titles ([commit: ed3c45c](#)) ([details](#) / [githubweb](#))



Started by [GitHub push](#) by [Nikopolik](#)



Revision: [ed3c45c5f4dff827511b726be2ec9b2ece6b3cf](#)

- [refs/remotes/origin/main](#)

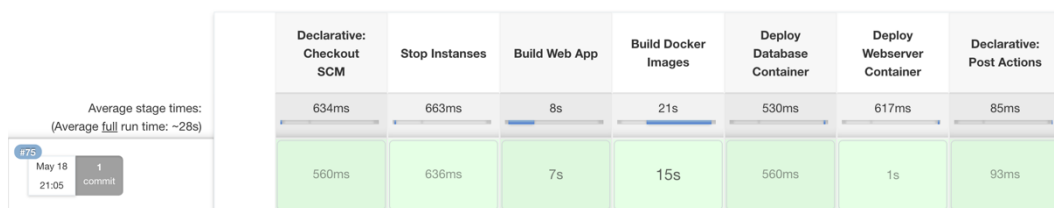


The following steps that have been detected may have insecure interpolation of sensitive variables ([click here for an explanation](#)):

- withEnv: [DOCKER_CRED_USR]
- sh: [DOCKER_CRED_USR, DOCKER_CRED_PSW]
- sh: [DOCKER_CRED_USR]
- sh: [DOCKER_CRED_USR]
- sh: [DOCKER_CRED_USR]
- sh: [DOCKER_CRED_USR]
- sh: [DOCKER_CRED_USR]
- sh: [DOCKER_CRED_USR]
- sh: [MYSQL_CRED_USR, MYSQL_CRED_PSW, DOCKER_CRED_USR]
- sh: [MYSQL_CRED_USR, MYSQL_CRED_PSW, DOCKER_CRED_USR]

REST API Jenkins 2.287

<<Jenkins Build>>



<<Jenkins Build_2>>

Page statistics reset

<<Page – Reset Statistics>>



Welcome to CCP6418!

Today's topic: Geometric Shapes.

Check the following:

1. [Circle](#)
2. [Ellipse](#)
3. [Right Triangle](#)
4. [Right Rectangle](#)
5. [Isosceles Trapezoid](#)
6. [Isosceles Triangle](#)
7. [Equilateral Triangle](#)
8. [Square](#)
9. [Pentagon](#)
10. [Hexagon](#)
11. [Rhombus](#)
12. [Parallelogram](#)
13. [Octagon](#)
14. [ResetStatistics](#)
15. [ViewStatistics](#)

© AthTech, 2021, all rights reserved.
email: oefermidis@athtech.gr
URL: www.athtech.gr
Current time: 2021-05-20T16:12:55.554

<<Page – Index Final>>

Circle: 2 visit(s).
2021-05-21 09:49:10.0
2021-05-21 09:49:23.0
Ellipse: 1 visit(s).
2021-05-21 09:49:24.0
EquilateralTriangle: 1 visit(s).
2021-05-21 09:49:17.0
IsoscelesTrapezoid: 1 visit(s).
2021-05-21 09:49:19.0
Square: 1 visit(s).
2021-05-21 09:49:15.0

<<Page – View Statistics>>