EXPERIMENTATION AND VALIDATION OPENNESS FOR LONGTERM
EVOLUTION OF VERTICAL INDUSTRIES IN 5G ERA AND BEYOND
[H2020 - Grant Agreement No.101016608]

Deliverable D3.2

NetApp Certification Tools and Marketplace development(intermediate)

Editor Alejandro Molina (TID)
Contributors (ATOS, FOGUS, MAG, INTRA)

Version 1.0
Date June 30th, 2022
Distribution PUBLIC (PU)

DISCLAIMER

REVISION HISTORY

Revision	*Date*	*Responsible*	*Comment*
0.1	*March 7th, 2022*	*Alejandro Molina (TID)*	*Edit ToC*
0.2	*April 30 th,2022*	*Alejandro Molina, David Artuñedo (TID)*	*TID inputs*
0.3	*April 30 th,2022*	*Yiannis Karadimas (MAG)*	*MAG inputs*
0.4	*April 30 th,2022*	*Angela Dimitriou (INTRA)*	*INTRA inputs*
0.7	*May 30 th,2022*	*Alejandro Molina, David Artuñedo (INTRA)*	*TID Review*
0.8	*May 30 th,2022*	*Ricardo Marco, Paula Encinar, Sonia Castro (ATOS)*	*ATOS Review*
0.9	*May 30 th,2022*	*Yiannis Karadimas (MAG)*	*MAG Review*
1.0	*June 30 th,2022*	*Alejandro Molina, David Artuñedo (TID)*	*Final Version*

LIST OF AUTHORS

Partner ACRONYM	Partner FULL NAME	Name & Surname
TID	TELEFONICA INVESTIGACIÓN Y DESARROLLO	Javier Garcia David Artuñedo Alejandro Molina
MAG	MAGGIOLI	Yiannis Karadimas
INTRA	INTRASOFT INTERNATIONAL SA	Angela Dimitriou
ATOS	ATOS IT SOLUTIONS ANDSERVICES IBERIA SL	Ricardo Marco Paula Encinar Sonia Castro
FOGUS	FOGUS INNOVATIONS & SERVICES P.C	Dimitris Tsolkas

GLOSSARY

Abbreviations/Acronym	*Description*
3GPP	*3rd Generation Partnership Project*
5GC	*5G Core*
AI	*Artificial Intelligence*
API	*Application Programming Interface*
CAPIF	*Common API Framework*
CI/CD	*Continuous Integration / Continuous Development*
CLI	*Command Line Interface*
COTS	*Commercial off-the-shelf*
CPE	*Customer Premises Equipment*
EC	European Commission
ELCM	*Experiment Life-Cycle Manager*
FoF	Factories of the Future
gNodeB / gNB	*Next Generation (5G) Base Station*
GUI	*Graphical User Interface*
HAT	*Hardware Attached on Top*
HTTP	*Hypertext Transfer Protocol*
IIoT	Industrial Internet of Things
JSON	*JavaScript Object Notation*
JWT	*JSON Web Token*
KPI	*Key Performance Indicator*
LDAP	*Lightweight Directory Access Protocol*
LTE	*Long Term Evolution*
LTE-A	*Long Term Evolution Advanced*
LTE-M	*Long Term Evolution for Machines*
MAC	*Mandatory Access Control*
MANO	*Management and Orchestration*
NB-IoT	*Narrow Band – Internet of Things*
NEF	*Network Exposure Function*
NetApp	*Network Application*
NFV	*Network Function Virtualization*
NSA	*Non StandAlone*
NSD	*Network Service Descriptor*
OAuth	*Open Authorization*
OEM	*Original Equipment Manufacturer*
ONAP	*Open Network Automation Platform*
OSM	*Open Source MANO*
PoP	*Platform to Platform*
PR	*Pull request*
QoS	*Quality of Service*
R&D	*Research and Development*

REST	*Representational State Transfer*
SA	*StandAlone*
SDK	*Software Development Kit*
SLA	*Service Level Agreement*
SME	Small Medium Companies
SSH	*Secure Shell*
SSL	*Secure Sockets Layer*
SSM	*Service Specific Manager*
TSL	*Transport Layer Security*
UE	*User Equipment*
UI	*User Interface*
vAPP	*Vertical Application*
VDU	*Virtual Display Unit*
VNF	*Virtual Network Function*
VulnDB	*Vulnerability Database*
WiFi	*Wireless Fidelity*

EXECUTIVE SUMMARY

EVOLVED-5G responds to the *5G PPP ICT-41-2020 5G innovations for verticals with third party services* call, whose main goal is to deliver enhanced experimentation facilities on top of which third party experimenters (e.g., SMEs or any service provider and target vertical users) will have the opportunity to test their applications.

The EVOLVED-5G project realises this vision by encouraging the creation of a NetApp ecosystem revolving around a 5G facility which will provide the tools and processes for the development, verification, validation, and certification of NetApps as well as their smooth running on top of actual 5G network infrastructures, and mechanisms for market releasing.

The primary objective of this deliverable is to present the implementation status of the tools, activities and methodologies that are materializing for EVOLVED-5G Certification Process and Environment, targeting implementation, efficiency, and security aspects of its architectural components that define NetApp certification Process and the Marketplace to release the certified NetApps.

This document describes the results of task ”T3.4-NetApp Certification Tools and Marketplace development” for first period of the project, providing a first description of the selected tools for the design of the automated Certification Process, the Test engine for running certification tests, and the description of tests that will be part of the Certification process. Commercial on the shelf tools have been selected, when possible, to cover the Certification requirements.

First section of the document describes the Deliverable target audience, objectives and structure.

Sections 2, 3 and 4 focus on Certification Environment, including the description of the CICD toolset selected, the Certification Tools selected to fulfill certification requirements, and provides an overview of the Certification Process design.

Finally, Sections 5 and 6 focus on Marketplace, describing the development approach and the designed integration of the Marketplace with the Open Repository.

There will be an updated version of this deliverable, to be released towards the end of the Project, with the final implementation details carried out during the rest of the Project.

TABLE OF CONTENTS

LIST OF FIGURES

1 INTRODUCTION

1.1 SCOPE

The scope of this deliverable is to expose implementation details for both the Evolved-5G Certification Environment and Tools and the Marketplace. Its intention is to be accessible to a broad variety of research individuals and communities.
The target audiences are described below:

- **Project Consortium**: To validate that all objectives and proposed technological advancements have been analysed and to ensure that, through the proposed implementations, further work can be derived. Furthermore, the deliverable sets a common understanding among the consortium with regards to:
    - The implementation details of the Certification Environment related to the NetApp lifecycle in the context of the EVOLVED-5G project, including tools and technologies to be used.
    - The implementation details of the Marketplace, the final step of the NetApp lifecycle in the context of the EVOLVED-5G project.
- **Industry 4.0/Industry 4.0 developers and Factories of the Future (FoF) vertical groups**: To set a common understanding of the technologies and the design principles that underline the Certification Process design and its implementation, along with the Marketplace release process.
- **Other vertical industries and groups:** To seek impact on other 5G-enabled vertical industries and groups in the long run. Indeed, all the architectural components of the facility are being designed to secure interoperability beyond vendor-specific implementation and across multiple domains. The same categorization can be applicable beyond the Industry 4.0 domain.
- **The scientific audience, general public, and the funding EC Organisation**: To document the work performed by the project and justify the effort reported for all relevant activities. The scientific audience can also get an insight of the design approach and underlying components behind the Evloved-5G Certification Process and Marketplace implementation.

1.2 OBJECTIVES

This deliverable is the second of a series of four WP3 reports to be delivered by the project consortium during its 36-month work plan. The primary objective of this second report, entitled “D3.2 NetApp Certification Tools and Marketplace development (Intermediate)”, is to present the implementation status of the tools, activities and methodologies that are materializing for EVOLVED-5G Certification Process and Environment, targeting implementation, efficiency, and security aspects of its architectural components that define NetApp certification Process and the Marketplace to release the certified NetApps. This document describes the initial results of task ”T3.4-NetApp Certification Tools and Marketplace development”, providing a first description of the selected tools for the design of the automated Certification Process, the Test engine for running certification tests, and the description of tests that will be part of the Certification process. Commercial on the shelf tools, such as Sonaqube, Trivy or Debriked have been selected, when possible, to cover the Certification requirements listed in D2.1 [1].

There will be an updated version of this deliverable, Deliverable 3.4 [2] to be released towards the end of the Project, with the final implementation details carried out during the rest of the Project.

1.3 Structure

The deliverable is organized in the following manner:

- Section 1. Introduction: This section describes the Deliverable target audience, objectives and structure.
- Section 2. Certification Environment: This section describes the Certification Environment focusing on the CICD toolset with a collection of software industry leading tools for automation.
- Section 3. Certification : This section describes the specific tools that have been developed by Evolved-5G project for NetApp Certification, namely the CAPIF (Common API Framework) Core Function and NEF (Network Exposure Function) Emulator, and commercial of the shelf tools selected to cover certification requirements, such as SonarQube, Trivy and Debriked.
- Section 4. Certification Process: This section provides a high-level description of the Certification process, including the multiple steps that are being implemented in the Certification Pipeline to automate the Certification Process.
- Section 5. Marketplace: This section describes the EVOLVED-5G Marketplace, developed specifically for this project. This platform is used to publish NetApps after Certification takes place.
- Section 6. Marketplace and Open Repository Integration: This section describes the approach adopted for the integration of the Open Repository and the Marketplace, so that the Certified NetApps can be published in the Marketplace.

2 CERTIFICATION ENVIRONMENT

This section enumerates the tools selected for building the certification environment. This environment enables the construction of the certification process described in section 4. The environment is built following software industry best practises, as NetApps as fundamentally software components. TID has contributed its own CICD system and tools to create an Evolved-5G area where all the tools are being integrated to build the certification process and execute the NetApp certifications.

This section introduces CICD TID´s solution and describes CICD principles, to then describe the main tools that articulate the CICD system: the Open Repository to store artifacts, Jenkins for Automation, Terraform to manage the Infrastructure as Code, Robot Framework to build automated test and finally, it describes the execution platforms where NetApps will be deployed, and tests will be executed.

2.1 CICD SUPPORT FOR CERTIFICATION PROCESS

CICD (Continuous Integration/Continuous Delivery)[4] is a method for delivering applications frequently. For this, automation is applied in the stages of application development. The key concepts attributed to this method are continuous integration, delivery and deployment.

Continuous delivery represents changes made by the developer in an application, which are automatically tested and loaded into a repository such as GitHub. In this repository, the operations team can deploy these changes in an active production environment. This solves the problem of poor visibility and communication between business and development teams. For this, the purpose of continuous delivery is to ensure minimal effort in the deployment of new code.

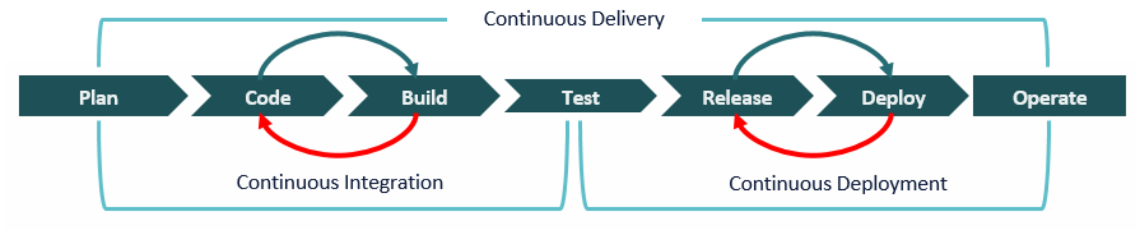


Figure 1 CICD overview

These are some benefits associated to the use of this methodology:

- **Automation of procedures:** Automating software delivery process procedures reduces the need for knowledgeable resources in the use of various tools, significantly decreases the likelihood of human failure, enables continuous execution of such procedures, and improves the overall maturity of the process.

- **Quality Anticipation:** Strengthen the culture of early quality practice by increasing the level of test execution by following the logic of the test pyramid application (more unit, integration, and automated tests performed earlier in the project) to reduce leaks of the test phase.

- **Team Performance Improvement:** Applying CICD practices help to reduce effort in repetitive activities, and also the need for rework due to failures, making the process more orderly and predictable.

EVOLVED-5G has adopted this methodology for the development of NetApps, and also for tools developed inside the project to support Certification, such as CAPIF Core Function, NEF Services and Marketplace.

The NetApp lifecycle in EVOLVED-5G has three phases: Verification, Validation and Certification. Each of these phases will have a separate process supported by CICD tools.

The following component stack is used for EVOLVED-5G CICD, to support the testing processes mentioned above.

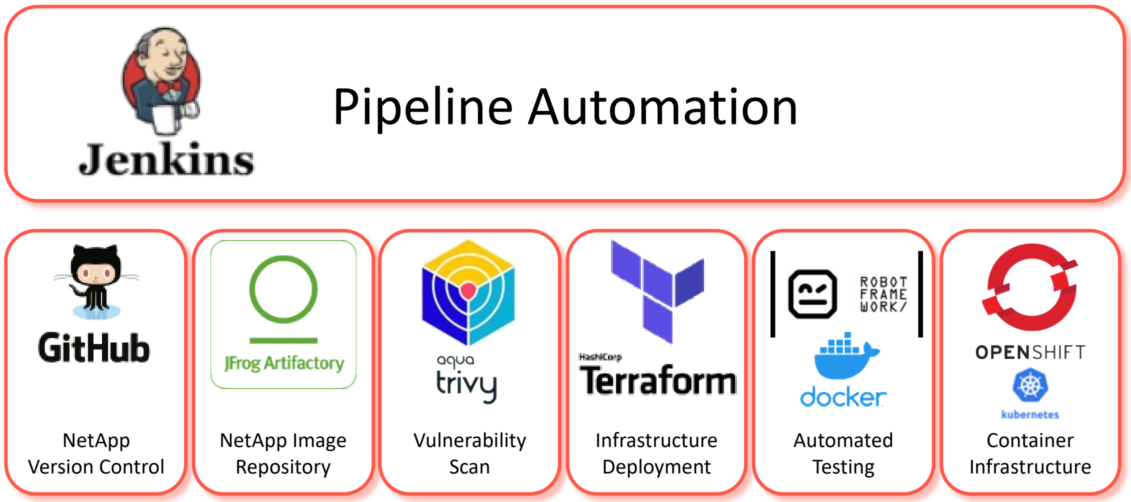
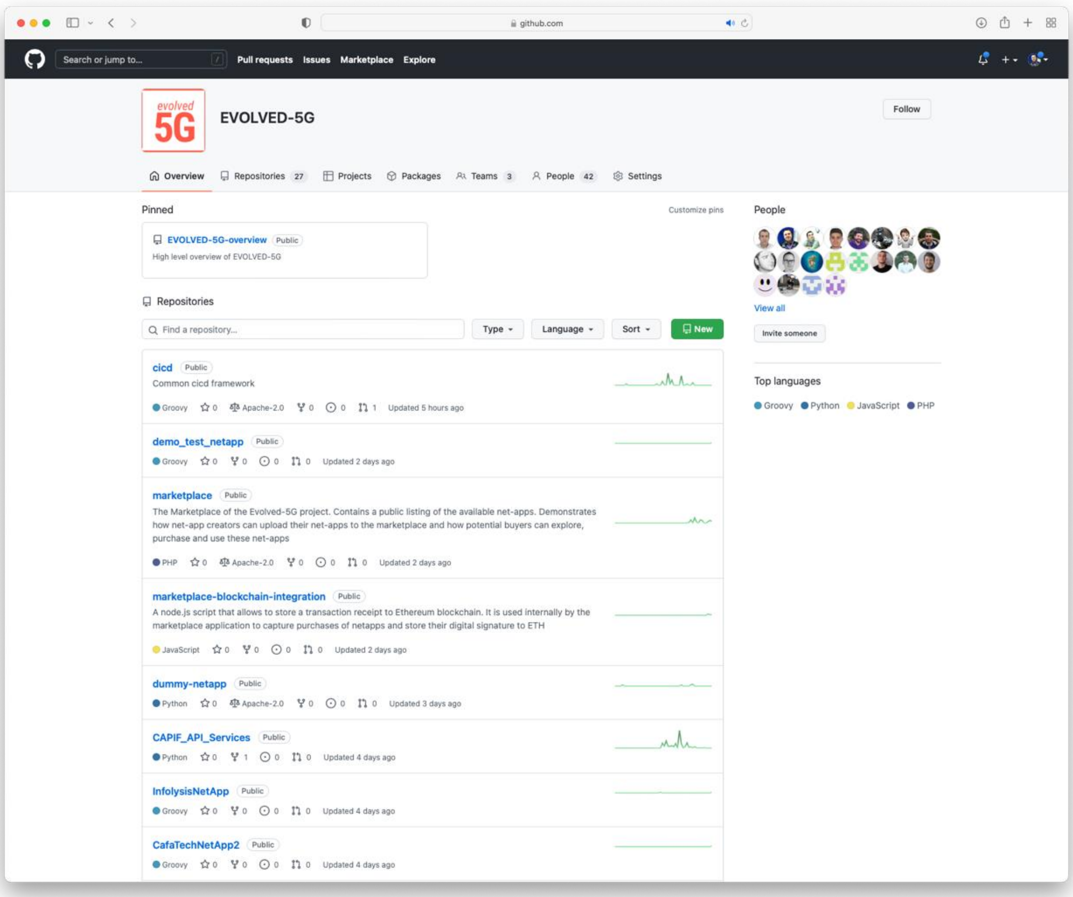


Figure 2 CICD Toolset used in EVOLVED-5G

2.2 Open Repository

EVOLVED-5G uses GitHub [4] as control version tool, along with git well-known best practices. GitHub is a cloud-managed code version control platform using Git. GitHub is widely used by programmers to publicize their work or for other programmers to contribute to projects, as well as to promote easy communication through features that report issues or merge remote repositories (issues, pull request, etc.).

This is an actual GitHub screenshot of the repositories of the project:


github.com

Search or jump to... Pull requests Issues Marketplace Explore

EVOLVED-5G

Follow

Overview · Repositories 27 · Projects · Packages · Teams 3 · People 42 · Settings

Pinned — Customize pins

EVOLVED-5G-overview (Public)
High level overview of EVOLVED-5G

Repositories

Find a repository... Type Language Sort New

cicd (Public)
Common cicd framework
Groovy ☆ 0 Apache-2.0 0 0 1 Updated 5 hours ago

demo_test_netapp (Public)
Groovy ☆ 0 0 0 0 Updated 2 days ago

marketplace (Public)
The Marketplace of the Evolved-5G project. Contains a public listing of the available net-apps. Demonstrates how net-app creators can upload their net-apps to the marketplace and how potential buyers can explore, purchase and use these net-apps
PHP ☆ 0 Apache-2.0 0 0 0 Updated 2 days ago

marketplace-blockchain-integration (Public)
A node.js script that allows to store a transaction receipt to Ethereum blockchain. It is used internally by the marketplace application to capture purchases of netapps and store their digital signature to ETH
JavaScript ☆ 0 0 0 0 Updated 2 days ago

dummy-netapp (Public)
Python ☆ 0 Apache-2.0 0 0 0 Updated 3 days ago

CAPIF_API_Services (Public)
Python ☆ 0 1 0 0 Updated 4 days ago

InfolysisNetApp (Public)
Groovy ☆ 0 0 0 0 Updated 4 days ago

CafaTechNetApp2 (Public)
Groovy ☆ 0 0 0 0 Updated 4 days ago

People
View all
Invite someone

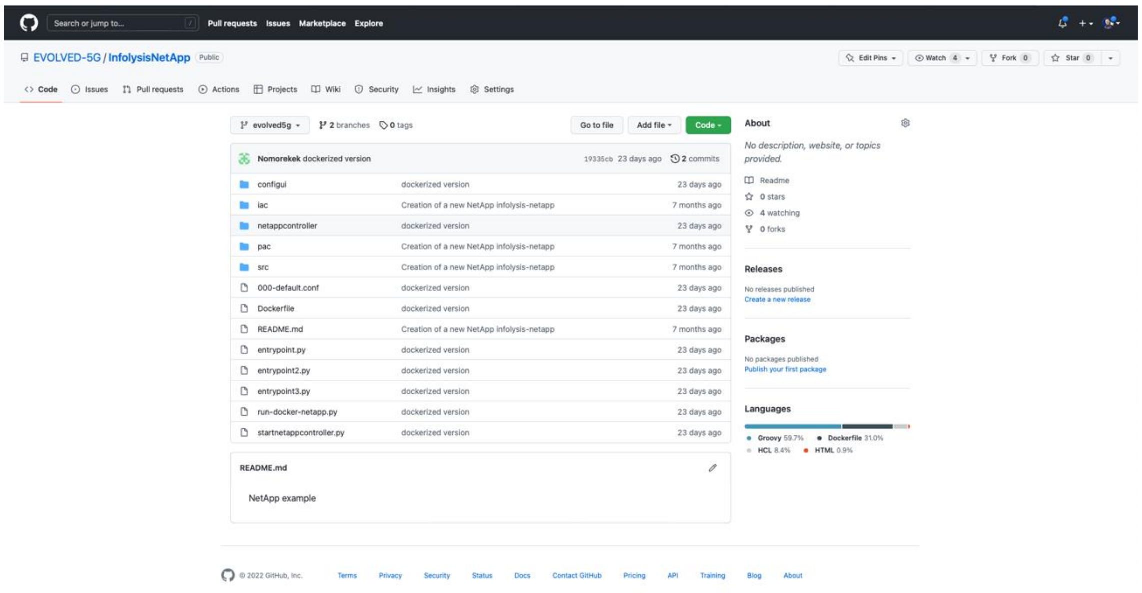
Top languages
● Groovy ● Python ● JavaScript ● PHP

Figure 3 EVOLVED-5G Github repository

EVOLVED5G project uses a GitFlow-like workflow approach, defining a branching model and providing a robust structure to manage the code repositories. GitFlow is an abstract idea of a Git workflow; this means that it determines what types of branches are configured and how to combine them.

2.2.1 Image Management

Evolved5G uses Docker [5] as its Container image builder tool. We use Docker to create the binary images of the NetApps based in the Dockerfile information published in the NetApp repository.



Nomorekek dockerized version 19335cb 23 days ago 2 commits

configui	dockerized version	23 days ago
iac	Creation of a new NetApp infolysis-netapp	7 months ago
netappcontroller	dockerized version	23 days ago
pac	Creation of a new NetApp infolysis-netapp	7 months ago
src	Creation of a new NetApp infolysis-netapp	7 months ago
000-default.conf	dockerized version	23 days ago
Dockerfile	dockerized version	23 days ago
README.md	Creation of a new NetApp infolysis-netapp	7 months ago
entrypoint.py	dockerized version	23 days ago
entrypoint2.py	dockerized version	23 days ago
entrypoint3.py	dockerized version	23 days ago
run-docker-netapp.py	dockerized version	23 days ago
startnetappcontroller.py	dockerized version	23 days ago

README.md

NetApp example

About

No description, website, or topics provided.

Readme
0 stars
4 watching
0 forks

Releases

No releases published
Create a new release

Packages

No packages published
Publish your first package

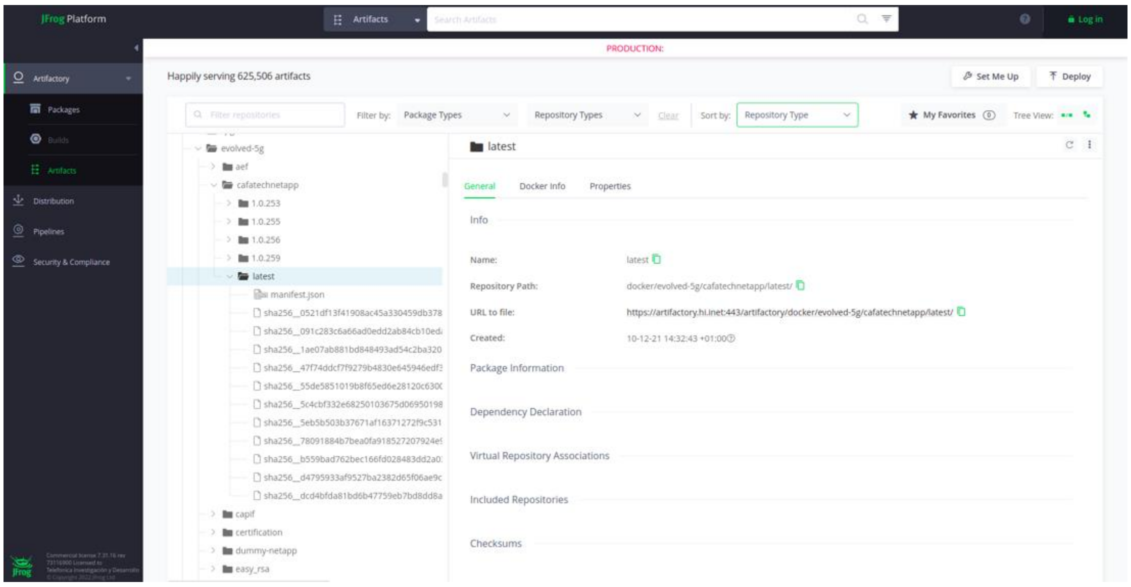
Languages

● Groovy 59.7% ● Dockerfile 31.0% ● HCL 8.4% ● HTML 0.9%

Figure 4 Dockerfile located in NetApp repository

For storing the binary images of the NetApps, two solutions will be used simultaneously: Artifactory and AWS Elastic Container Registry ECR [6]. Artifactory is the repository inside CICD environment and therefore is not accessible from Internet. To enable deploying NetApps in Athens and Malaga platforms, a private repository has been replicated in AWS with public IP access.

JFrog Artifactory [8] is a repository manager that supports all available software package types, enabling automated continuous integration, deployment and delivery. Artifactory is used to store and manage all the NetApps docker container images and all generated artefacts. Artifactory usually stores individual application components that can be put together at a later stage, thus allowing to break bigger apps in smaller chunks, getting more efficient building processes and tracking building errors with ease.



Happily serving 625,506 artifacts

```
evolved-5g
  aef
  cafatechnetapp
    1.0.253
    1.0.255
    1.0.256
    1.0.259
    latest
      manifest.json
      sha256__0521df13f41908ac45a330459db378
      sha256__091c283c6a66ad0edd2ab84cb10ed
      sha256__1ae07ab881bd848493ad54c2ba320
      sha256__47f74ddcf7f9279b4830e645946edf3
      sha256__55de5851019b8f65ed6e28120c630
      sha256__5c4cbf332e68250103675d06950198
      sha256__5eb5b503b37671af16371272f9c531
      sha256__78091884b7bea0fa918527207924e
      sha256__b559bad762bec166fd028483dd2a0
      sha256__d4795933af9527ba2382d65f06ae9c
      sha256__dcd4bfda81bd6b47759eb7bd8dd8a
  capif
  certification
  dummy-netapp
  easy_rsa
```

latest

Info

Name:	latest
Repository Path:	docker/evolved-5g/cafatechnetapp/latest/
URL to file:	https://artifactory.hi.inet:443/artifactory/docker/evolved-5g/cafatechnetapp/latest/
Created:	10-12-21 14:32:43 +01:00

Package Information

Dependency Declaration

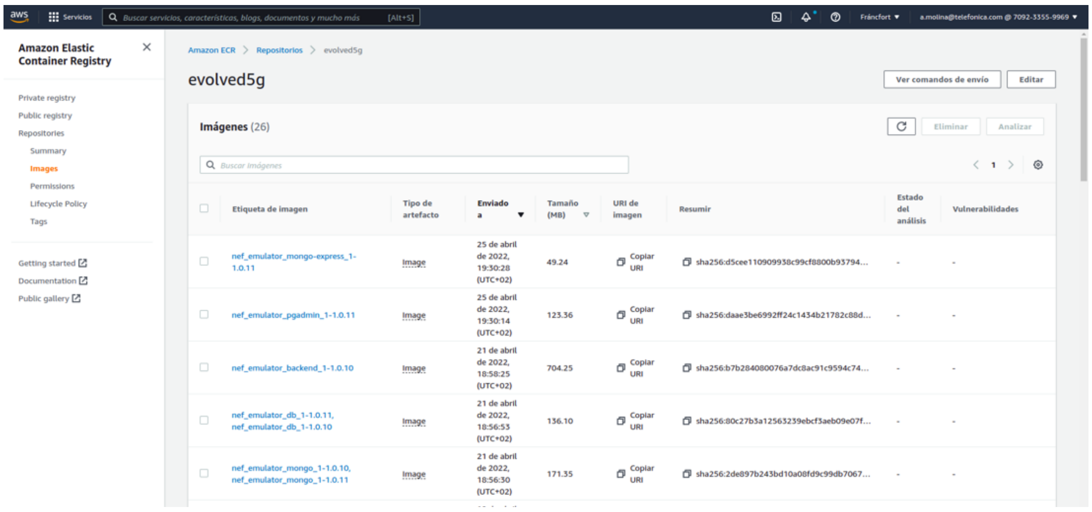
Virtual Repository Associations

Included Repositories

Checksums

Figure 5 JFrog Artifactory screenshot

AWS ECR is an AWS solution to easily store and share container images. AWS ECR has numerous advantages, but the most important one, in our case, is that we can push and pull images from anywhere as it is available in the Public Cloud.



evolved5g

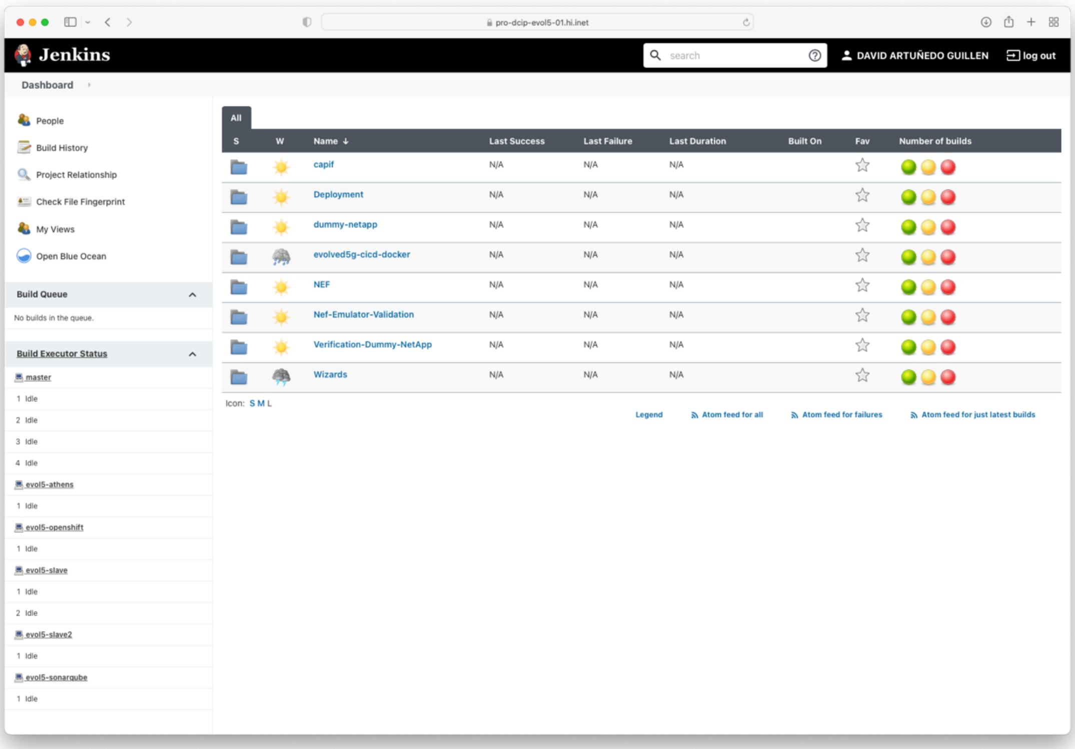
Imágenes (26)

Etiqueta de imagen	Tipo de artefacto	Enviado a	Tamaño (MB)	URI de imagen	Resumir	Estado del análisis	Vulnerabilidades
nef_emulator_mongo-express_1-1.0.11	Image	25 de abril de 2022, 19:30:28 (UTC+02)	49.24	Copiar URI	sha256:d5cee110909938c99cf8800b93794...	-	-
nef_emulator_pgadmin_1-1.0.11	Image	25 de abril de 2022, 19:30:14 (UTC+02)	123.36	Copiar URI	sha256:daae3be6992ff24c1434b21782c88d...	-	-
nef_emulator_backend_1-1.0.10	Image	21 de abril de 2022, 18:58:25 (UTC+02)	704.25	Copiar URI	sha256:b7b284080076a7dc8ac91c9594c74...	-	-
nef_emulator_db_1-1.0.11, nef_emulator_db_1-1.0.10	Image	21 de abril de 2022, 18:56:53 (UTC+02)	136.10	Copiar URI	sha256:80c27b3a12563239ebcf3aeb09e07f...	-	-
nef_emulator_mongo_1-1.0.10, nef_emulator_mongo_1-1.0.11	Image	21 de abril de 2022, 18:56:30 (UTC+02)	171.35	Copiar URI	sha256:2de897b243bd10a08fd9c99db7067...	-	-

Figure 6 AWS Elastic Container Registry

2.3 AUTOMATION TOOL

Evolved5G uses Jenkins [7] as its pipeline automation tool. Jenkins is a free and open-source automation server, which helps to automate continuous integration and facilitating continuous delivery. It supports version control tools, including Git. Builds can be triggered by various means, for example by commit in a version control system, by scheduling via a cron-like mechanism and by requesting a specific build URL. It can also be triggered after the other builds in the queue have completed. Jenkins can be extended with plugins in order to improve or add new functionalities to the Jenkins' instance. In Evolved5G, for example, we have added SonarQube and Roboframework plugins to enhance their cross-functions with Jenkins and generate reports inside of each execution.


pro-dcip-evol5-01.hi.inet

Jenkins

DAVID ARTUÑEDO GUILLEN

Dashboard

People
Build History
Project Relationship
Check File Fingerprint
My Views
Open Blue Ocean

Build Queue

No builds in the queue.

Build Executor Status

master
1 Idle
2 Idle
3 Idle
4 Idle
evol5-athens
1 Idle
evol5-openshift
1 Idle
evol5-slave
1 Idle
2 Idle
evol5-slave2
1 Idle
evol5-sonarqube
1 Idle

All

S	W	Name ↓	Last Success	Last Failure	Last Duration	Built On	Fav	Number of builds
		capif	N/A	N/A	N/A		☆	🟢🟡🔴
		Deployment	N/A	N/A	N/A		☆	🟢🟡🔴
		dummy-netapp	N/A	N/A	N/A		☆	🟢🟡🔴
		evolved5g-cicd-docker	N/A	N/A	N/A		☆	🟢🟡🔴
		NEF	N/A	N/A	N/A		☆	🟢🟡🔴
		Nef-Emulator-Validation	N/A	N/A	N/A		☆	🟢🟡🔴
		Verification-Dummy-NetApp	N/A	N/A	N/A		☆	🟢🟡🔴
		Wizards	N/A	N/A	N/A		☆	🟢🟡🔴

Icon: S M L

Legend   Atom feed for all   Atom feed for failures   Atom feed for just latest builds

Figure 7 Evolved5G Jenkins Dashboard

Pipeline configuration in Jenkins enables to define the whole NetApp lifecycle defined in Evolved5G. The pipeline plugin was built with requirements for a flexible, extensible, and script based Continuous Deployment workflow capability in mind. Accordingly, pipeline functionality is:

- Durable: Pipelines can survive both planned and unplanned restarts of your Jenkins master.
- Pausable: Pipelines can optionally stop and wait for human input or approval before completing the jobs for which they were built.
- Versatile: Pipelines support complex real-world CD requirements, including the ability to fork or join, loop, and work in parallel with each other.
- Efficient: Pipelines can restart from any of several saved checkpoints.
- Extensible: The Pipeline plugin supports custom extensions to its DSL (domain scripting language) and multiple options for integration with other plugins.

Below is an example of one continuous delivery scenario enabled by the pipeline plugin:

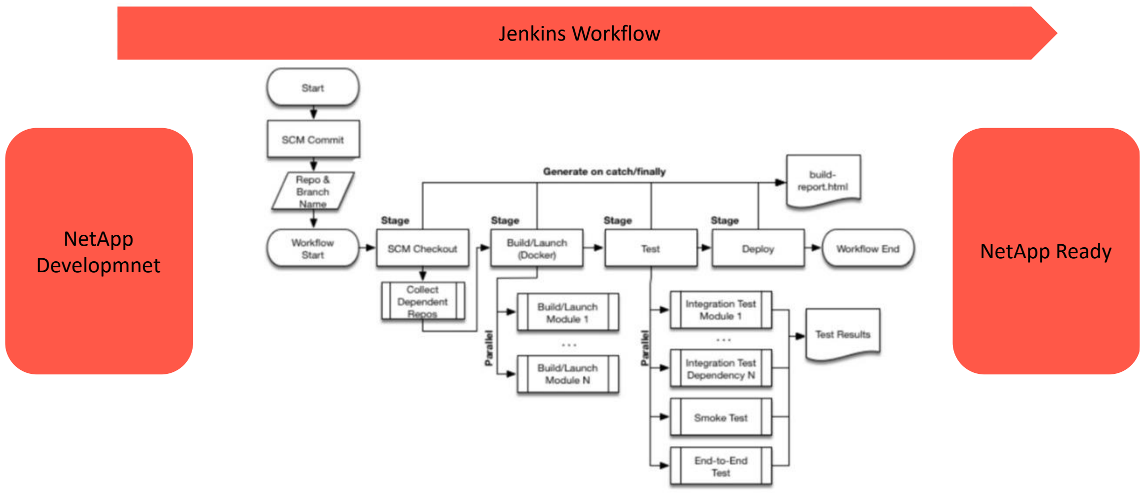


Figure 8 Jenkins Flow for NetApps

Pipelines are Jenkins jobs, typically built with simple text scripts that use a Pipeline DSL (Domain-Specific Language) based on the Groovy programming language. Pipelines leverage the power of multiple steps to execute both simple and complex tasks according to parameters established. Once created, pipelines can build code and orchestrate the work required to drive applications from commit to delivery.

Pipeline terms such a “step,” “node,” and “stage” are a subset of the vocabulary used for Jenkins in general.

- Step: A “step” (often called a “build step”) is a single task that is part of a sequence. Steps tell Jenkins what to do.
- Node: In Jenkins generally, a “node” means any computer that is part of your Jenkins installation, whether that computer is used as a master or as an agent. In pipeline coding contexts, as opposed to Jenkins generally, a “node” is a step that does two things, typically by enlisting help from available executors on agents:
    - Schedules the steps contained within it to run by adding them to the Jenkins build queue (so that as soon as a slot is free on a node, the appropriate steps run).
    - Creates a workspace, meaning a file directory specific to a particular job, where resource-intensive processing can occur without negatively impacting your pipeline performance. Workspaces last for the duration of the tasks assigned to them.
- Stage: A “stage” is a step that calls supported APIs. Pipeline syntax is comprised of stages. Each stage can have one or more build steps within it.

Familiarity with Jenkins terms such as “master,” “agent,” and “executor” also helps with understanding how pipelines work. These terms are not specific to pipelines:

- Master - A “master” is the basic installation of Jenkins on a computer; it handles tasks for your build system. Pipeline scripts are parsed on masters, and steps wrapped in node blocks are performed on available executors.

- Agent - An “agent” (formerly "slave") is a computer set up to offload particular projects from the master. Your configuration determines the number and scope of operations that an agent can perform. Operations are performed by executors.

- Executor - An “executor” is a computational resource for compiling code. It can run on master or agent machines, either by itself or in parallel with other executors. Jenkins assigns a (Java) thread to each executor.

A multi-branch pipeline project type enables you to configure different jobs for different branches of the same project. In a multi-branch pipeline configuration, Jenkins automatically discovers, manages, and executes jobs for multiple source repositories and branches. This eliminates the need for manual job creation and management, as would otherwise be necessary when, for example, a developer adds a new feature to an existing product. Multi-branch pipelines also enable you to stop or suspend jobs automatically if circumstances make that appropriate.

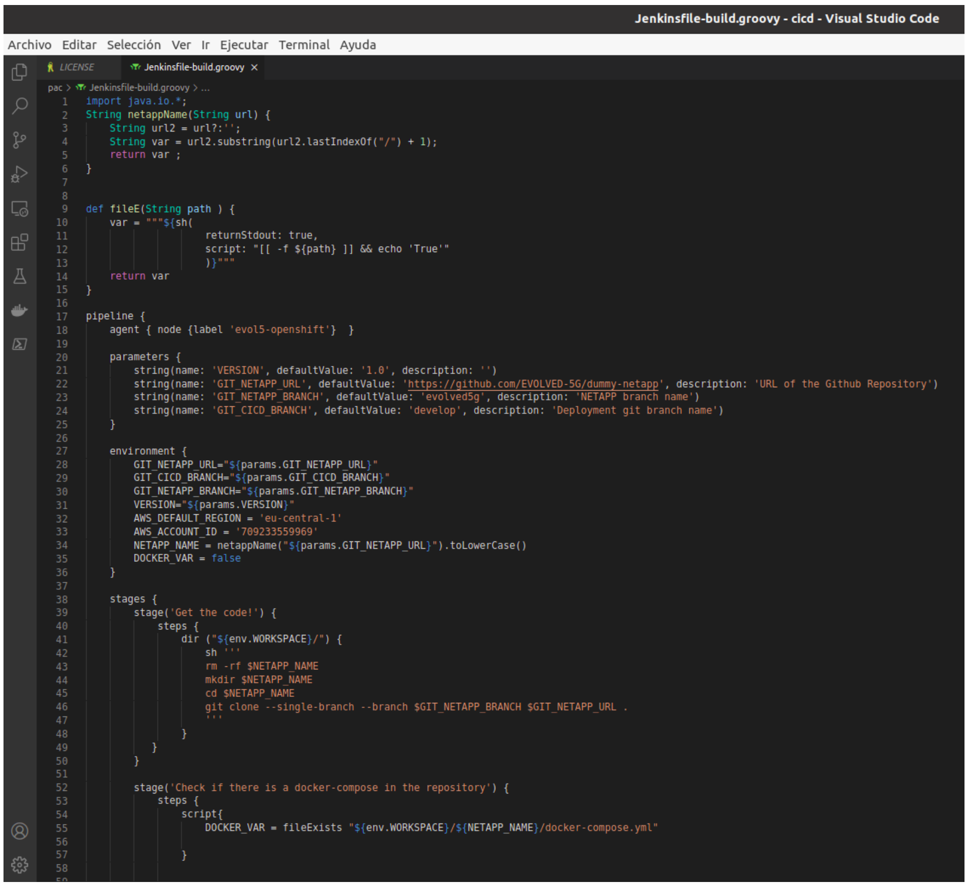
A multi-branch pipeline project always includes a Jenkins-file in its repository root. Jenkins automatically creates a sub-project for each branch that it finds in a repository with a Jenkins-file. Multi-branch pipelines use the same version control as the rest of your software development process. This “pipeline as code” approach has the following advantages:

- You can modify pipeline code without special editing permissions.
- Finding out who changed what and why no longer depends on whether developers remember to comment their code changes in configuration files.
- Version control makes the history of changes to code readily apparent.

Complex pipelines would be cumbersome to write and maintain if you could only do that in the text area provided by the Jenkins job configuration page. Accordingly, there is an option, called “Pipeline Script from SCM”, to load those scripts into Jenkins, enabled by the workflow-scm-step plugin, which is one of the plugins that the Pipeline plugin depends on and automatically installs.

Loading pipeline scripts from another source leverages the idea of “pipeline as code,” and allow developers to maintain that source using version control and standalone Groovy editors. When the designated repository is updated, a new build will be triggered, as long as the job is configured with an SCM polling trigger.

A Groovy pipeline configuration example is shown below:


Jenkinsfile-build.groovy - cicd - Visual Studio Code
Archivo Editar Selección Ver Ir Ejecutar Terminal Ayuda
LICENSE
Jenkinsfile-build.groovy
pac > Jenkinsfile-build.groovy > ...

```groovy
import java.io.*;
String netappName(String url) {
    String url2 = url?:'';
    String var = url2.substring(url2.lastIndexOf("/") + 1);
    return var ;
}


def fileE(String path ) {
    var = """${sh(
                    returnStdout: true,
                    script: "[[ -f ${path} ]] && echo 'True'"
                    )}"""
    return var
}

pipeline {
    agent { node {label 'evol5-openshift'}  }

    parameters {
        string(name: 'VERSION', defaultValue: '1.0', description: '')
        string(name: 'GIT_NETAPP_URL', defaultValue: 'https://github.com/EVOLVED-5G/dummy-netapp', description: 'URL of the Github Repository')
        string(name: 'GIT_NETAPP_BRANCH', defaultValue: 'evolved5g', description: 'NETAPP branch name')
        string(name: 'GIT_CICD_BRANCH', defaultValue: 'develop', description: 'Deployment git branch name')
    }

    environment {
        GIT_NETAPP_URL="${params.GIT_NETAPP_URL}"
        GIT_CICD_BRANCH="${params.GIT_CICD_BRANCH}"
        GIT_NETAPP_BRANCH="${params.GIT_NETAPP_BRANCH}"
        VERSION="${params.VERSION}"
        AWS_DEFAULT_REGION = 'eu-central-1'
        AWS_ACCOUNT_ID = '709233559969'
        NETAPP_NAME = netappName("${params.GIT_NETAPP_URL}").toLowerCase()
        DOCKER_VAR = false
    }

    stages {
        stage('Get the code!') {
            steps {
                dir ("${env.WORKSPACE}/") {
                    sh '''
                    rm -rf $NETAPP_NAME
                    mkdir $NETAPP_NAME
                    cd $NETAPP_NAME
                    git clone --single-branch --branch $GIT_NETAPP_BRANCH $GIT_NETAPP_URL .
                    '''
                }
            }
        }

        stage('Check if there is a docker-compose in the repository') {
            steps {
                script{
                    DOCKER_VAR = fileExists "${env.WORKSPACE}/${NETAPP_NAME}/docker-compose.yml"

                }
```

Figure 9 Groovy language example

When a Jenkins job runs, workload of the single-node Jenkins will take master resources and wait for the queued process to finish. This is time-consuming and sometimes, bigger build task may break the Jenkins server. Even in some cases, a single build job needs a different environment for different tasks. To overcome this issue, Jenkins has a feature called "Distributed Build" and followed by the Jenkins Master-Slave Setup. Typically, the machine where the complete Jenkins installation is located will be Jenkins master. On the slave machine, a runtime program called "Agent" will be installed. Installing Agent will not install Jenkins completely, but this agent will run on Java Virtual Machine (JVM) [9]. It is capable enough to run the subtask or main task of the Jenkins in a dedicated executor.

A Jenkins system like the aforementioned would be like the one illustrated below:

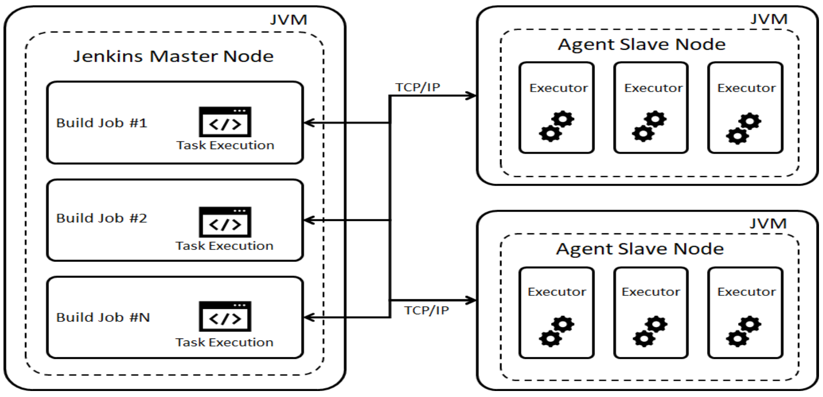


Figure 10 Jenkins Master/Slave architecture

Any number of agent nodes can be configured, and master node can be configured to decide which task or job should run which agent and how many executors the agents can have. The communication between master and Jenkins slave nodes is bi-directional and it is happening with TCP/IP connection protocol.

2.4 Infrastructure Deployment Tool

Terraform is an open-source infrastructure as code software tool created by HashiCorp. Users define and provide data center infrastructure using a declarative configuration language known as HashiCorp Configuration Language (HCL), or optionally JSON.

Using infrastructure as code enables the automation of the provisioning of NetApp infrastructure including servers and containers, databases, networking policies and almost any other aspect. This is critical to make the certification process repeatable and deterministic, as all the required infrastructure for Certification will be deployed and set up from scratch for every certification cycle.

Terraform abstract the cloud provider using Terraform Providers. The infrastructure as code written in declarative language is translated to a specific cloud provider (target APIs) using the provider for this cloud provider.

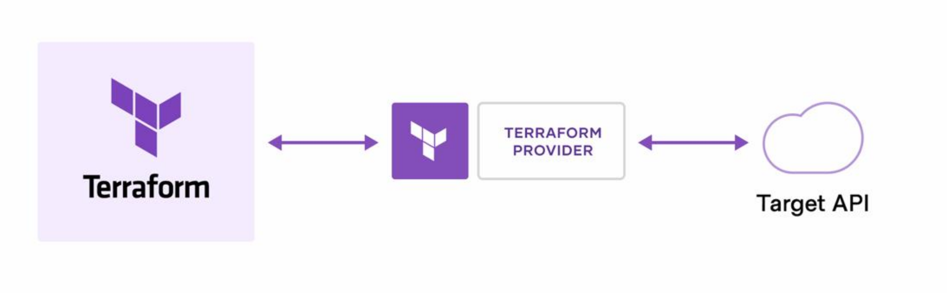
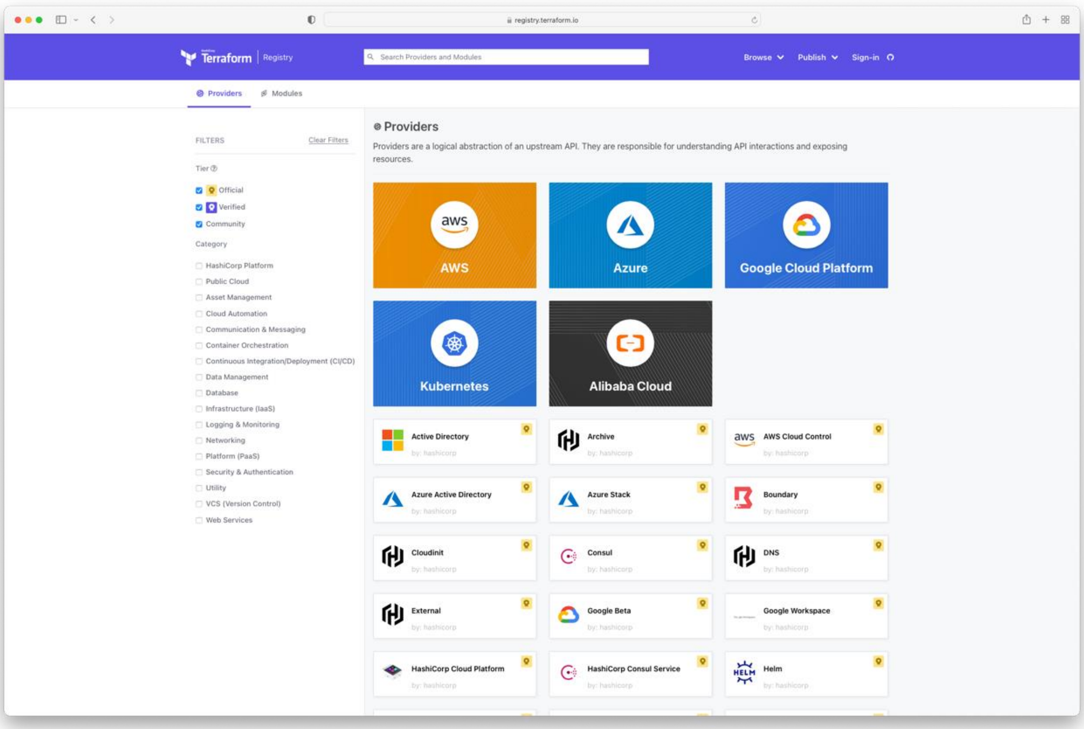


Figure 11 Terraform Provider concept

Terraform offers a large catalogue of cloud providers as shown in the following picture:



Providers are a logical abstraction of an upstream API. They are responsible for understanding API interactions and exposing resources.

Figure 12 Terraform Providers catalogue screenshot

Evolved5G uses Terraform to create, build and deploy the infrastructure required for NetApp certification every time a Certification process is launched by using the Certification Pipeline. As explained later in section 2.6, Evolved5G uses three different environments for deploying NetApps, and Terraform let us abstract the underlying infrastructure of each of the environments, meaning, we use the same declarative infrastructure as code description to deploy NetApps in all the environments.

2.5 Robot Framework Testing Engine

Robot Framework [10] as a generic open-source automation framework. It can be used for test automation and robotic process automation (RPA). Robot Framework is supported by Robot Framework Foundation.

Robot Framework is open and extensible. Robot Framework can be virtually integrated with any other tool (e.g., Jenkins) to create powerful and flexible automation solutions. Robot Framework is free to use without licensing costs.

Robot Framework has an easy syntax, utilizing human-readable keywords. Its capabilities can be extended by libraries implemented with Python, Java or many other programming languages. It has a rich ecosystem around it, consisting of libraries and tools that are developed as separate projects.

Evolved5G uses Robot Framework to automate some of the tests required for Certification (see Figure 20 Result of Robot framework CAPIF Functional Tests). Evolved5G packs all components required to execute Robot Framework tests in a Docker image that Evolved5G CICD deploys and executes using Jenkins to collect tests results. These tests results are integrated into Jenkins pipelines flow.

Examples of these tests are the CAPIF Core Function Functional tests and NEF Functional tests used in Certification as described in section 4.2.

2.6 Container Execution Platforms

Evolved5G uses several Container execution platforms to verify, validate and certify NetApps. Evolved5G CICD Environment incorporates OpenShift container platform to allow CICD to deploy and execute tests internally in the CICD platform.
Additionally, Evolved5G CICD platform connects to two other container execution environments embedded in 5G Platforms in Athens and Málaga.

2.6.1 OpenShift 4.0

Evoled5G CICD Platform integrates OpenShift 4.0 [11] container solution. Evolved5G CICD Platform includes a Jenkins slave to interact with Open Shift infrastructure. Thus, Jenkins using Terraform can deploy, manage and destroy containers in Open Shift execution platform.

Following the master-slave Jenkins design pattern, the architecture of this scenario with OpenShift is the following:

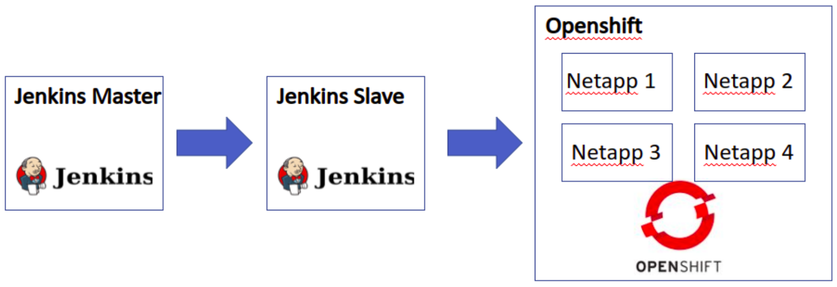


Figure 13 Jenkins slave for OpenShift 4.0

As we can see in Figure 13 , there is direct connectivity between the Jenkins slave and OpenShift.

OpenShift architecture is displayed in the following figure:

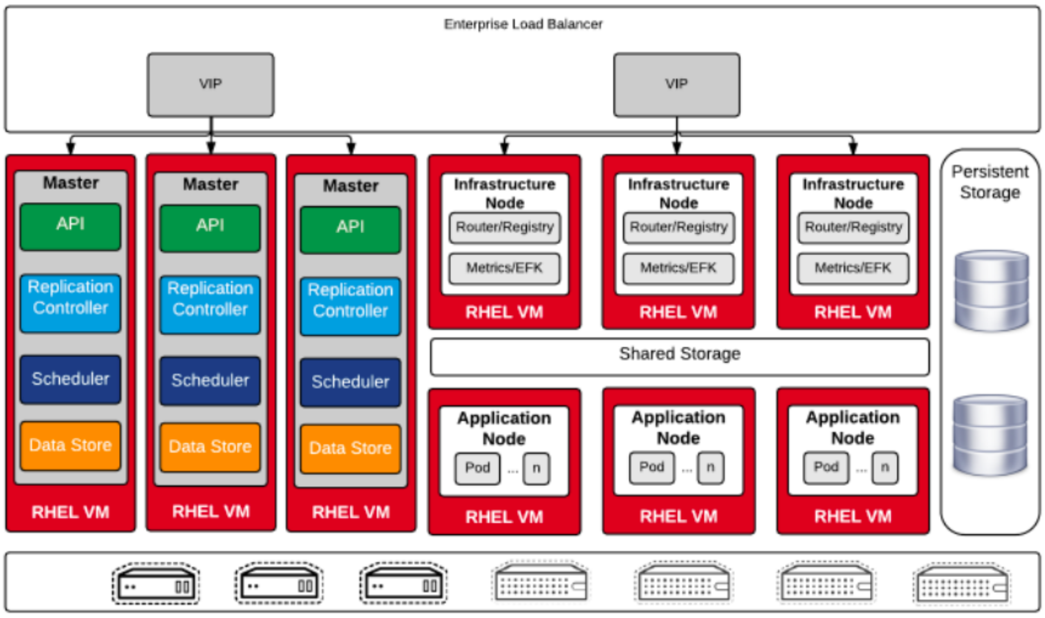
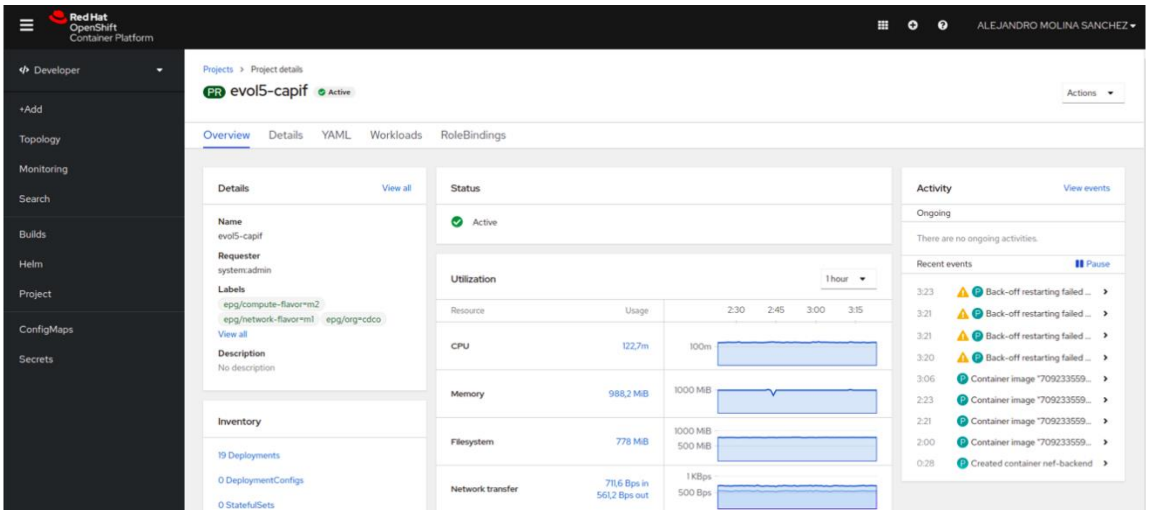


Figure 14 OpenShift architecture [14]

The reason for using version 4.0 of OpenShift is mainly for the simplicity to expose services and assign them to public URLs.

Figure 15 OpenShift Console screenshot displays the user interface that appears with all the information about deployments in OpenShift 4.0.



PR evol5-capif Active

Details

Name
evol5-capif
Requester
system:admin
Labels
epg/compute-flavor=m2
epg/network-flavor=m1 epg/org=cdco
View all
Description
No description

Inventory

19 Deployments
0 DeploymentConfigs
0 StatefulSets

Status

Active

Utilization 1 hour

Resource	Usage
CPU	122,7m
Memory	988,2 MiB
Filesystem	778 MiB
Network transfer	711,6 Bps in 561,2 Bps out

Activity

Ongoing
There are no ongoing activities.

Recent events

3:23	Back-off restarting failed ...
3:21	Back-off restarting failed ...
3:21	Back-off restarting failed ...
3:20	Back-off restarting failed ...
3:06	Container image "709233559...
2:23	Container image "709233559...
2:21	Container image "709233559...
2:00	Container image "709233559...
0:28	Created container nef-backend

Figure 15 OpenShift Console screenshot

2.6.2 Malaga Platform Kubernetes

Malaga uses Kubernetes in a multi-master architecture with multiple workers. In this configuration and set up there will not be service exposition to public IPs in this platform. Access to deployed containers will be done using VPN access.

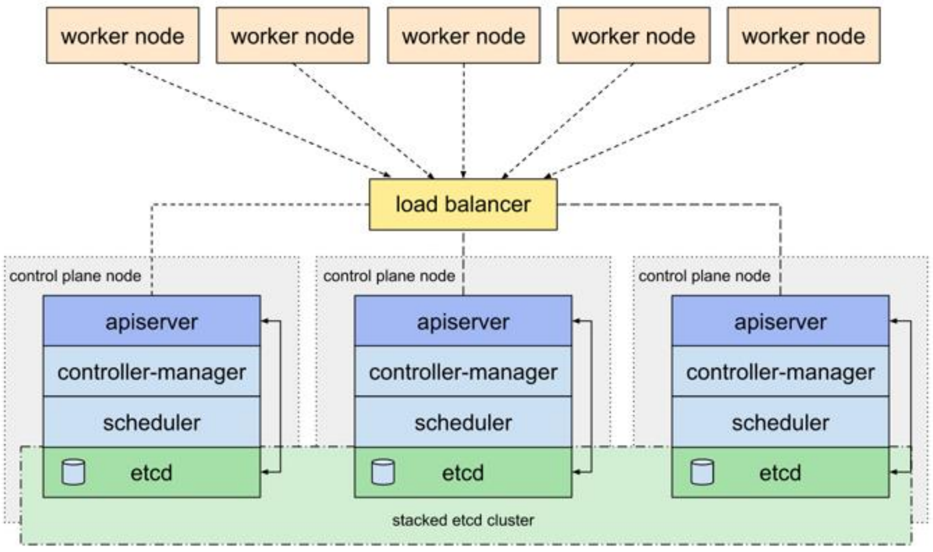


Figure 16 Malaga K8s setup

2.6.3 Athens Platform Kubernetes

Athens' platform, displayed in Figure 17, uses similar solution to Malaga and OpenShift deployments explained, with two main differences:

- VPN Technology. In this case the technology for the VPN access is the OpenConnect [12] (based almost entirely on the standard HTTPS and DTLS [13] protocols).

- Kubernetes architecture: Athens' platform is composed of one master node and two worker nodes.

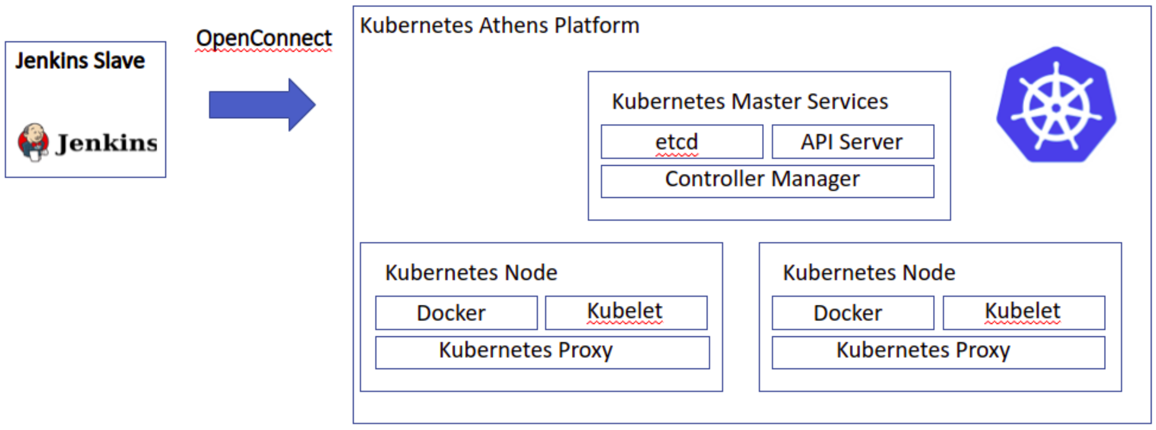


Figure 17 Athens K8s setup

3 Certification Tools

This section outlines the tools developed to fulfil Certification requirements specified in Deliverable 2.2 [15]. This document groups the requirements in three categories:

- Software product quality: This category includes CAPIF compliance and 5G integration through NEF APIs.
- Security: This category outlines a series of tests to guarantee NetApp security compliance.
- Marketplace: This category refers to Licensing, Open Source and validation of binary images of the NetApps.

3.1 Software Quality Tools

This subsection describes two tools developed in EVOLVED-5G to enable certification of NetApps against 5G exposure APIs. These tools realize the 3GPP NEF (Network Exposure Function) and CAPIF (Common API Framework) APIs. The tools are based in 3GPPP Release 17 specification listed at a repository in GitHub [16].

- NEF Services tool was already described in Deliverable 3.1 [17] and here we provide an update on the progress made since the publication of this deliverable. Documentation Quality is described to be checked manually and therefore there is no tool developed for that purpose planed in the project.
- CAPIF Services tool (developed by TID and FOGUS) is described in detail for the first time in the project. This tool has already a first release, and work continues to improve the tool adding support for more APIs and the TLS transport layer.

3.1.1 NEF Services

Network Exposure Function (NEF) enables third-party application providers utilize the network capabilities and services that the 5G system offers. NEF services are of vital importance to the 5G NPN (Non-Public Network) in the context of the EVOLVED-5G ecosystem. NetApp, which is the main component developed in the project, utilize NEF services in order to expose business APIs to vertical applications. Therefore, it is considered mandatory that a NetApp is capable of communicating with the NEF services provided by the 5GC network. To expose the network capabilities, NEF needs to interface with a set of network functions within the 5GC (i.e., Southbound APIs). Considering the fact that at this stage both Athens and Malaga platforms do not implement the entire service-based architecture and the southbound interfaces that NEF requires to offer the standardized APIs, the communication with network functions is a challenging task. To this end, NEF services are provided through an emulator that creates simulated events (e.g., mobility aware event where UEs are moving in predefined paths). The details of implementation aspects of the NEF emulator are provided in D3.1[17]. Last but not least, NEF services are described thoroughly in D4.1 [23].

3.1.2 CAPIF Services

The CAPIF has been standardized to guarantee a unique reference model for API-based service provisioning in 3GPP systems. The great potential of such framework is currently revealed due to the need for interaction between vertical industries and mobile networks.

In particular, EVOLVED-5G has selected CAPIF as the API Framework to enable the interaction between the NetApps (vertical industries domain) and the 5G Core exposed APIs (mobile networks domain). Towards realizing the above-mentioned interaction, we describe key functional aspects of CAPIF by focusing on related functional entities and APIs. In addition, we implemented the CAPIF Core Function, i.e., the main entity of CAPIF, as an effort to set the basis for an ecosystem where developers from vertical industries can easily develop network-interacting applications. To facilitate any further contribution in the area, our CAPIF implementation follows the principles of microservice programming, and it is released (open source) [18] together with a set of evaluation tests.

The CAPIF functional architecture is covered in 3GPP TS 23.222 [19] "Functional architecture and information flows to support Common API Framework for 3GPP Northbound APIs" (since Release 15) and they are also published in the related ETSI TS 123.222 [20] "Common API Framework for 3GPP Northbound APIs". Based on the architecture and the procedures defined in these reports, additional information and requirements are considered, regarding CAPIF security features and security mechanisms, as presented in 3GPP TS 33.122 [21]. The specification of the CAPIF APIs that are needed for realizing the CAPIF functionality are part of 3GPP TS 29.222 [22]. Actually, within this technical specification the interacting protocol for the CAPIF Northbound APIs is described.

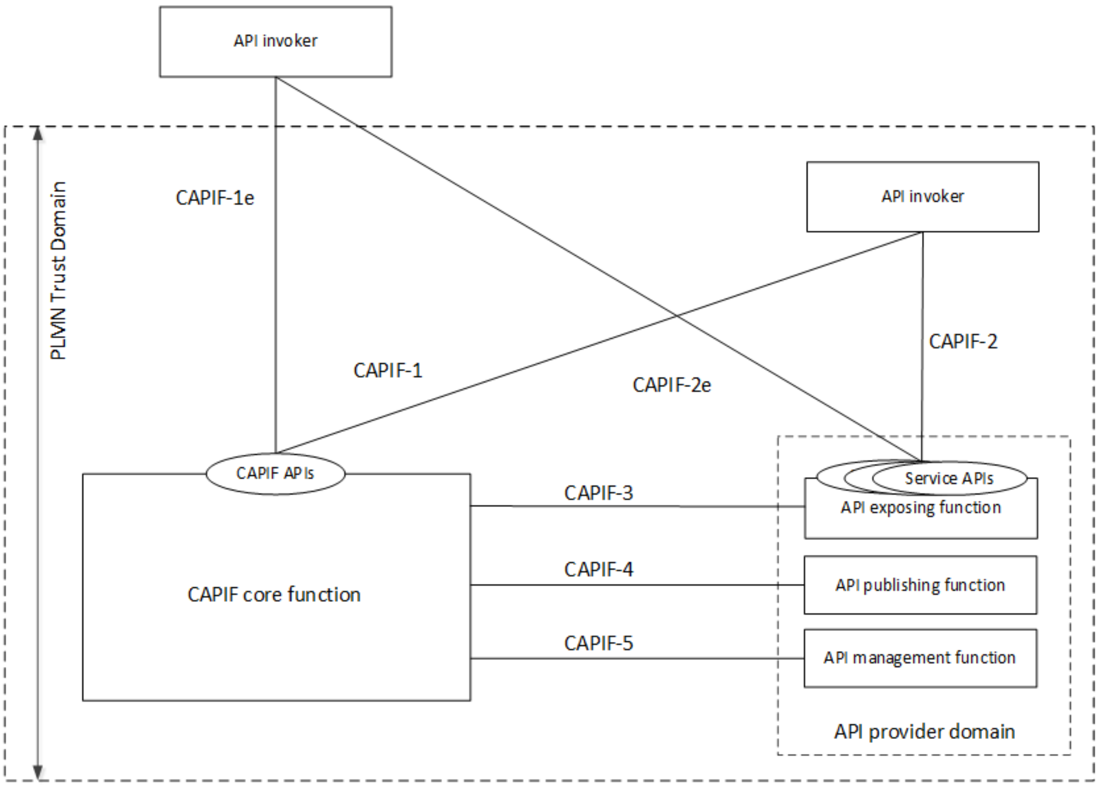


Figure 18. Functional entities and Reference points of CAPIF architecture (3GPP TS 23.222)

CAPIF Architecture is displayed in Figure 18 . Three main entities can be observed, namely, the API invoker, the CAPIF Core Function, and the API provider.

The API invoker is typically provided by a 3rd party application that supports the following capabilities:

- Supporting the authentication by providing the API invoker identity and other information required for authentication of the API invoker;
- Supporting mutual authentication with CAPIF;

- Obtaining the authorization prior to accessing the API;
- Discovering service APIs information; and
- Invoking the service APIs.

The CAPIF core function is the main entity of the CAPIF, and it consists of the following capabilities:

- Authenticating the API invoker based on the identity and other information required for authentication of the API invoker;
- Supporting mutual authentication with the API invoker;
- Providing authorization for the API invoker prior to accessing the service API;
- Publishing, storing and supporting the discovery of service APIs information;
- Controlling the service API access based on PLMN operator configured policies;
- Storing the logs for the service API invocations and providing the service API invocation logs to authorized entities;
- Charging based on the logs of the service API invocations;
- Monitoring the service API invocations;
- Onboarding a new API invoker and offboarding an API invoker;
- Storing policy configurations related to CAPIF and service APIs;
- Support accessing the logs for auditing (e.g., detecting abuse); and
- Supports publishing, discovery of service APIs information with another CAPIF core function in CAPIF interconnection.

The API provider is an entity that provides API Exposing, Publishing and Management functions. For each one of the functions specific reference points have been defined (CAPIF-3,4, and 5, as presented in Figure 18.

- **The API exposing function (AEF)** is the provider of the service APIs and is also the service communication entry point of the service API to the API invokers. The API exposing function consists of the following capabilities:
    - Authenticating the API invoker based on the identity and other information required for authentication of the API invoker provided by the CAPIF core function;
    - Validating the authorization provided by the CAPIF core function; and
    - Logging the service API invocations at the CAPIF core function.
- **The API publishing function (APF)** enables the API provider to publish the service APIs information in order to enable the discovery of service APIs by the API invoker. The API publishing function consists of the following capability:
    - Publishing the service API information of the API provider to the CAPIF core function.
- **The API management function (AMF)** enables the API provider to perform administration of the service APIs. The API management function consists of the following capabilities:
    - Auditing the service API invocation logs received from the CAPIF core function;
    - Monitoring the events reported by the CAPIF core function;
    - Configuring the API provider policies to the CAPIF core function;
    - Monitoring the status of the service APIs; and
    - Registering and maintaining registration information of the API provider domain functions on the CAPIF core function.

3.1.2.1 CAPIF Implementation Aspects

To implement the API services of the CAPIF Core Function (CCF), the first thing needed is the CAPIF API definitions/signatures. Those have been specified in 3GPP TS

29.222[22], and 3GPP has published the related YAML files[1] as well. The following services have been defined for the CCF:

- Discover Service: API to ask CCF the list of APIs published and available in CAPIF.
- Publish Service: API to publish APIs information from APF/AEFs.
- Events: API to manage Event subscriptions that enable Event Notification from CCF.
- API Invoker Management: API to enable the onboarding of API Invokers into CCF.
- Security: API to enable setting security profiles and retrieve security Tokens.
- Access Control Policy: API to manage access control rules in CCF.
- Logging API Invocation: API to add logs on API consumption.
- Auditing: API to query and retrieve service API invocation logs stored on the CAPIF core function
- AEF Authentication: API for AEF security management
- API Provider Management: API for API provider domain functions management
- Routing Information: API to provide API routing information.

The YAML files of those services, can be used by a Swagger editor to inspect all information elements in JSON. Each of these YAML files defines one or more APIs and the supported methods to use them (POST, GET, DELETE, PUT). With the YAML files of the services described above, it is possible to generate automatic code that implements HTTP/HTTPS Endpoints that act as Servers that accept HTTP requests.

To build CAPIF Core Function several software tools have been used to develop, implement, build, and test the CAPIF API services.

- **OpenApi Generator[2]:** This software program allows generation of API clients SDKs (Software Development Kit tools), or API servers given an OpenAPI specification. Moreover, it is possible to generate code in more than 20 different programming languages.
- **MongoDB:** Mongo is a non-SQL and open-source database tool used to provide storage to different CAPIF core functionalities.
- **Nginx:** Nginx is an open-source web serving technology that we use as a reverse proxy to forward requests to the different CAPIF modules
- **Flask:** Flask is a micro-service web framework written in python used to build CAPIF. The main advantage of this framework is its modularity. This feature allows us to run different CAPIF services and mix them directly with other services as database with relative ease.
- **Robot framework:** Robot is a generic test automation framework used for acceptance testing and acceptance test-driven development.
- **Docker:** Docker is an open-source containerization tool for building, running, and managing containers, where software is deployed. We use Docker to build each service of the CAPIF. In this way, we can keep each CAPIF service development isolated from other services. This design pattern is known as a micro-services-oriented architecture and is widely used in software development due to its

[1] https://github.com/jdegre/5GC_APIs
[2] https://github.com/OpenAPITools/openapi-generator

innumerable advantages like better fault isolation and improved scalability, among others.

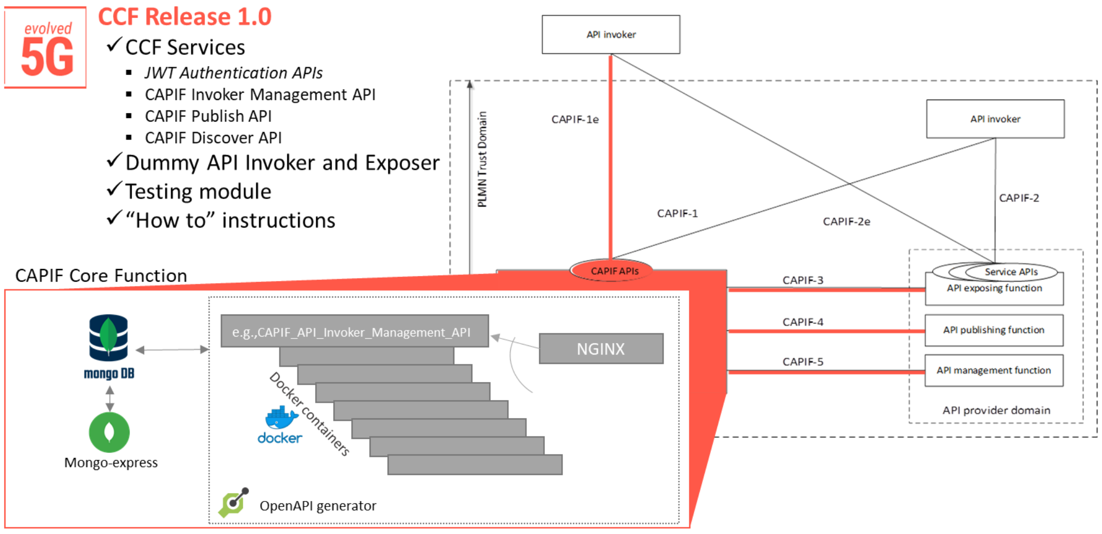
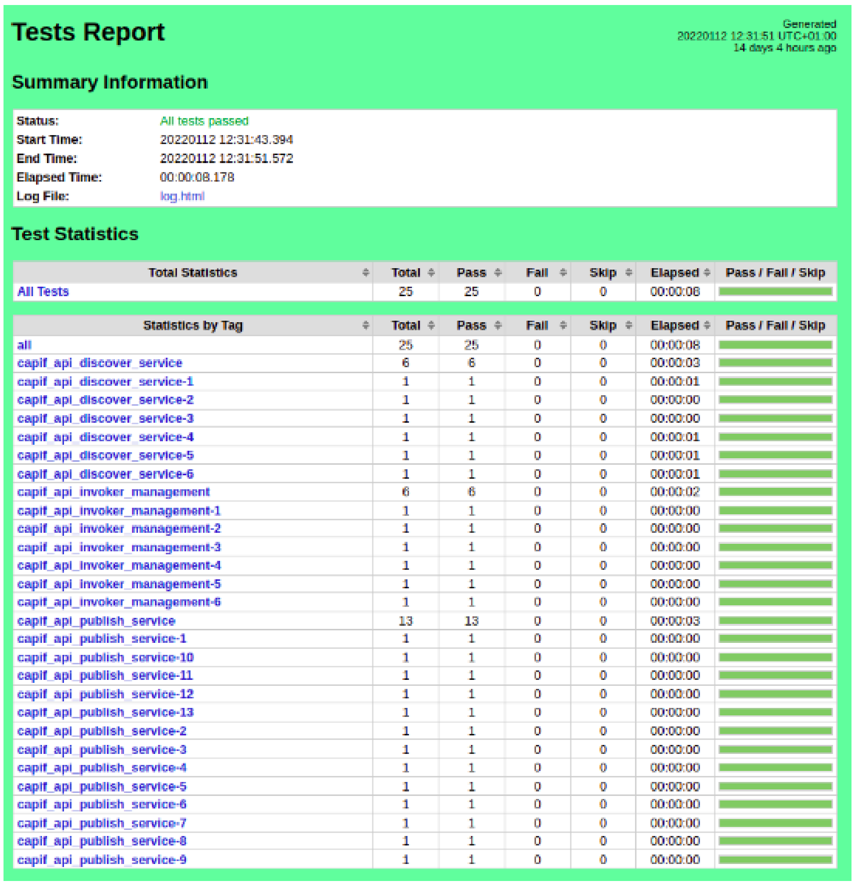


Figure 19 Tools used in the Capif Core Function (CCF) Implementation

In order to guarantee the integrity of our implementation, we exploited an automated test suite covering the core functionality of each CAPIF API service. Robot framework, described in Section 2.5, is the testing tool we have curated to ensure the quality and robustness of developed code. Moreover, we have defined a test strategy to improve the code quality. This test strategy is composed of two steps:

- **Test plan document elaboration**: In this step, test plans are described, including various behavior scenarios (considering both success and failure cases). The test plan structure includes clarifications on the pre-conditions, the action that takes place, and the post-conditions (response/result expected). The horizon of the test plans that can be defined, moves beyond the request-respond information that is available in the related 3GPP specifications, in a sense that behavioral scenarios beyond the basic functionality are defined to stress test the implementation integrity.
- **Test implementation and execution**: This step continues after finishing the elaboration of the test plan documentation. Each test suite is implemented and included into an automation pipeline that checks the status of the code after every deployment in the platform. As depicted in (Figure 20) generated reports are very descriptive and self-explanatory, making test cases easy to read/understand.



Tests Report

Generated
20220112 12:31:51 UTC+01:00
14 days 4 hours ago

Summary Information

Status:	All tests passed
Start Time:	20220112 12:31:43.394
End Time:	20220112 12:31:51.572
Elapsed Time:	00:00:08.178
Log File:	log.html

Test Statistics

Total Statistics	Total	Pass	Fail	Skip	Elapsed	Pass / Fail / Skip
All Tests	25	25	0	0	00:00:08	🟩

Statistics by Tag	Total	Pass	Fail	Skip	Elapsed	Pass / Fail / Skip
all	25	25	0	0	00:00:08	🟩
capif_api_discover_service	6	6	0	0	00:00:03	🟩
capif_api_discover_service-1	1	1	0	0	00:00:01	🟩
capif_api_discover_service-2	1	1	0	0	00:00:00	🟩
capif_api_discover_service-3	1	1	0	0	00:00:00	🟩
capif_api_discover_service-4	1	1	0	0	00:00:01	🟩
capif_api_discover_service-5	1	1	0	0	00:00:01	🟩
capif_api_discover_service-6	1	1	0	0	00:00:01	🟩
capif_api_invoker_management	6	6	0	0	00:00:02	🟩
capif_api_invoker_management-1	1	1	0	0	00:00:00	🟩
capif_api_invoker_management-2	1	1	0	0	00:00:00	🟩
capif_api_invoker_management-3	1	1	0	0	00:00:00	🟩
capif_api_invoker_management-4	1	1	0	0	00:00:00	🟩
capif_api_invoker_management-5	1	1	0	0	00:00:00	🟩
capif_api_invoker_management-6	1	1	0	0	00:00:00	🟩
capif_api_publish_service	13	13	0	0	00:00:03	🟩
capif_api_publish_service-1	1	1	0	0	00:00:00	🟩
capif_api_publish_service-10	1	1	0	0	00:00:00	🟩
capif_api_publish_service-11	1	1	0	0	00:00:00	🟩
capif_api_publish_service-12	1	1	0	0	00:00:00	🟩
capif_api_publish_service-13	1	1	0	0	00:00:00	🟩
capif_api_publish_service-2	1	1	0	0	00:00:00	🟩
capif_api_publish_service-3	1	1	0	0	00:00:00	🟩
capif_api_publish_service-4	1	1	0	0	00:00:00	🟩
capif_api_publish_service-5	1	1	0	0	00:00:00	🟩
capif_api_publish_service-6	1	1	0	0	00:00:00	🟩
capif_api_publish_service-7	1	1	0	0	00:00:00	🟩
capif_api_publish_service-8	1	1	0	0	00:00:00	🟩
capif_api_publish_service-9	1	1	0	0	00:00:00	🟩

Figure 20 Result of Robot framework CAPIF Functional Tests

3.1.2.2 Remarks on CAPIF Core Function RELEASE 1.0

The entire code of the CAPIF core function Release 1.0 can be found in the following public repository of the EVOLVED-5G project:

Code: EVOLVED-5G_CAPIF_API_Services_RELEASE1.0
Development branch: EVOLVED-5G_CAPIF_API_services_development_branch

The above-mentioned repository includes a set of the Python-flask Mockup servers that provide the CAPIF core function APIs as well as auxiliary code and instructions for installing and testing the API services.
More precisely, the related repository includes the following folders:

- **services**: Services developed following CAPIF API specifications. Also, other complementary services (e.g., NGINX and JWTauth services for the authentication of API consuming entities).
- **tools**: Auxiliary tools. Robot Framework related code and OpenAPI scripts
- **test**: Tests developed using Robot Framework
- **docs**: Documents related to the code in the repository
    - images: images used in the repository
    - test_plan: test plan descriptions for each API service referring to the tests that are executed with the robot framework.
    - testing_with_postman: auxiliary JSON file needed for the Postman-based examples.

- **iac**: Infrastructure as Code, contains all the files needed for the deployment of the structure that supports the services. (It is used only for the case of non-local deployment of the CCF services e.g., in the Openshift of EVOLVED-5G project))
    - Terraform: Deploy file
- **pac**: Jenkins files to manage different automated tasks. (It is used only for the case of non-local deployment of the CCF services e.g., in the Openshift of EVOLVED-5G project)
    - Jenkins Pipelines

In CAPIF Release 1.0, three main CCF APIs services (plus an auxiliary one) are provided:

- *JWT Authentication APIs (Auxiliary API service)*
- **CAPIF Invoker Management API**
- **CAPIF Publish API**
- **CAPIF Discover API**

3.1.2.3 Exploring CAPIF Core Function RELEASE 1.0

Complete "HOW TO" instructions are available in the README file in the repository.

3.1.2.3.1 Install and run CAPIF services in a Local Repository

Three different options are offered as described in the related Github repository:

- Run All CAPIF Services locally with Docker images
- Run each service using Docker
- Run each service using Python

NOTE 1: The use of OpenAPI generator during the initial set up process will provide the images of all the CCF API services. However, it should be noted that in Release 1.0, the functionalities of CAPIF Invoker Management API, CAPIF Publish API, and the CAPIF Discover API are available and ready to use/test.

NOTE 2: After the successful installation the Mongo DB Dashboard will be available on http://0.0.0.0:8082/ (if accessed from localhost) or http://<Mongo Express Host IP>:8082/ (if accessed from another host)

3.1.2.3.2 Familiarize with CAPIF core function APIs

Two different options are offered, as listed below:

- Execute CAPIF API calls using PostMan
- Execute CAPIF API calls using Curl

3.1.2.3.3 Execute test plans for the CAPIF core function APIs

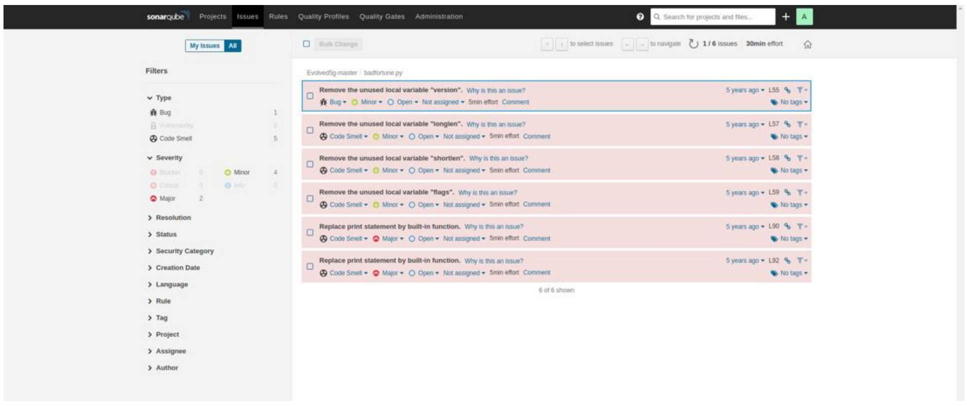
For testing the functionality of the APIs provided, a complete set of tests has been prepared (CCF_Release1.0_test_plans). For the execution of the tests the Robot framework is used. Instructions for setting up the Robot Framework and running the tests are also provided.

3.2 SECURITY CERTIFICATION

From the security requirements, described in Deliverable 2.2 [15], we have selected commercial tools to address static code analysis and vulnerability scanning. NetApp binary images security analysis is also performed but enclosed in next subsection.

3.2.1 Static Code Analysis

SonarQube [24] is an open-source platform developed by SonarSource for continuous inspection of code quality. Sonarqube is a static code analysis tool that measures the overall quality of a software project. SonarQube can provide a detailed report of the code quality, detecting problems like code duplication, code smells and vulnerabilities.



Evolved5g-master / badfortune.py

Remove the unused local variable "version". Why is this an issue? 5 years ago L55
Bug Minor Open Not assigned 5min effort Comment No tags

Remove the unused local variable "longlen". Why is this an issue? 5 years ago L57
Code Smell Minor Open Not assigned 5min effort Comment No tags

Remove the unused local variable "shortlen". Why is this an issue? 5 years ago L58
Code Smell Minor Open Not assigned 5min effort Comment No tags

Remove the unused local variable "flags". Why is this an issue? 5 years ago L59
Code Smell Minor Open Not assigned 5min effort Comment No tags

Replace print statement by built-in function. Why is this an issue? 5 years ago L90
Code Smell Major Open Not assigned 5min effort Comment No tags

Replace print statement by built-in function. Why is this an issue? 5 years ago L92
Code Smell Major Open Not assigned 5min effort Comment No tags

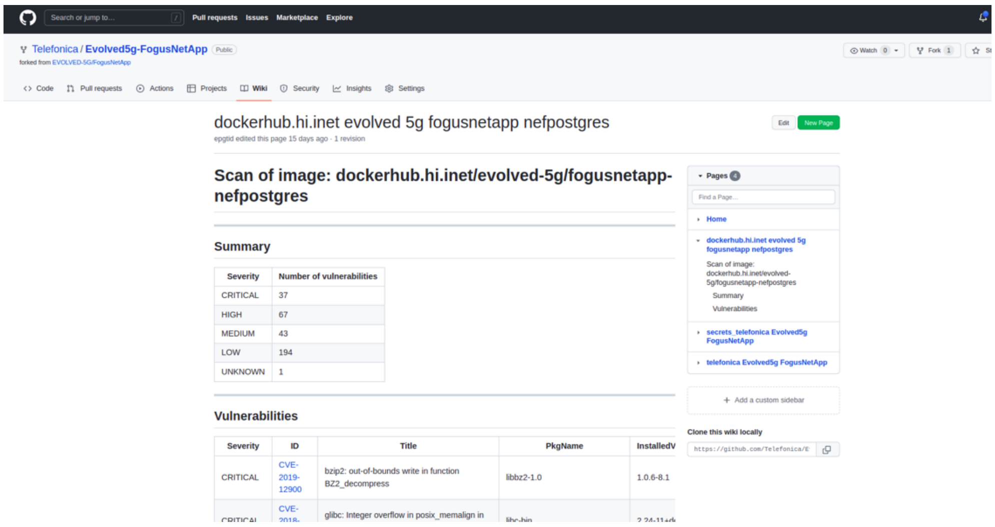
6 of 6 shown

Figure 21 Sonarqube report screenshot

Sonarqube is the quality assessment tool selected in Evolved5G for measuring the overall quality of NetApps. Sonarqube brings to the NetApp developers countless advantages such as increased productivity, improved developer skills and increased consistency to the NetApps being implemented in the Evolved5G ecosystem.

3.2.2 Vulnerability Scanning

Trivy [25] is a comprehensive vulnerability/misconfiguration/secret scanner for containers and other artifacts. Trivy detects vulnerabilities of OS packages (Alpine, RHEL, CentOS, etc.) and language-specific packages (Bundler, Composer, npm, yarn, etc.). In addition, Trivy scans Infrastructure as Code (IaC) files such as Terraform and Kubernetes, to detect potential configuration issues that expose your deployments to the risk of attack. Trivy also scans hardcoded secrets like passwords, API keys and tokens.


Search or jump to...
Pull requests
Issues
Marketplace
Explore
Telefonica / Evolved5g-FogusNetApp
Public
forked from EVOLVED-5G/FogusNetApp
Watch 0
Fork 1
Code
Pull requests
Actions
Projects
Wiki
Security
Insights
Settings
Edit
New Page
Pages 4
Find a Page...
Home
dockerhub.hi.inet evolved 5g fogusnetapp nefpostgres
Scan of image: dockerhub.hi.inet/evolved-5g/fogusnetapp-nefpostgres
Summary
Vulnerabilities
secrets_telefonica Evolved5g FogusNetApp
telefonica Evolved5g FogusNetApp
Add a custom sidebar
Clone this wiki locally
https://github.com/Telefonica/E

dockerhub.hi.inet evolved 5g fogusnetapp nefpostgres
epgtid edited this page 15 days ago · 1 revision

Scan of image: dockerhub.hi.inet/evolved-5g/fogusnetapp-nefpostgres

Summary

Severity	Number of vulnerabilities
CRITICAL	37
HIGH	67
MEDIUM	43
LOW	194
UNKNOWN	1

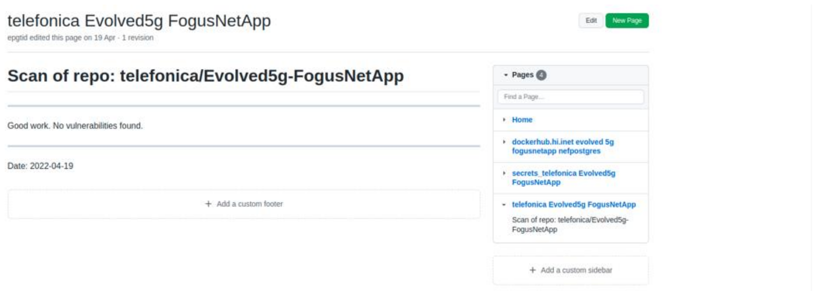
Vulnerabilities

Severity	ID	Title	PkgName	InstalledV
CRITICAL	CVE-2019-12900	bzip2: out-of-bounds write in function BZ2_decompress	libbz2-1.0	1.0.6-8.1
CRITICAL	CVE-2018-	glibc: Integer overflow in posix_memalign in	libc-bin	2.24-11+d

Figure 22 Trivy scanning results

In Figure 22 Trivy scanning results, it is shown the three different sections that will be analysed with Trivy:

- Code Vulnerabilities: Trivy does a static code analysis as well of the repository, in a similar way that SonarQube does but with a lower degree of exhaustiveness and more focused in a security aspect rather than code quality.


Edit
New Page
Pages 4
Find a Page...
Home
dockerhub.hi.inet evolved 5g fogusnetapp nefpostgres
secrets_telefonica Evolved5g FogusNetApp
telefonica Evolved5g FogusNetApp
Scan of repo: telefonica/Evolved5g-FogusNetApp
Add a custom footer
Add a custom sidebar

telefonica Evolved5g FogusNetApp
epgtid edited this page on 19 Apr · 1 revision

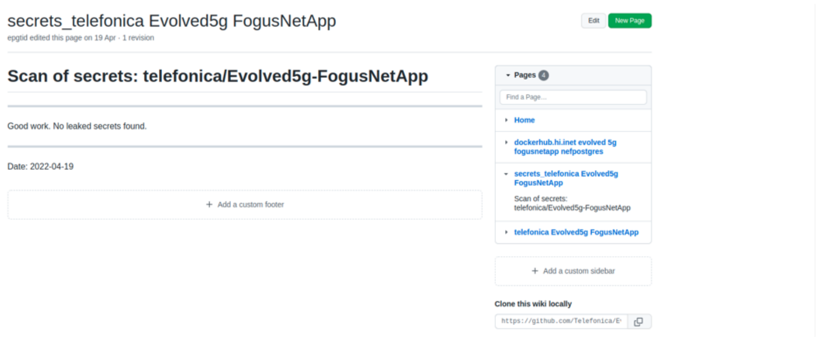
Scan of repo: telefonica/Evolved5g-FogusNetApp

Good work. No vulnerabilities found.

Date: 2022-04-19

Figure 23 Trivy Code Vulnerabilities example

- Secrets Leakage: Trivy scans all the commits that have been made in the repository searching for any secret leaks in the code. This functionality is so important because it is likely that some secrets are hardcoded during the first phases of the software project creation, and then once removed, those secrets will be persisted in the commit history.


Edit
New Page
Pages 4
Find a Page...
Home
dockerhub.hi.inet evolved 5g fogusnetapp nefpostgres
secrets_telefonica Evolved5g FogusNetApp
Scan of secrets: telefonica/Evolved5g-FogusNetApp
telefonica Evolved5g FogusNetApp
Add a custom footer
Add a custom sidebar
Clone this wiki locally
https://github.com/Telefonica/E

secrets_telefonica Evolved5g FogusNetApp
epgtid edited this page on 19 Apr · 1 revision

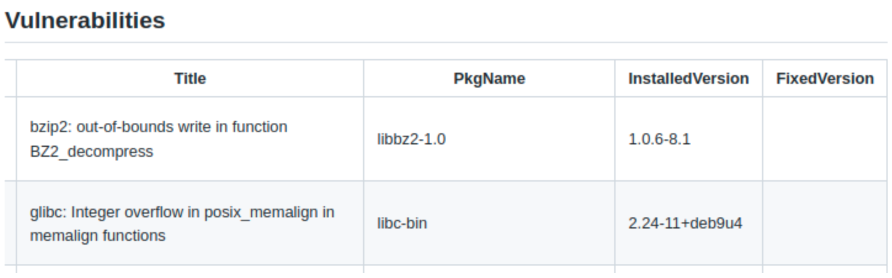
Scan of secrets: telefonica/Evolved5g-FogusNetApp

Good work. No leaked secrets found.

Date: 2022-04-19

Figure 24 Trivy secret leaked analysis example

- Container vulnerabilities: In this part of the analysis, Trivy will show all the vulnerabilities found in the container images. Here, we can see how Trivy shows a detailed description of the problem, giving the name of the affected library, and what would be the version that fixes that problem (if exists)


Vulnerabilities

Title	PkgName	InstalledVersion	FixedVersion
bzip2: out-of-bounds write in function BZ2_decompress	libbz2-1.0	1.0.6-8.1	
glibc: Integer overflow in posix_memalign in memalign functions	libc-bin	2.24-11+deb9u4	

Figure 25 Trivy Vulnerabilities detected example

3.3 MARKETPLACE CERTIFICATION

In order to upload Certified NetApps to the Marketplace, some information needs to be generated and attached to the Certification Report. This information relates to License requirements (form commercial or open-source products used to develop the NetApp). For this purpose, Debriked tool has been selected to illustrate this process.

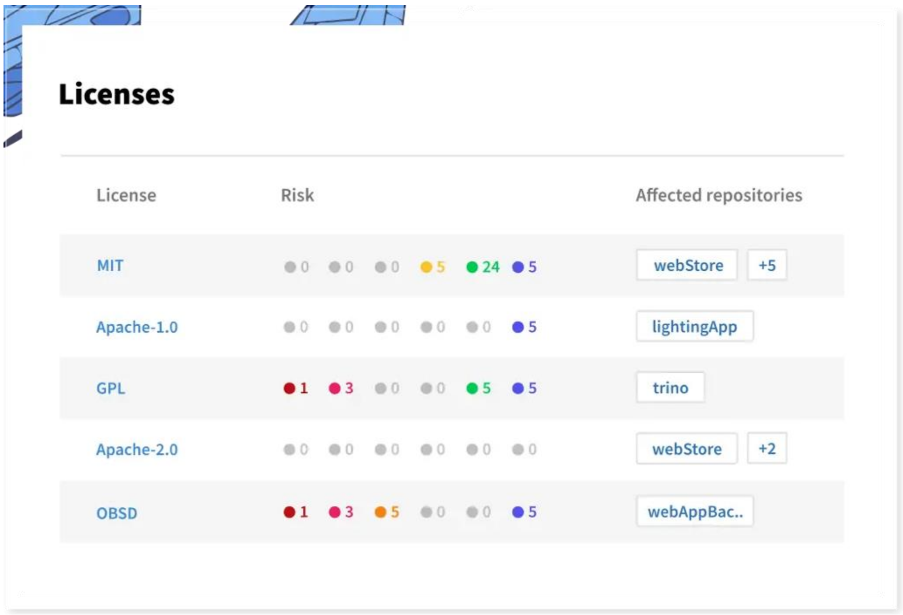
Additionally, the binary image of the NetApp will be analysed for security compliance to production ready container images industry standards using Trivy tool again.

3.3.1 Use Policy/Terms of service/ License files/ Open SourceScan Report

For analysing required Licenses by the NetApp being Certified, we have integrated Debriked Compliance tool [26].

This tool allows automating license compliance processes with customizable intake rules. Debriked compliance tools serves three purposes:

- **Automate intake control:** Stop non-compliant components from entering NetApp codebase by enabling automated intake rules.
- **Understand risk:** Evaluate risk level for NetApp by simply setting an intended use case for NetApp repositories.
- **Report and analyse:** Easily export a license report to be attached to Certification report and keep track of NetApp compliance progress over time.



Licenses

License	Risk	Affected repositories
MIT	● 0 ● 0 ● 0 ● 5 ● 24 ● 5	webStore +5
Apache-1.0	● 0 ● 0 ● 0 ● 0 ● 0 ● 5	lightingApp
GPL	● 1 ● 3 ● 0 ● 0 ● 5 ● 5	trino
Apache-2.0	● 0 ● 0 ● 0 ● 0 ● 0 ● 0	webStore +2
OBSD	● 1 ● 3 ● 5 ● 0 ● 0 ● 5	webAppBac..

Figure 26 Debriked License Analysis

Certification process will use Debriked to analyze License dependencies and report them in the Certification report produced.

3.3.2 Valid Container Image and/or End-point details

For certifying the NetApp image security, we rely on Trivy . This is a repo that has controls that cover both PodSecurityPolicy (PSP) and the Kubernetes Pod Security Standards (PSS), plus additional best practices.
The Kubernetes Pod Security Standards (PSS) are the official standard for security best practices for pods. These standards overlap with the checks that PodSecurityPolicies can enforce.
PSS has 14 controls that are grouped into three standards: Baseline, Restricted and Privileged. Appshield uses Baseline and Restricted; the Privileged standard specifically allows privileged execution. This table maps PSS controls to PSP controls:

PSS Control	PSP Control
1-Host Namespaces	2-Usage of host namespaces. 3-Usage of host networking and ports
2-Privileged Containers	1-Running of privileged containers
3-Capabilities	11-Linux capabilities
4-HostPath Volumes	5-Usage of the host filesystem
5-Host Ports	Not covered in PSP
6-AppArmor (optional)	14-The AppArmor profile used by containers
7-SELinux (optional)	12-The SELinux context of the container
8-/proc Mount Type	13-The Allowed Proc Mount types for the container
9-Sysctls	16-The sysctl profile used by containers

4 CERTIFICATION PROCESS

4.1 CERTIFICATION PROCESS OVERVIEW

NetApp Certification Process is responsible for Certifying that NetApps are compliant with the Evolved5G criteria defined for considering a NetApp an Evolved5G NetApp.

This criterion is based in principles defined in D 2.2 [15]:

- The NetApp is secure
- The NetApp can be deployed in Cloud infrastructure
- The NetApp is Cloud Native
- The NetApp uses CAPIF APIs to Discover and Consume APIs
- The NetApp uses NEF APIs to interact with 5G Infrastructure

All these conditions are being Certified during the Certification Process using Industry standard Automation technology and design principles. Thus, Evolved5G guarantees that Certification process is:

- Repeatable: Certification can be performed many times to NetApps
- Deterministic: the repetition of Certification process over a NetApp with no modifications always produce the same results
- Traceable: all test results can be tracked and analysed after process completes
- Automated: certification process is fully automated and requires no manual configuration or intervention eliminating variability in the execution conditions

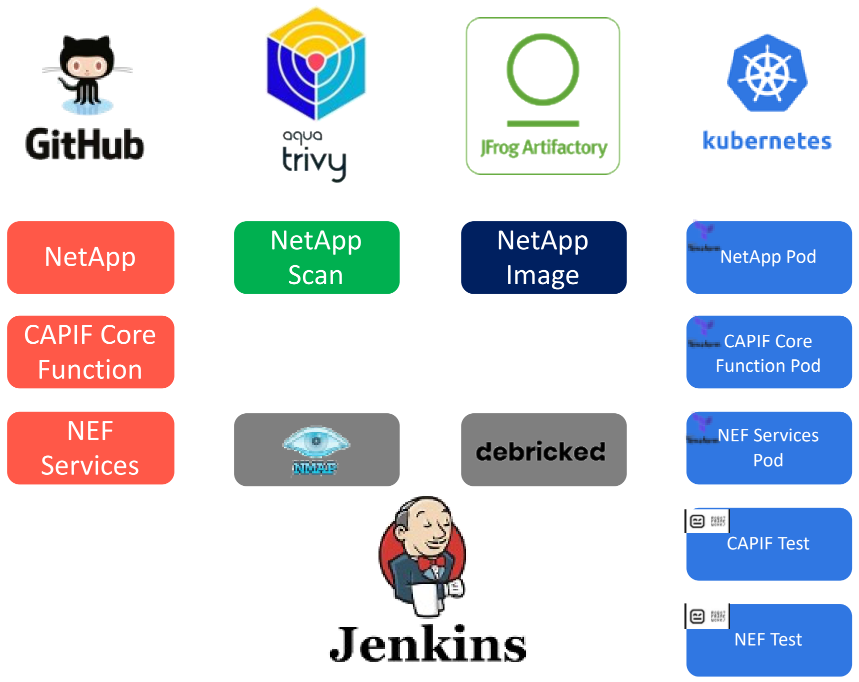
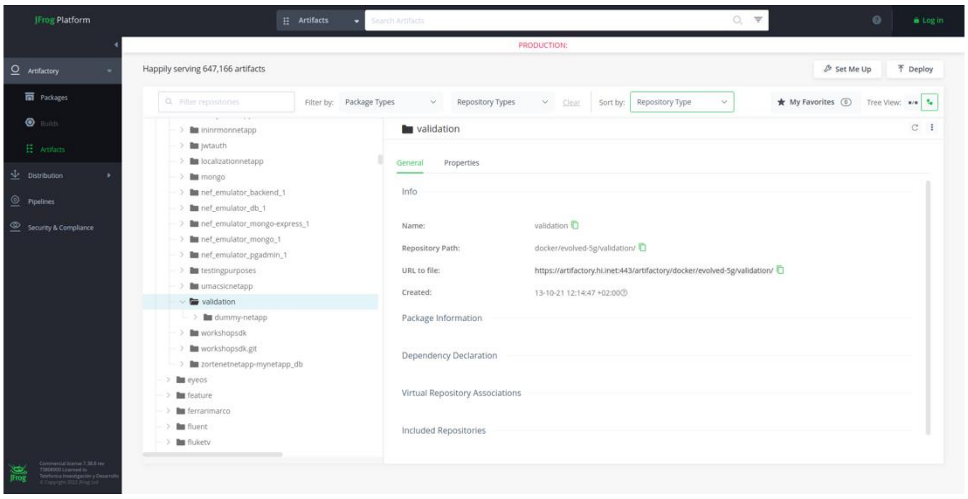


Figure 27 Certification Process elements

The starting point of the Certification Process is always a Validated NetApp. During Validation process, the NetApp image is uploaded to Open repository (Artifactory) in the Validation folder.



Happily serving 647,166 artifacts

ininrmonnetapp
jwtauth
localizationnetapp
mongo
nef_emulator_backend_1
nef_emulator_db_1
nef_emulator_mongo-express_1
nef_emulator_mongo_1
nef_emulator_pgadmin_1
testingpurposes
umacsicnetapp
validation
dummy-netapp
workshopsdk
workshopsdk.git
zortenetnetapp-mynetapp_db
eyeos
feature
ferrarimarco
fluent
fluketv

validation

General | Properties

Info

Name: validation
Repository Path: docker/evolved-5g/validation/
URL to file: https://artifactory.hi.inet:443/artifactory/docker/evolved-5g/validation/
Created: 13-10-21 12:14:47 +02:00

Package Information

Dependency Declaration

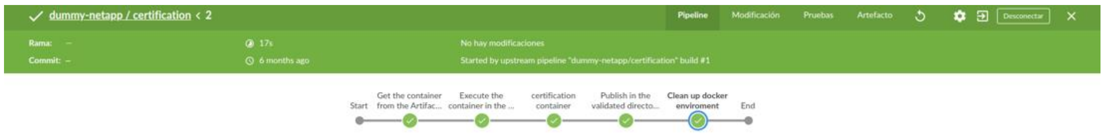
Virtual Repository Associations

Included Repositories

Figure 28 Artifactory Validation Folder

Certification Process will grab NetApp Source Code from Github repository and the Netapp image from the Validation Folder in Artifactory. These are the two fundamental content inputs to the process. Additionally, it is necessary to indicate the 5G Platform environment selected to run the Certification tests. This environment can be either Athens or Málaga.

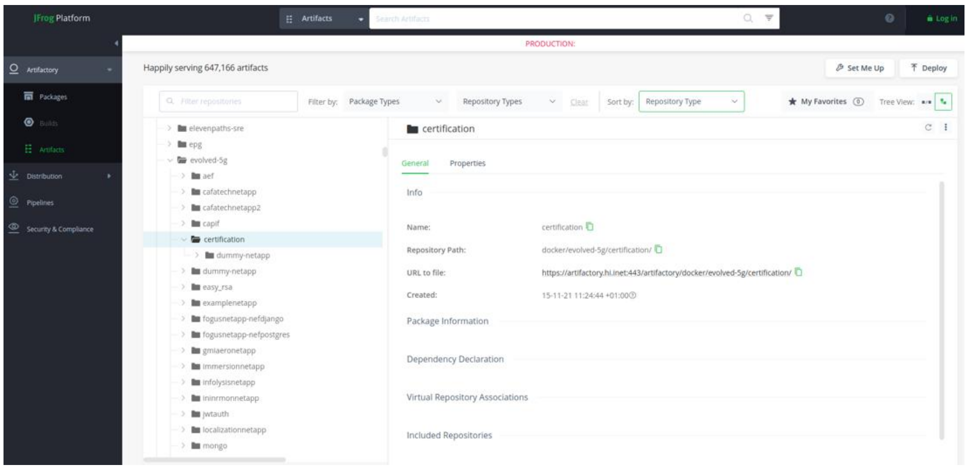
With these inputs, it is possible to launch the Certification Pipeline that will automatically execute all certifications tests to complete Evolved5G Certification. During Certification, additional inputs (CAPIF Core Function, NEF Services) and tools (NMAP [27], Trivy. [25], Debriked[26]) will be used as described in the next section 4.2.


dummy-netapp / certification < 2
Pipeline
Modificación
Pruebas
Artefacto
Desconectar
Start
Get the container from the Artifac...
Execute the container in the ...
certification container
Publish in the validated directo...
Clean up docker enviroment
End

Rama: –
Commit: –
17s
6 months ago
No hay modificaciones
Started by upstream pipeline "dummy-netapp/certification" build #1

Figure 29 Jenkins Certification Pipeline execution

Upon completion of the process, a file with results of the Certification will be uploaded, along with the certified NetApp image to Artifactory Certification folder.



Happily serving 647,166 artifacts

elevenpaths-sre
epg
evolved-5g
aef
cafatechnetapp
cafatechnetapp2
capif
certification
dummy-netapp
dummy-netapp
easy_rsa
examplenetapp
fogusnetapp-nefdjango
fogusnetapp-nefpostgres
gmiaeronetapp
immersionnetapp
infolysisnetapp
ininrmonnetapp
jwtauth
localizationnetapp
mongo

certification

General | Properties

Info

Name: certification
Repository Path: docker/evolved-5g/certification/
URL to file: https://artifactory.hi.inet:443/artifactory/docker/evolved-5g/certification/
Created: 15-11-21 11:24:44 +01:00

Package Information

Dependency Declaration

Virtual Repository Associations

Included Repositories

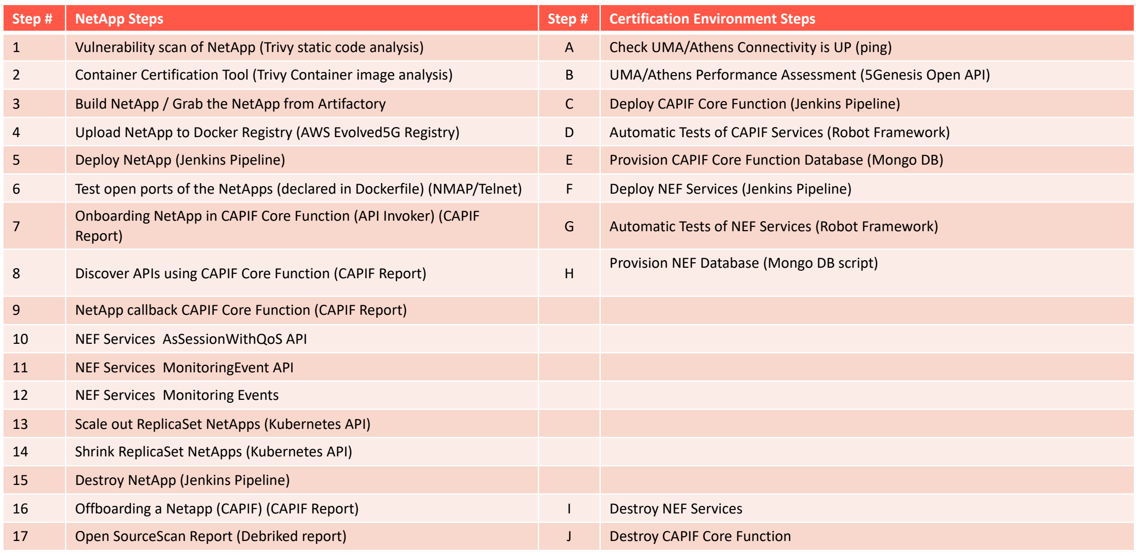
Figure 30 Artifactory Certification Folder

When certification fails, feedback is provided to the NetApp developer to fix the root cause of failure and start the Certification process again.

4.2 Certification Pipeline

Certification Pipeline running in Jenkins is the responsible to perform the Certification Process defined in Evolved5G. It uses nested pipelines for some of the operations and both internal tools (Robot Framework) and external tools (Debriked, Trivy, Robot Framework) to perform some of the certification tests defined.

The Certification pipeline is summarized in Figure 31 Certification Process Pipeline Steps OverviewFigure 31 Certification Process Pipeline Steps Overview



Step #	NetApp Steps	Step #	Certification Environment Steps
1	Vulnerability scan of NetApp (Trivy static code analysis)	A	Check UMA/Athens Connectivity is UP (ping)
2	Container Certification Tool (Trivy Container image analysis)	B	UMA/Athens Performance Assessment (5Genesis Open API)
3	Build NetApp / Grab the NetApp from Artifactory	C	Deploy CAPIF Core Function (Jenkins Pipeline)
4	Upload NetApp to Docker Registry (AWS Evolved5G Registry)	D	Automatic Tests of CAPIF Services (Robot Framework)
5	Deploy NetApp (Jenkins Pipeline)	E	Provision CAPIF Core Function Database (Mongo DB)
6	Test open ports of the NetApps (declared in Dockerfile) (NMAP/Telnet)	F	Deploy NEF Services (Jenkins Pipeline)
7	Onboarding NetApp in CAPIF Core Function (API Invoker) (CAPIF Report)	G	Automatic Tests of NEF Services (Robot Framework)
8	Discover APIs using CAPIF Core Function (CAPIF Report)	H	Provision NEF Database (Mongo DB script)
9	NetApp callback CAPIF Core Function (CAPIF Report)		
10	NEF Services AsSessionWithQoS API		
11	NEF Services MonitoringEvent API		
12	NEF Services Monitoring Events		
13	Scale out ReplicaSet NetApps (Kubernetes API)		
14	Shrink ReplicaSet NetApps (Kubernetes API)		
15	Destroy NetApp (Jenkins Pipeline)		
16	Offboarding a Netapp (CAPIF) (CAPIF Report)	I	Destroy NEF Services
17	Open SourceScan Report (Debriked report)	J	Destroy CAPIF Core Function

Figure 31 Certification Process Pipeline Steps Overview

4.2.1 NetApp Steps

Step 1: The first step is to check vulnerabilities in the source Code of the NetApp. For this checking, Trivy tool will be used. The pipeline will stop if critical vulnerabilities with available patch are reported by Trivy. Otherwise, after completing the vulnerability scan, the pipeline will continue to step 2.

Step 2: Before deploying the NetApp, Jenkins will make sure that NetApp image is secure. For this, Jenkins will use again Trivy as external tool to validate the security vulnerabilities of the Netapp. If security critical issues are reported, the pipeline will stop. Otherwise, the pipeline will continue to **Step 4**.

Step 3: This step will verify that the NetApp has completed the Validation phase. To check this, Jenkins will search for the NetApp image inside the Validation folder in Artifactory. If the NetApp image exists in the folder, it will be grabbed from there and continue to **Step 4**. Otherwise, the pipeline will stop.

Step 4: Next Jenkins need to upload the NetApp image to a Docker Registry in order to enable the deployment of the NetApp in Athens and Málaga Kubernetes infrastructure in step 5. If the upload completes successfully, the pipeline will continue to **Step 4**. Otherwise, the pipeline will stop.

Step 4: Jenkins will use nested Pipeline to deploy the NetApp in the selected environment (Athens or Málaga). If the pipeline reports any error during the deployment, the certification pipeline will jump to Step 15 to clean the environment and stop. If no errors are reported, the pipeline will continue to **Step 5**.

Step 5: Once the NetApp is deployed, Jenkins will check that declared ports in Dockerfile of the NetApp are open. For this operation, connectivity tools such as NMAP [27] or Telnet [28] will be used.

Step 6: With the NetApp Deployed, next step is to check that the NetApp registers in CAPIF Core Function properly as an API Invoker. To certify the registration, Jenkins will query CAPIF Core Function Database to double check that the NetApp have been registered properly by CAPIF Core Function.

Step 7: Once the NetApp has registered in CAPIF Core Function as API Invoker, next steps is for the NetApp to Discover APIs published in CAPIF. To certify that NetApp is using Discover API from CAPIF Core Function, Jenkins will parse the logs from CAPIF Core Function Discover API to double check that the REST API have been consumed by the NetApp.

Step 8: As part of the API dialogue between the NetApp and CAPIF Core Function, the NetApp will receive CAPIF Events. These events are notifications coming from CAPIF Core Function to the NetApp, and the CAPIF Event callback URL is set by the NetApp in the API Invoker Registration. Jenkins will certify that CAPIF Events are received by NetApp properly parsing the logs from CAPIF Core Function Events API. In the logs, "200 OK" responses from the NetApp must appear as the result of Events sent.

Step 9 and 10: The NetApp Discover NEF API using CAPIF Core Function Discovery API. Two APIs are exposed by NEF: AsSessionWithQoS and MonitoringEvent. Jenkins will check which of these APIs are consumed by the NetApp and add them to the Certification Report.

Step 11: As part of the API dialogue between the NetApp and NEF Services, the NetApp will receive NEF Events. These events are notifications coming from NEF Services to the NetApp, and the NEF Event callback URL is set by the NetApp in the NEF API Requests. Jenkins will certify that NEF Events are received by NetApp properly parsing the logs from NEF Services. In the logs, "200 OK" responses from the NetApp must appear as the result of NEF Events sent.

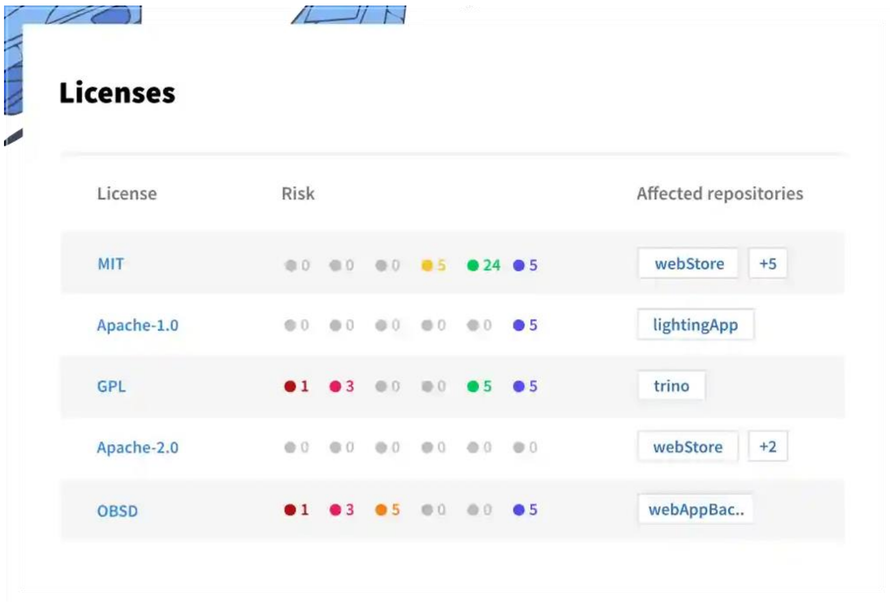
Step 12: The NetApp might support scaling out horizontally by defining a ReplicaSet workload resource. A ReplicaSet's purpose is to maintain a stable set of replica Pods running at any given time. As such, it is often used to guarantee the availability of a specified number of identical Pods. ReplicaSets' are recommended to be used defining Deployments. Deployment provides declarative updates for Pods and ReplicaSets. Jenkins will increase the number of instances of NetApp deployment and certify that NetApp Pods scale properly.

Step 13: After scaling out the NetApp increasing the number of instances, Jenkins will shrink the number of instances setting the NetApp to default deployment and will certify that the NetApp adjust to the scale defined.

Step 14: Once all previous tests have been executed, it is time to Destroy the NetApp and certify that NetApp removes properly. Jenkins will remove the NetApp from the container infrastructure, which will trigger NetApp removal clean up.

Step 15: As part of the NetApp removal clean up, the NetApp needs to unregister from CAPIF Core Function. Jenkins will certify that the NetApp has unregistered properly by querying CAPIF Core Function Database.

Step 16: With all functional tests executed, Certification pipeline will finally collect information about open-source Licenses used by the NetApp. For this operation, Jenkins will use Debriked Compliance external tool [26].


Licenses

License	Risk						Affected repositories
MIT	● 0	● 0	● 0	● 5	● 24	● 5	webStore +5
Apache-1.0	● 0	● 0	● 0	● 0	● 0	● 5	lightingApp
GPL	● 1	● 3	● 0	● 0	● 5	● 5	trino
Apache-2.0	● 0	● 0	● 0	● 0	● 0	● 0	webStore +2
OBSD	● 1	● 3	● 5	● 0	● 0	● 5	webAppBac..

Figure 32 Debriked Compliance Tool License report

4.2.2 Certification Environment Steps

Certification Pipeline needs to interact with Athens or Málaga 5G Platform environments to prepare some of the NetApp tests. Certification Pipeline needs to certify that Athens and Málaga environments are available to execute the Certification process, and after the Certification process is complete, it need to clean up the environment to make sure that certification of other NetApps can be executed properly.

We have identified the following interactions that will be modelled in the Certification Pipeline as Jenkins steps.

Env Step A: Before starting any tests, Jenkins needs to certify that connectivity to Athens or Málaga platform is working as many tests depend on this connectivity.

Env Step B: Connectivity is required but is not enough. Jenkins will execute a Performance Assessment experiment in the planforms to characterize the performance in which the tests will be executed. Specific thresholds will be defined for bandwidth and latency to guarantee that tests will be executed in an environment that is not restricting or affecting the tests results.

Env Step C: One of the components required to perform NetApp tests is CAPIF Core Function. A clean deployment of CAPIF Core function will guarantee that CAPIF Core Function is not affected by previous interactions or behaviour of other NetApps.

Env Step D: As the CAPIF Core Function has been deployed in Málaga or Athens for testing, in order to guarantee that CAPIF Core Function APIs work properly, a set of functional tests will be executed using Robot Framework to certify that CAPIF Core Function APIs work properly.

Env Step E: Once the CAPIF Core Function tests have been executed, CAPIF Core function database will be cleaned up and prepared for testing with the NetApp. This step prepares the Database for the NetApp tests on CAPIF.

Env Step F: The other component required to perform NetApp tests is NEF Services. Again, a clean deployment of NEF Services will guarantee that NEF Services are not affected by previous interactions or behaviour of other NetApps.

Env Step G: As the NEF Services have been deployed in Athens or Malaga for testing, in order to guarantee that NEF Services APIs work properly, a set of functional tests will be executed using Robot Framework to certify that NEF Services APIs work properly.

Env Step H: Once the NEF Services tests have been executed, NEF Services database will be cleaned up and prepared for testing with the NetApp. This step prepares the Database for the NetApp tests on NEF Services.

Env Step I: After all NetApp tests have been executed, we need to remove NEF Services used for testing to leave the environment clean from the certification tests execution. This step will Destroy NEF Services deployed in Athens or Malaga to clean the environment.

Env Step J: Similarly, we need to remove CAPIF Core Function used for testing to leave the environment clean from the certification tests execution. This step will Destroy CAPIF Core Function deployed in Athens or Malaga to clean the environment.

5 MARKETPLACE

5.1 MARKETPLACE OVERVIEW/INTRO

Evolved-5G Marketplace is an online, cross-industry API Marketplace that enables developers, entrepreneurs and businesses to come together to create, discover and integrate services by consuming the NetApps that have been created in the Evolved-5G ecosystem.
The Marketplace platform provides support to its users by a forum. The forum allows NetApp creators, NetApp consumers and Visitors ask questions and exchange knowledge with other marketplace users and get support by marketplace administrators.
In the Evolve-5G context we have identified as NetApp creators the SMEs or individual NetApp developers that implement and release their solutions in the Marketplace. NetApp consumers are SMEs that want to use the released NetApps in order to provide a solution to their end customers.

The Marketplace targets four primary profiles. These profiles represent an approximation of a segment of the marketplace users: “Visitors”, “NetApp creators”, “NetApp consumers” and "Marketplace Administrators".
“Visitors” interact for the first time with the Evolved-5G ecosystem and are able to understand the Evolved-5G offering and its value propositions. They can use the marketplace to explore the product catalogue, view promotional material related with each netapp, request support through the marketplace’s forum or seek technical advice in order to participate in the marketplace as a NetApp creator or as a NetApp consumer. On the other hand, “NetApp creators” use the Marketplace as the last step in the life cycle of their NetApp development. By registering, they can connect their certified NetApp with the marketplace, upload marketing and branding material, use wizards to provide CPQ (Configure, Price, Quote) functionality and provide technical support for their users through online tutorials or the forum. “NetApp consumers” are able to use the marketplace to “purchase” NetApps, gain access to their APIs and utilize them. A smart contract is responsible for storing the digital print of the purchase in Ethereum's distributed database. This blockchain transaction acts as a “proof of purchase” and will publicly exist even if the marketplace is no longer in place. Finally, Marketplace Administrators are responsible for all the administration tasks of the platform.

The marketplace has been developed taking into account best practices from the Software Development Life Cycle (SDLC) [29]. SDLC is a set of steps used to create software applications. These steps divide the development process into tasks that can then be assigned, completed, and measured. In the upcoming sections we describe the design phase of the marketplace (including planning, requirements collection, design and prototyping), the development phase of the marketplace (including architecture definition and actual implementation) and the deployment phase of the marketplace (including the dockerization of all of the components) to be available in the Evolved5G OpenShift platform. Finally, we conclude with some screenshots that demonstrate how a NetApp can be released to the Marketplace.

5.2 Marketplace Design

The marketplace design phase included 2 phases: 1) Finalization of the requirements, and 2) Mockups creation.

5.2.1 Phase 1 - Finalization of the requirements

The marketplace is a complex web application with the goal to support a community of different users. During the requirements collection phase, the following have been finalized:

a) Definition of the platform roles: the different user profiles that the platform will support as described in Table 1.

Table 1 List of Platform Roles

#	Role name	Description
1	Visitor	A user that visits MarketPlace landing pages. (S)he can quickly understand how the MarketPlace works and its value propositions
2	NetApp creators (SMEs / Developers)	A NetApp creator can register to the marketplace in order to - manage the NetApp that he implemented - release the NetApp to the marketplace (make it public) - set pricing schemes for the NetApp
3	NetApp consumers	A NetApp consumer is an entity that uses the marketplace to buy one or more NetApps
4	Marketplace administrator	A marketplace administrator has access to key performance indicators (KPIs) of the platform like: 1. number of submitted NetApps 2. number of purchased NetApps

b) Definition of high-level functional areas: that is high level categories of functionality that should exist in order to support the marketplace's scenarios of usage

Table 1 List of Functional Areas

#	***Component / Functional Area***	***Description***
1	Landing pages	The goal of these pages is for a user to easily understand what the Marketplace offers and how it works, as well as to start interacting with the platform.

2	Authentication / Authorization	Allow the user to login / register to the platform
3	Public Product catalogue	Allow the user to search/ view NetApps.
4	NetApps management	Allow the user to manage (create/edit/delete) NetApps
5	NetApps purchases	Allow the user to purchase NetApps and select from different available pricing packages
6	Email notifications	The platform should be able to notify users about events
8	Dashboards	Allow: - NetApp creators to perform management and monitoring of NetApps - NetApp consumers to monitor NetApps status- Marketplace administrators to view high level KPIs of the platform
9	Forum integration	The platform should provide support to its users by a forum. The forum will allow NetApp creators, NetApp consumers and Visitors ask questions and get support by other marketplace users and marketplace administrators.
10	Blockchain integration	The platform should store a digital receipt of a NetApp purchase to a blockchain network. The user will have access to this digital receipt in the Ethereum network.

c) A set of user stories, for each of the role mentioned, above. A user story is an informal, general explanation of a software feature written from the perspective of the end user. Its purpose is to articulate how a software feature will provide value for each of the aforementioned roles. For example, “a visitor should be able to search for NetApps”, “a visitor should be able to view details of a specific NetApp and understand its value propositions” etc.
d) A user interface flow diagram, which illustrates the interactions that users will have with the application. An example of the interface flow diagram is provided below:

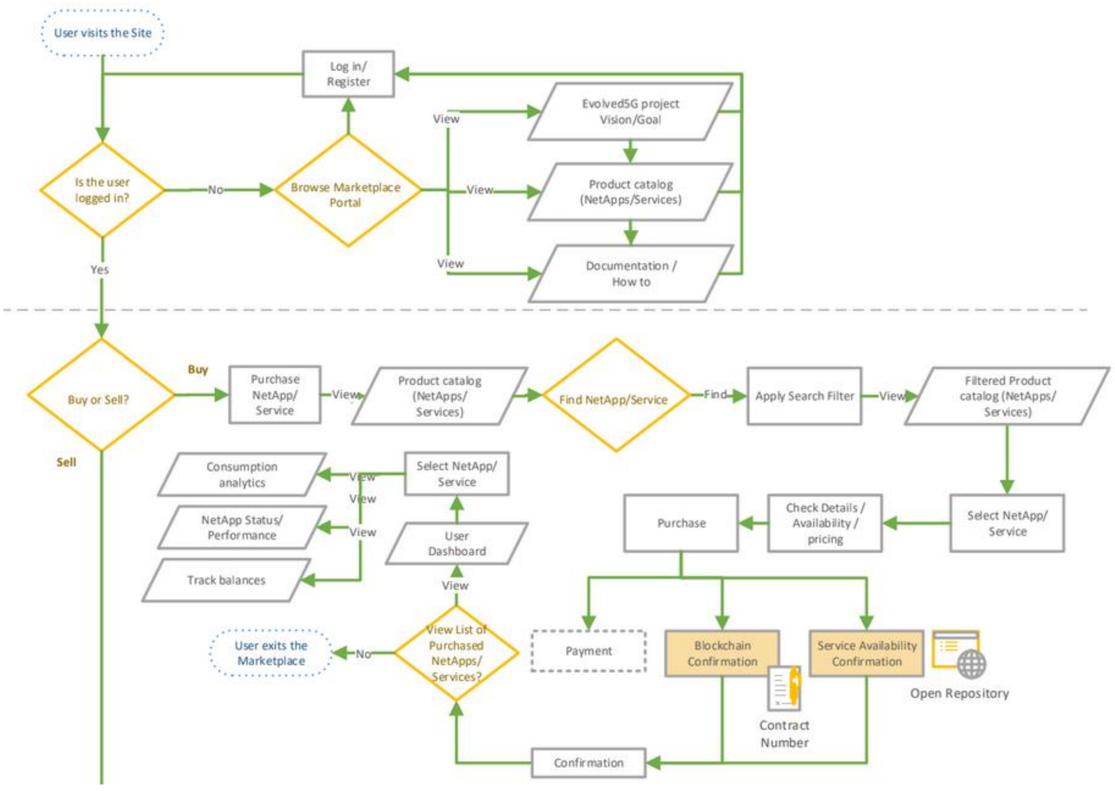


Figure 33 Part of the Evolved-5G user interface flow chart

As depicted in Figure 33 when a user visits the Evolved-5G Marketplace portal and before registering or logging in he/she can browse through the contents of the portal and get some information about the Evolved5G project Vision and goal, read the Technical Documentation and How to' s and even take a look in the product catalog to get some idea of the NetApps already published in the Marketplace. Only a high-level description of the NetApps is visible for the not-registered users.
After a visitor registers the Marketplace and logs-in he/she can selectively buy or sell a NetApp.

5.2.2 Phase 2 - Mockups creation

Based on the requirements listed in the previous section, a set of mockups have been created. A mockup is a static design of a web page or application that features many of its final design elements but is not functional. The mockups have been created with the help of a popular online prototyping tool Adobe XD [30].

Adobe XD helps you craft prototypes that look and feel like the real thing, so you can communicate your design vision and maintain alignment across your team efficiently. Adobe XD is a powerful and easy-to-use vector-based experience design platform that gives teams the tools they need to craft the world's best experiences collaboratively. The creation of the mockups allowed feedback collection about the Marketplace, early in the process, and facilitated fast improvements, since changing a UI Component in a static mockup is much more efficient compared to making the change in the implemented version of each screen.

In Figure 34 Examples of mockups that have been created in the online prototyping tool, you can find a snapshot of all the mockups that have been created using Adobe XD which can also be accessed in the following url: https://xd.adobe.com/view/6cffe34c-ceb6-4ec8-9d53-8d7bc36b3bab-39ac/grid

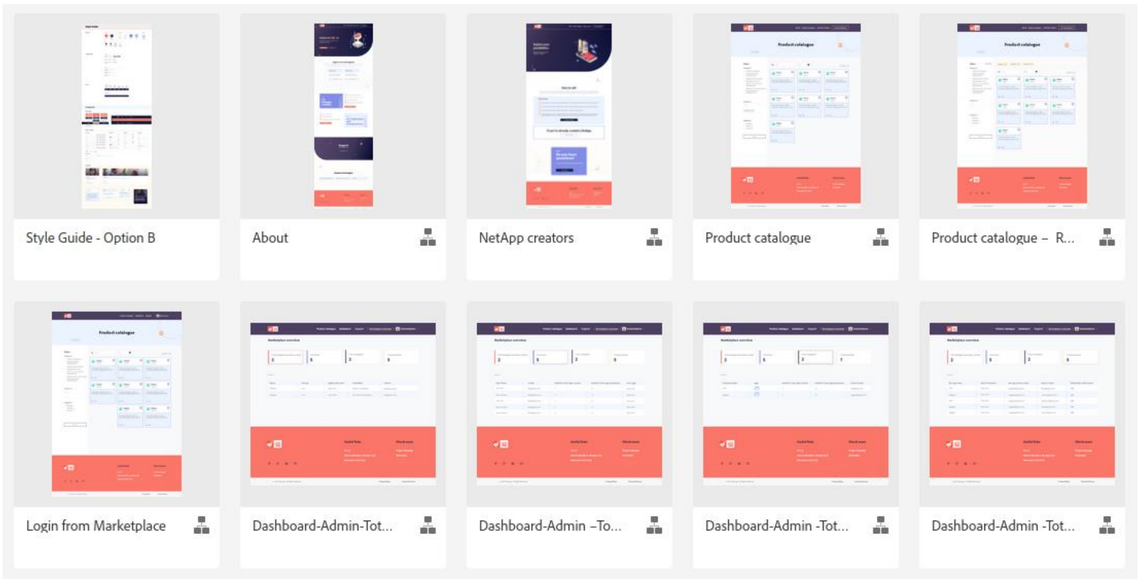
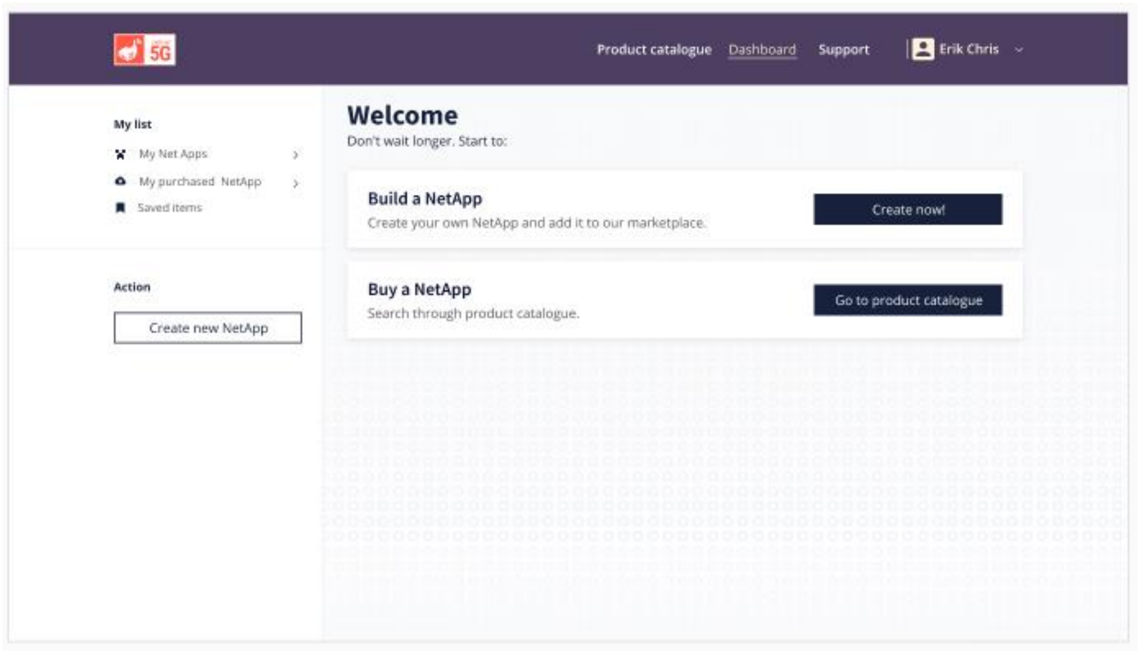


Figure 34 Examples of mockups that have been created in the online prototyping tool



Welcome
Don't wait longer. Start to:

Build a NetApp
Create your own NetApp and add it to our marketplace.

Buy a NetApp
Search through product catalogue.

Figure 35 Example of mockup that demonstrates the Welcome screen that the user sees after login or register

5.3 Marketplace Implementation

The functional areas defined in the previous development phases along with the user stories and the mockups allowed the technical team to define the overall architecture of the marketplace and start the implementation.
In the following diagram a high-level architecture of the Marketplace is shown.

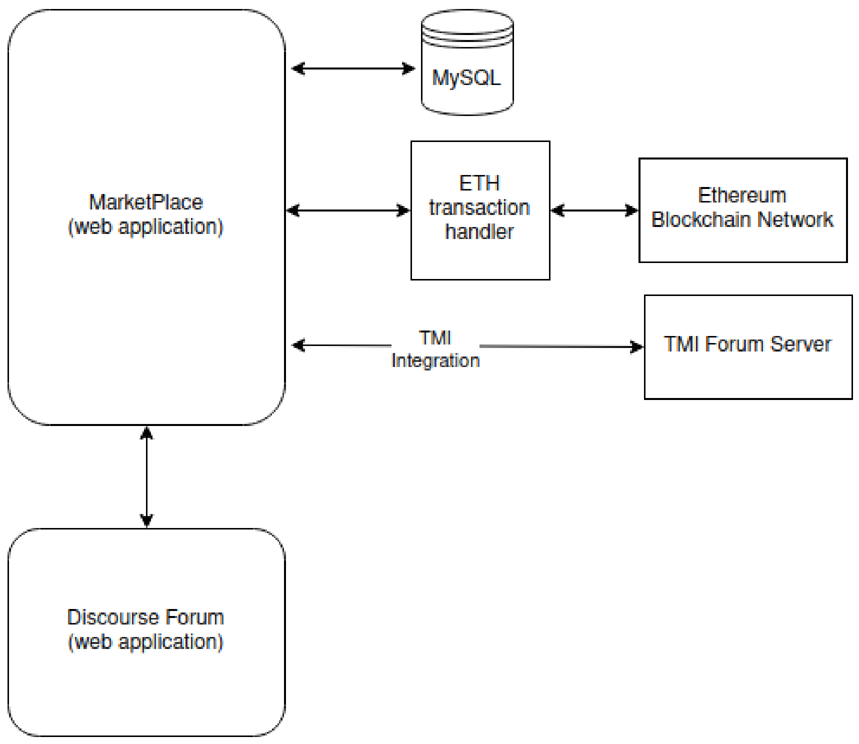

MySQL
MarketPlace (web application)
ETH transaction handler
Ethereum Blockchain Network
TMI Integration
TMI Forum Server
Discourse Forum (web application)

Figure 36 High level architecture of the Marketplace

5.3.1 Marketplace web application

The Marketplace web application supports all the user scenarios for visitors, NetApp creators, NetApp consumers and Marketplace Administrators. Additionally, it acts as an entry point to the Marketplace's forum, where users can find support on various Evolved-5G topics. It is developed using Laravel, one of the most popular frameworks for building web applications, JavaScript Frameworks, VueJs [33] and CSS frameworks (Bootstrap 5 [34] and Sass [35]). The storage layer was implemented in MySQL [36], a widely used relational database management system (RDBMS).

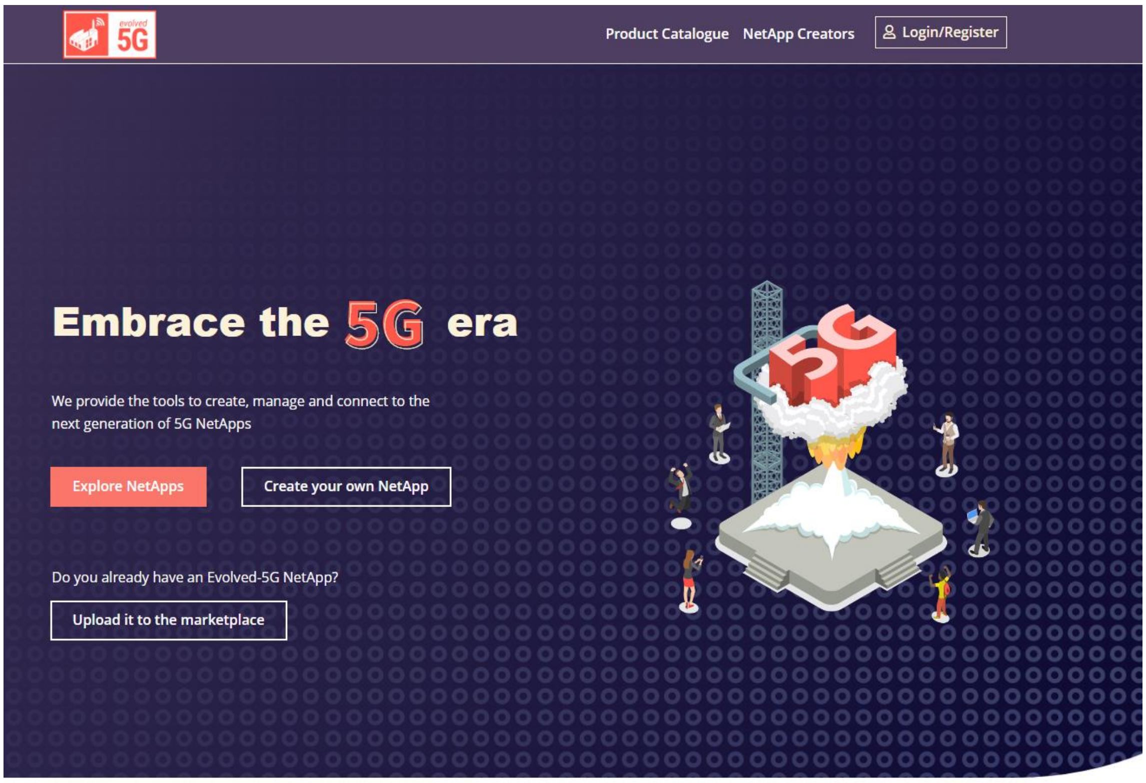
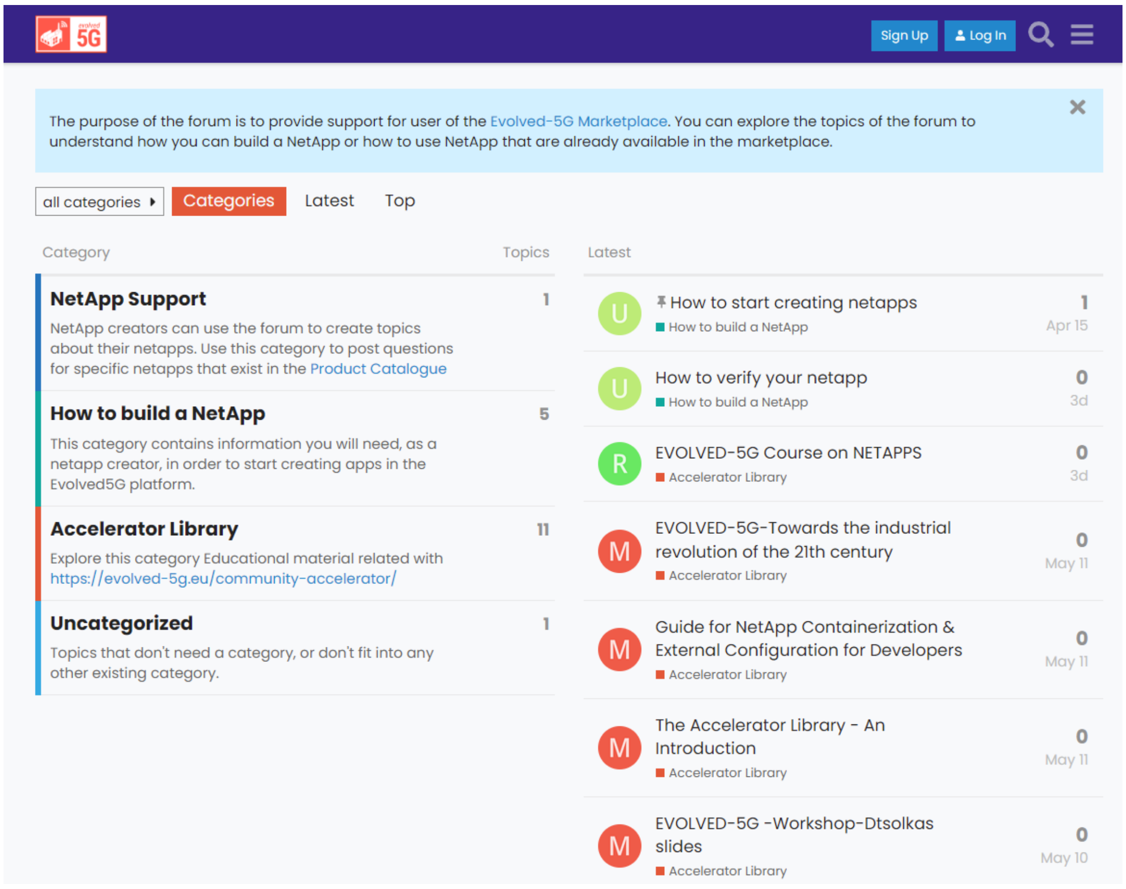

Product Catalogue
NetApp Creators
Login/Register
Embrace the 5G era
We provide the tools to create, manage and connect to the next generation of 5G NetApps
Explore NetApps
Create your own NetApp
Do you already have an Evolved-5G NetApp?
Upload it to the marketplace

Figure 37 Marketplace entry page

The Marketplace backend implements the integration with other Marketplace components like a) the TM forum server and b) the Ethereum transaction handler. Both are explained later in the chapter.
The relevant repository can be found at https://github.com/EVOLVED-5G/marketplace, and the production environment can be found at https://marketplace.evolved-5g.eu/

5.3.2 Marketplace forum

The Marketplace forum has been implemented using Discourse [37], an open-source, powerful platform for communities. A Discourse server has been initialized and its theme has been configured in order to match the branding of the Evolved-5G communities.



The purpose of the forum is to provide support for user of the Evolved-5G Marketplace. You can explore the topics of the forum to understand how you can build a NetApp or how to use NetApp that are already available in the marketplace.

Category	Topics
NetApp Support NetApp creators can use the forum to create topics about their netapps. Use this category to post questions for specific netapps that exist in the Product Catalogue	1
How to build a NetApp This category contains information you will need, as a netapp creator, in order to start creating apps in the Evolved5G platform.	5
Accelerator Library Explore this category Educational material related with https://evolved-5g.eu/community-accelerator/	11
Uncategorized Topics that don't need a category, or don't fit into any other existing category.	1

Latest		
U	How to start creating netapps How to build a NetApp	1 Apr 15
U	How to verify your netapp How to build a NetApp	0 3d
R	EVOLVED-5G Course on NETAPPS Accelerator Library	0 3d
M	EVOLVED-5G-Towards the industrial revolution of the 21th century Accelerator Library	0 May 11
M	Guide for NetApp Containerization & External Configuration for Developers Accelerator Library	0 May 11
M	The Accelerator Library - An Introduction Accelerator Library	0 May 11
M	EVOLVED-5G -Workshop-Dtsolkas slides Accelerator Library	0 May 10

Figure 38 Forum and accelerator entry page

A NetApp creator can use the forum in order to create topics about their NetApps and to provide relevant support. Visitors of the Evolved-5G Marketplace can use the forum to understand how to build a NetApp and to get relevant support. Finally, educational material can be found in the forum related with the "Community Accelerator"of Evolved5G [38].
The online environment of the forum is available at https://forum.evolved-5g.eu/ and is integrated with the web application, that is, various screens in the web application point to relevant topics in the forum (e.g., when a user seeks for support for a specific NetApp, or when a user needs to find out how to build a NetApp).

5.3.3 ETH Transaction Sender / Blockchain integration

The blockchain integration in the marketplace was driven by the need to create a digital signature of a purchase, when a NetApp consumer purchases a NetApp, and to store it in a public distributed database. This ledger acts as a "proof of purchase" and will exist publicly even if the marketplace is no longer in place. The Ethereum Blockchain Network covers this need.

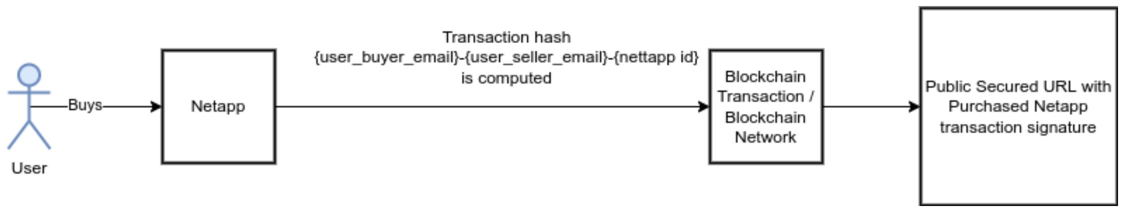

User
Buys
Netapp
Transaction hash
{user_buyer_email}-{user_seller_email}-{nettapp id}
is computed
Blockchain Transaction / Blockchain Network
Public Secured URL with Purchased Netapp transaction signature

Figure 39 High level overview of the blockchain integration

Every time a user buys a NetApp via the marketplace, a digital signature (hash string) is created. For example, the user with email "*buyer@test.com*" buys a NetApp with id "*123*", that was created by the user "seller@test.com". The digital signature is a hashed version of the string: "buyer@test.com-buyer@*test.com-123"*. This signature, after being hashed, is posted to the Ethereum Network.

5.3.4 How it works

An Ethereum Wallet (which belongs to Evolved-5G), is responsible for creating a new Blockchain Transaction. This Transaction has the digital signature of the purchase, as its Input Data

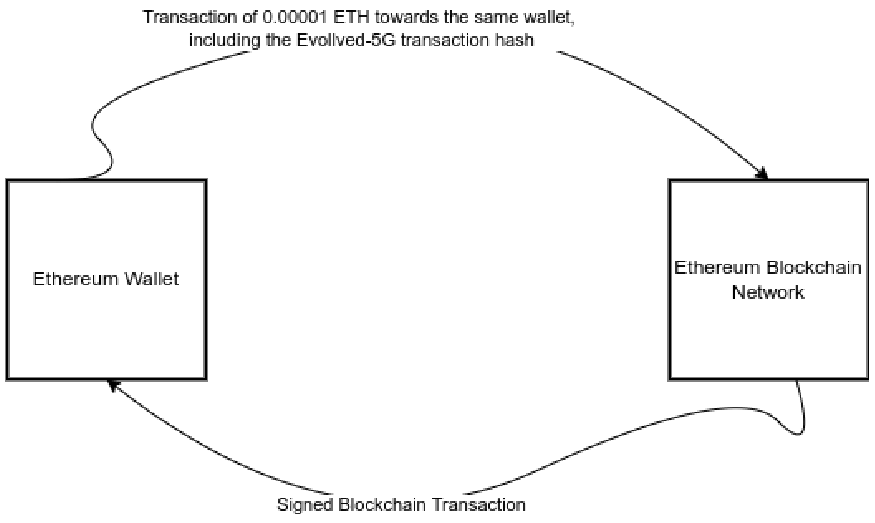
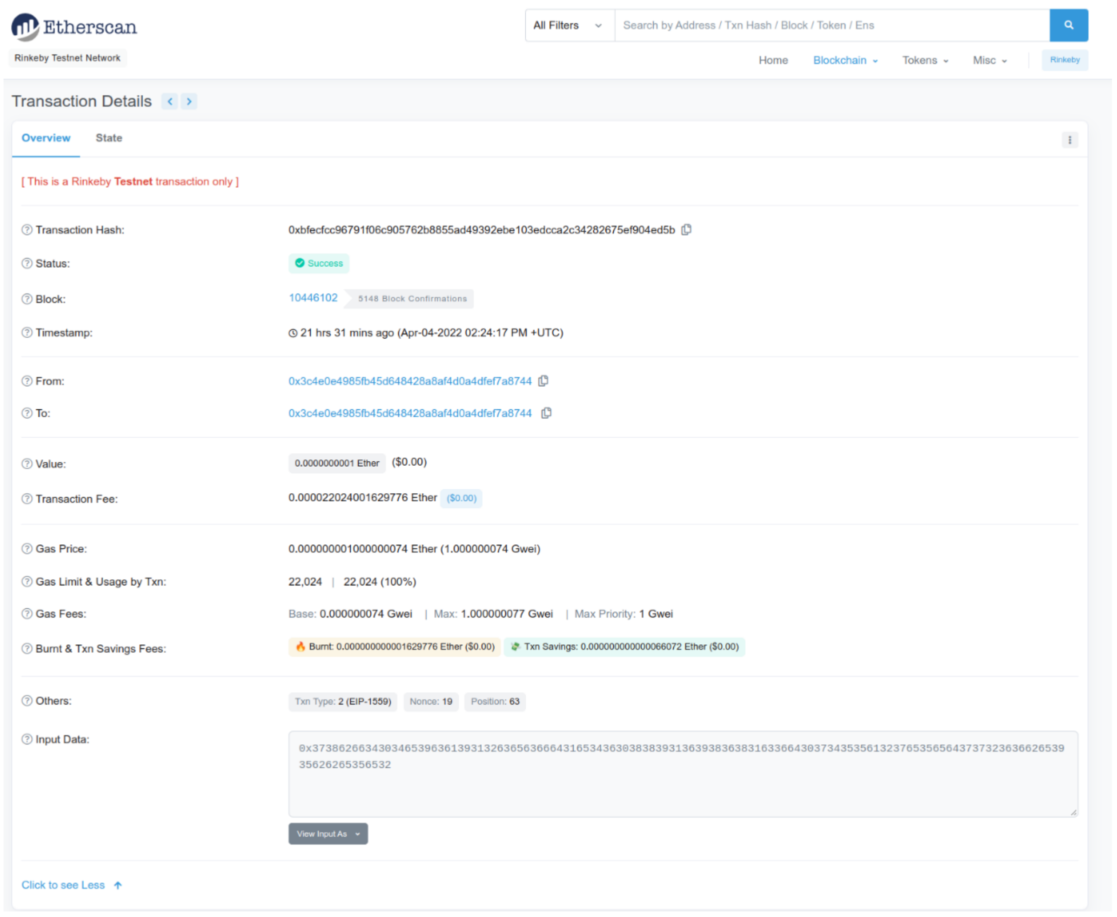

Transaction of 0.00001 ETH towards the same wallet, including the Evollved-5G transaction hash
Ethereum Wallet
Ethereum Blockchain Network
Signed Blockchain Transaction

Figure 40 Blockchain transaction

The transaction is posted to the Ethereum Blockchain Network and is visible to all, containing also the digital signature: For example, at transaction https://rinkeby.etherscan.io/tx/0xbfecfcc96791f06c905762b8855ad49392ebe103edcca2c34282675ef904ed5b, if one clicks the "click to see more" field, they can view the "Input Data" field that contains the hashed version of the digital signature.



[This is a Rinkeby **Testnet** transaction only]

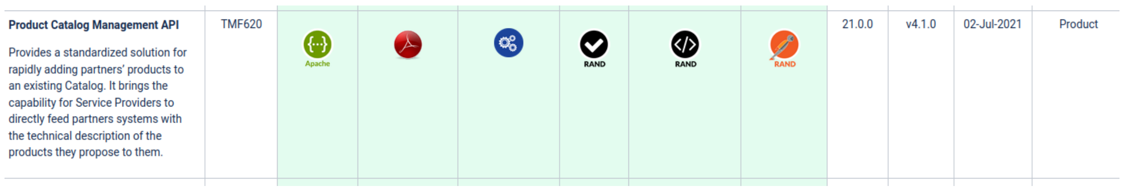
Transaction Hash:	0xbfecfcc96791f06c905762b8855ad49392ebe103edcca2c34282675ef904ed5b
Status:	Success
Block:	10446102 5148 Block Confirmations
Timestamp:	21 hrs 31 mins ago (Apr-04-2022 02:24:17 PM +UTC)
From:	0x3c4e0e4985fb45d648428a8af4d0a4dfef7a8744
To:	0x3c4e0e4985fb45d648428a8af4d0a4dfef7a8744
Value:	0.0000000001 Ether ($0.00)
Transaction Fee:	0.000022024001629776 Ether ($0.00)
Gas Price:	0.000000001000000074 Ether (1.000000074 Gwei)
Gas Limit & Usage by Txn:	22,024 | 22,024 (100%)
Gas Fees:	Base: 0.000000074 Gwei | Max: 1.000000077 Gwei | Max Priority: 1 Gwei
Burnt & Txn Savings Fees:	Burnt: 0.000000000001629776 Ether ($0.00) Txn Savings: 0.000000000000066072 Ether ($0.00)
Others:	Txn Type: 2 (EIP-1559) Nonce: 19 Position: 63
Input Data:	0x37386266343034653963613931326365636664316534363038383931363938363831633664303734353561323765356564373732363662653935626265356532

Figure 41 Digital signature is saved as Input Data in the Etherscan

The Blockchain transaction consists of a transfer of Ethereum (0.00000001 ETH => 0.00000001 EUR), using the same wallet as origin and destination. These wallets belong to and are controlled by the Evolved-5G marketplace.
At the end of the procedure, the wallet pays only the transaction costs, for creating and storing the transaction on the Blockchain Network. The pilot currently runs on the Rinkeby Test Ethereum Network, which is cost free. For the Smart Contract implementation, infura.io has been used, which is a 3rd party service that allows free requests towards the Ethereum network for up to 100K requests/day.

5.3.5 TM forum integration

The purpose of the TM forum integration is to pilot how an integration can be achieved with
TM Forum Open APIs. For this reason, the Product Catalogue Management API, TMF620-API has been integrated to the marketplace, which provides a standardized solution for adding products to a catalogue.



Product Catalog Management API Provides a standardized solution for rapidly adding partners' products to an existing Catalog. It brings the capability for Service Providers to directly feed partners systems with the technical description of the products they propose to them.	TMF620	Apache			RAND	RAND	RAND	21.0.0	v4.1.0	02-Jul-2021	Product

Figure 42 TMF620 API - Product Catalogue Management API

A TM Forum server has been initialized and the Marketplace backend utilizes the forum server in order to a) store the categories of the NetApps, and b) store pricing info about the NetApps.

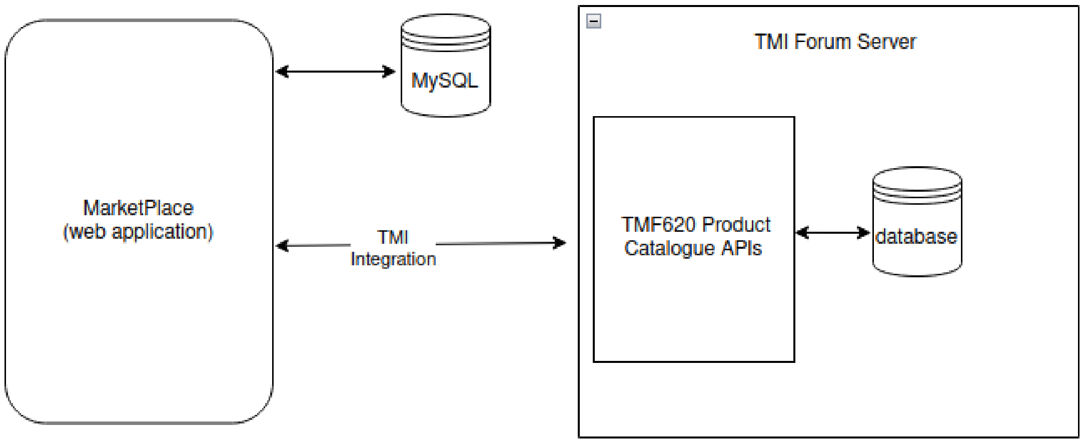


Figure 43 High level architecture of the TM Forum integration

5.4 Marketplace deployment

The marketplace will be deployed in the OpenShift platform. For this purpose, all Marketplace components have been dockerized:

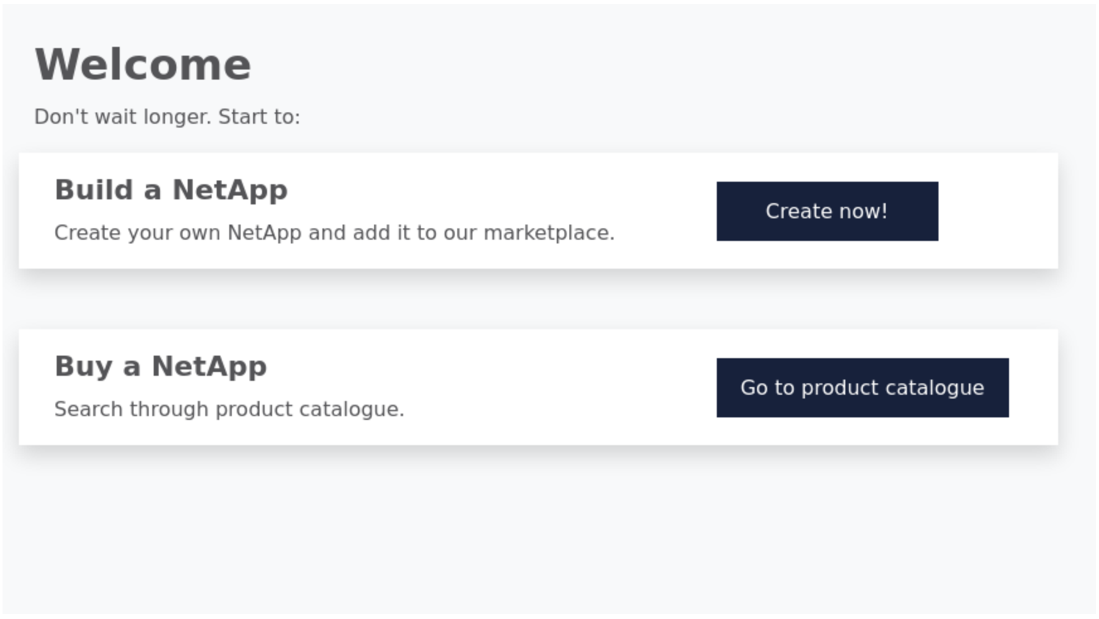
a) The web application (related repo: https://github.com/EVOLVED-5G/marketplace)
b) The blockchain component responsible for sending transactions to the Ethereum network (related repo: https://github.com/EVOLVED-5G/marketplace-blockchain-integration)
c) The TMI forum server that contains the TMF60-API, Product Catalogue Management API endpoints (related repo: https://github.com/EVOLVED-5G/marketplace-tmf620-api
d) The Discourse Forum (related repo: https://github.com/discourse/discourse_docker)

Additionally, staging environments for testing purposes of upcoming releases, are available at:

a) Marketplace: https://marketplace.evolved-5g.eu/
b) Forum: https://forum.evolved-5g.eu/

5.5 NetApp Release to Marketplace

"NetApp creators" use the Marketplace as the last step in the life cycle of their NetApp. Once registered to the platform, they can initialize a wizard in order to release the NetApp to the marketplace.


Welcome

Don't wait longer. Start to:

Build a NetApp

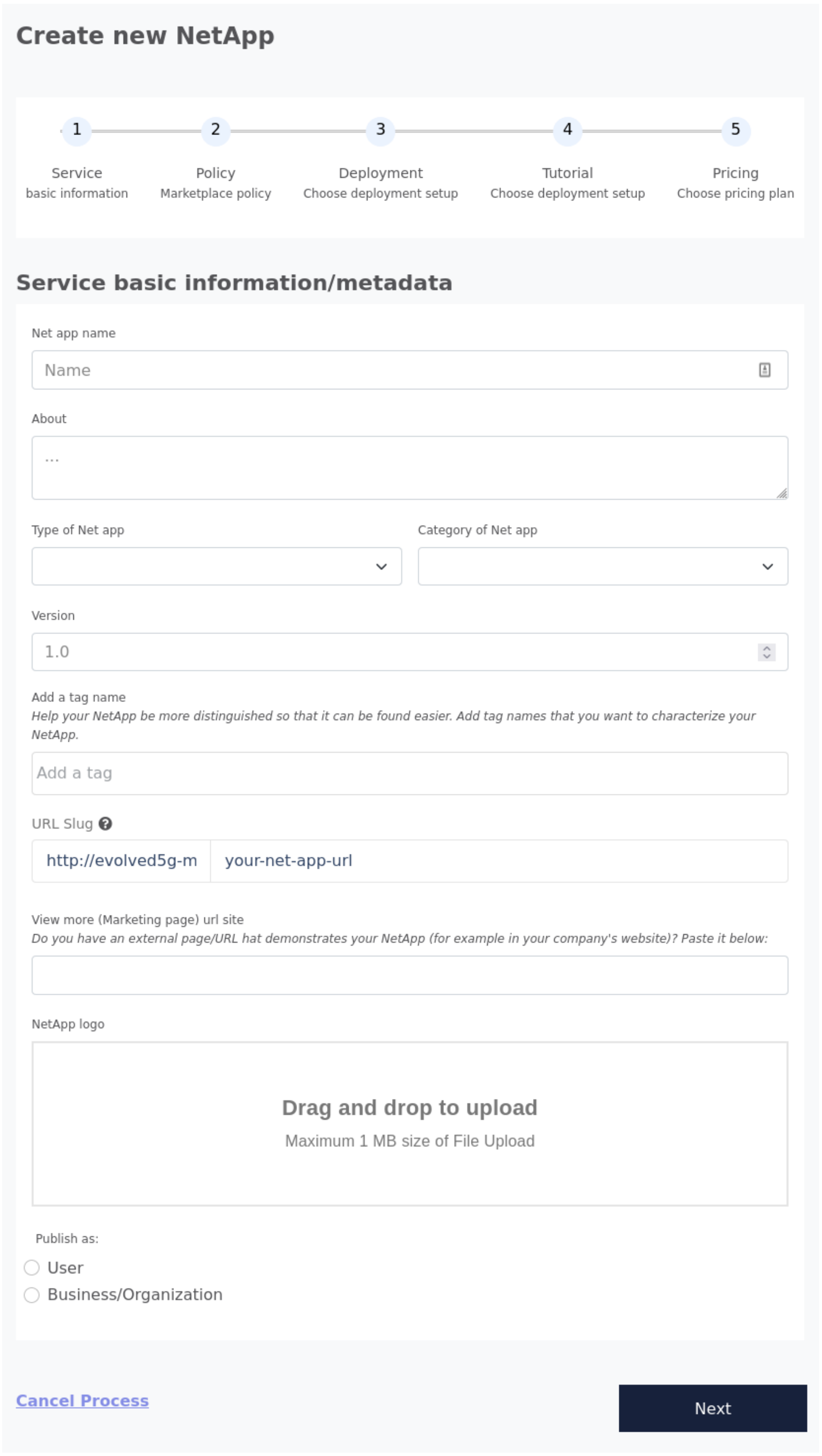
Create your own NetApp and add it to our marketplace.

Buy a NetApp

Search through product catalogue.

Figure 44 Welcome screen - The NetApp creator is invited to start the wizard

During the first step, the NetApp's basic details are provided, e.g., the NetApp name, a short description, the category of the NetApp etc.



Create new NetApp

1 Service
basic information

2 Policy
Marketplace policy

3 Deployment
Choose deployment setup

4 Tutorial
Choose deployment setup

5 Pricing
Choose pricing plan

Service basic information/metadata

Net app name

Name

About

...

Type of Net app

Category of Net app

Version

1.0

Add a tag name

Help your NetApp be more distinguished so that it can be found easier. Add tag names that you want to characterize your NetApp.

Add a tag

URL Slug

http://evolved5g-m your-net-app-url

View more (Marketing page) url site

Do you have an external page/URL hat demonstrates your NetApp (for example in your company's website)? Paste it below:

NetApp logo

Drag and drop to upload

Maximum 1 MB size of File Upload

Publish as:

User

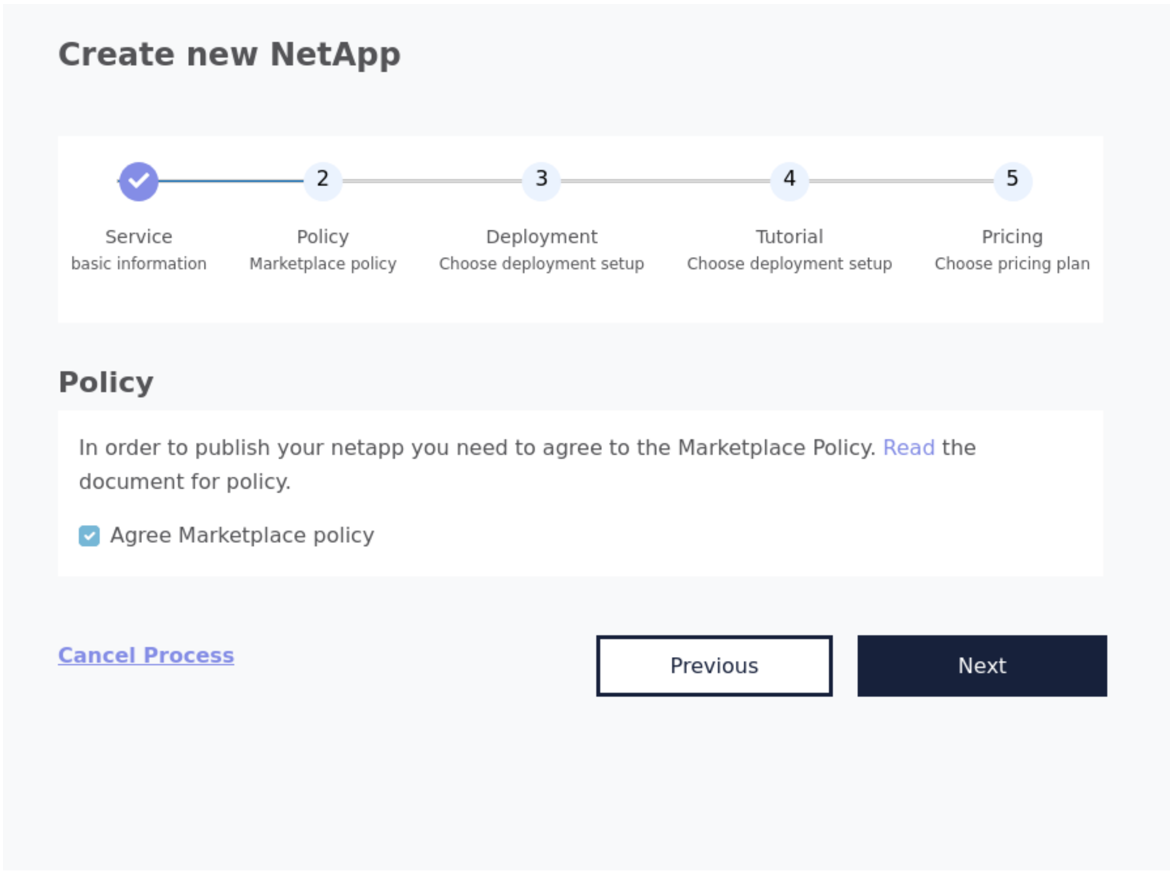
Business/Organization

Cancel Process

Next

Figure 45 Step 1: Net app creator adds basic information

During the second step, the user has to agree to the Marketplace policy/terms and conditions


Create new NetApp

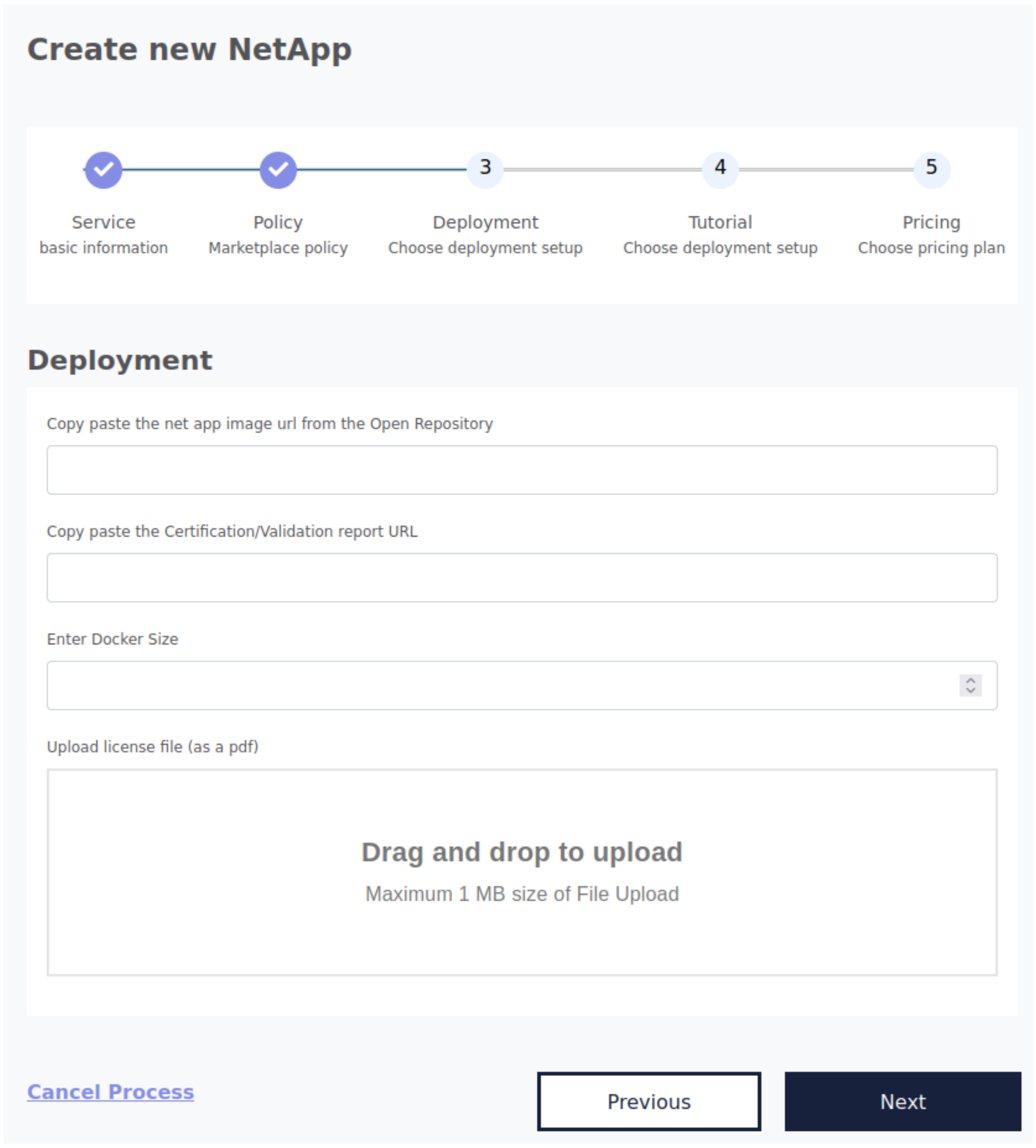
Policy

In order to publish your netapp you need to agree to the Marketplace Policy. Read the document for policy.

✓ Agree Marketplace policy

Figure 46 Step 2: Net app creator agrees to the privacy policy

During the 3rd step, the docker image URL is provided from the open repository, as well as the certification/validation URL from the certification environment. A NetApp creator has also the option to upload a file that contains license information.



Create new NetApp

Deployment

Copy paste the net app image url from the Open Repository

Copy paste the Certification/Validation report URL

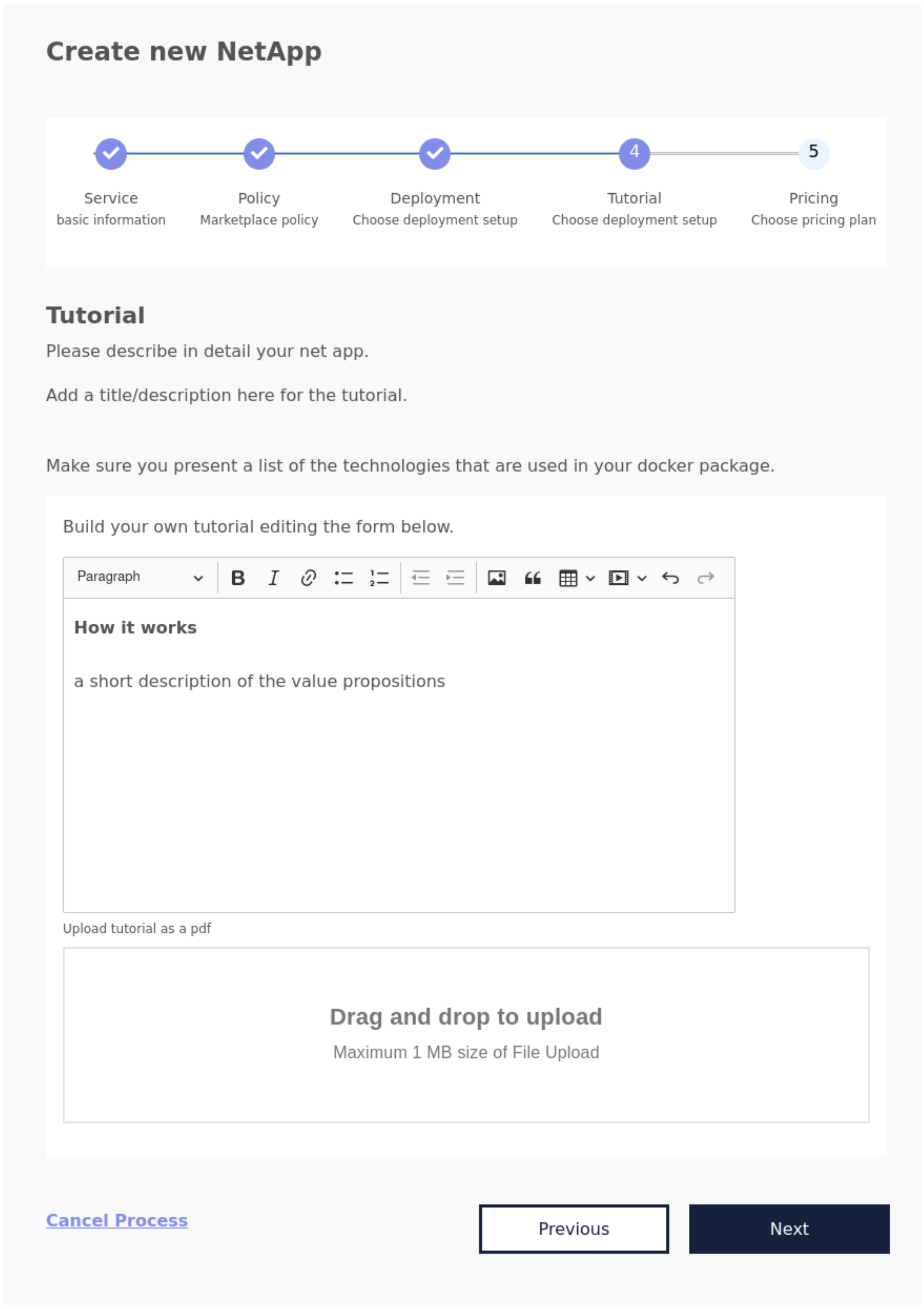
Enter Docker Size

Upload license file (as a pdf)

Drag and drop to upload
Maximum 1 MB size of File Upload

Figure 47 Step 3 - NetApp creator inserts the Docker image url and the certification/validation report URL

During the fourth step, the user provides information about the NetApp. The goal is to provide a tutorial of usage, describing how someone can use the NetApp.


Create new NetApp

Tutorial

Please describe in detail your net app.

Add a title/description here for the tutorial.

Make sure you present a list of the technologies that are used in your docker package.

Build your own tutorial editing the form below.

How it works

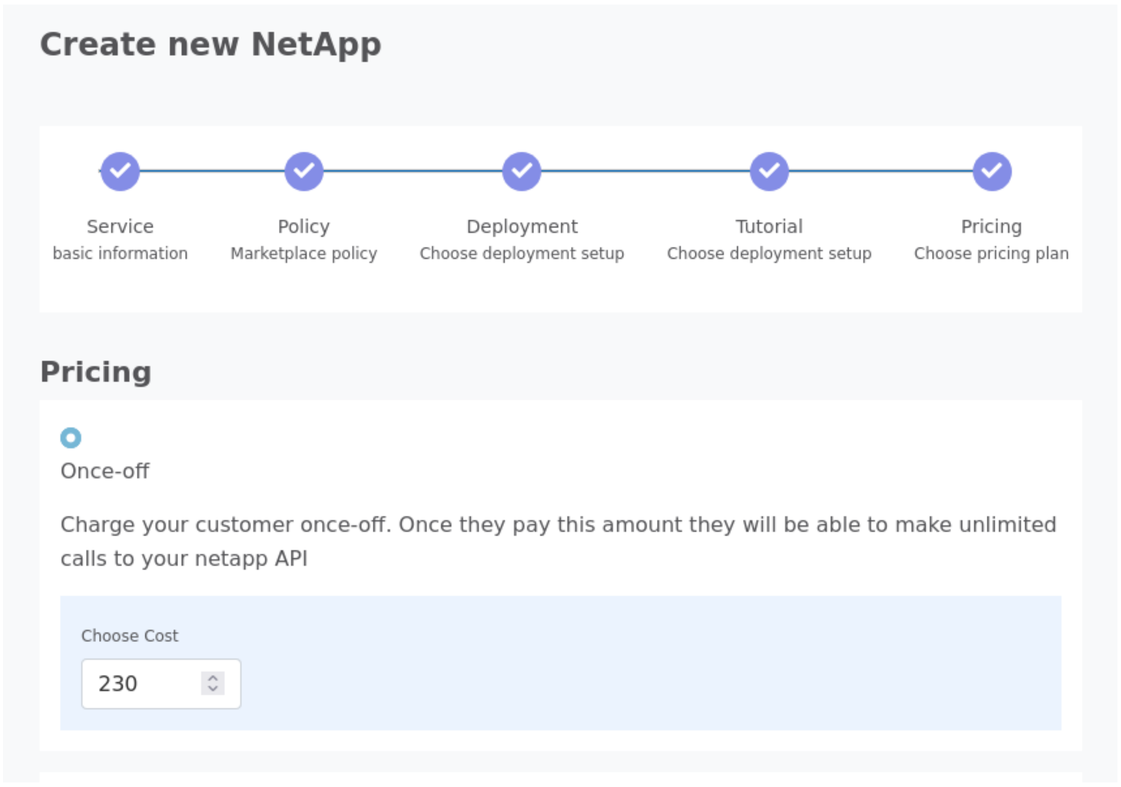
a short description of the value propositions

Upload tutorial as a pdf

Drag and drop to upload
Maximum 1 MB size of File Upload

Figure 48 Step 4 - NetApp creator adds a tutorial of how one can use the NetApp

During the fifth step, user can provide pricing information about the NetApp. Two options are available. In the “Once-off pricing” option, a customer pays a fixed amount in order to start using the NetApp.


Create new NetApp

Pricing

Once-off

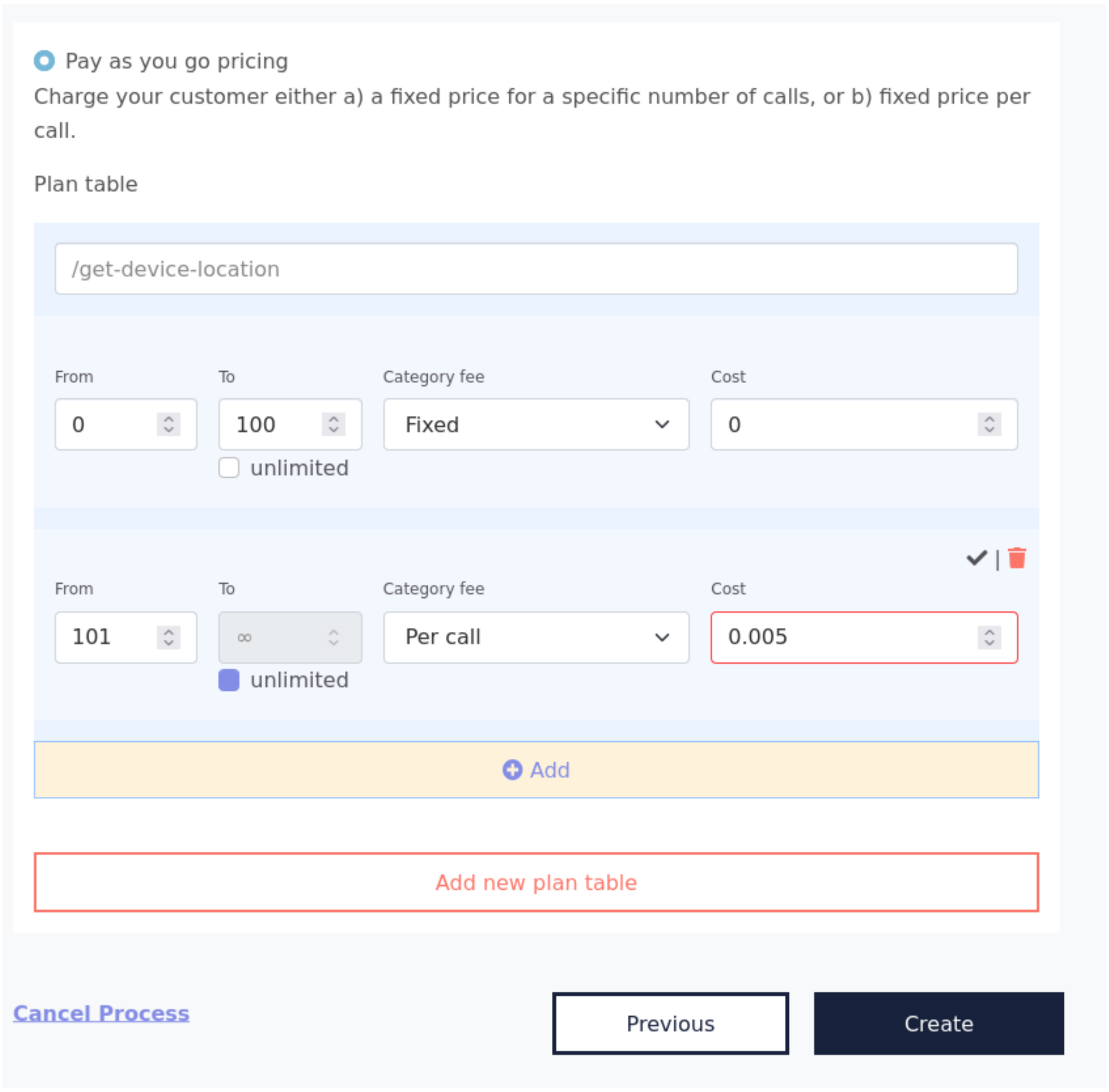
Charge your customer once-off. Once they pay this amount they will be able to make unlimited calls to your netapp API

Choose Cost

230

Figure 49 Step 5 - NetApp creator add pricing information - Once off scenario

In the “pay as go” pricing scenario, a NetApp creator can define a more flexible and detailed pricing plan.
For example, API endpoints of the NetApp can have a relevant cost each time they are consumed.
In the screenshot below the NetApp creator has created a pricing plan, where the first 100 calls to endpoint “/get-device-location” are free, and all of the upcoming calls have a fixed price of 0.005€ per call



● Pay as you go pricing
Charge your customer either a) a fixed price for a specific number of calls, or b) fixed price per call.

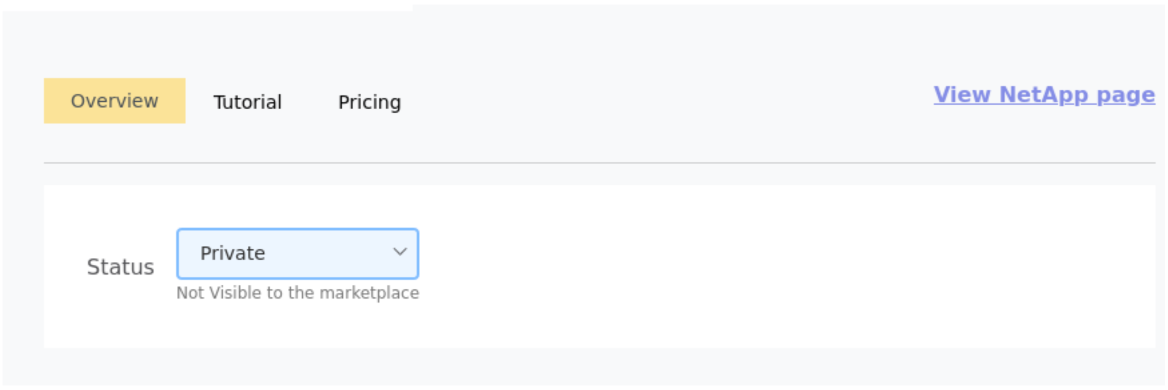
Plan table

/get-device-location

From	To	Category fee	Cost
0	100 (☐ unlimited)	Fixed	0
101	∞ (☑ unlimited)	Per call	0.005

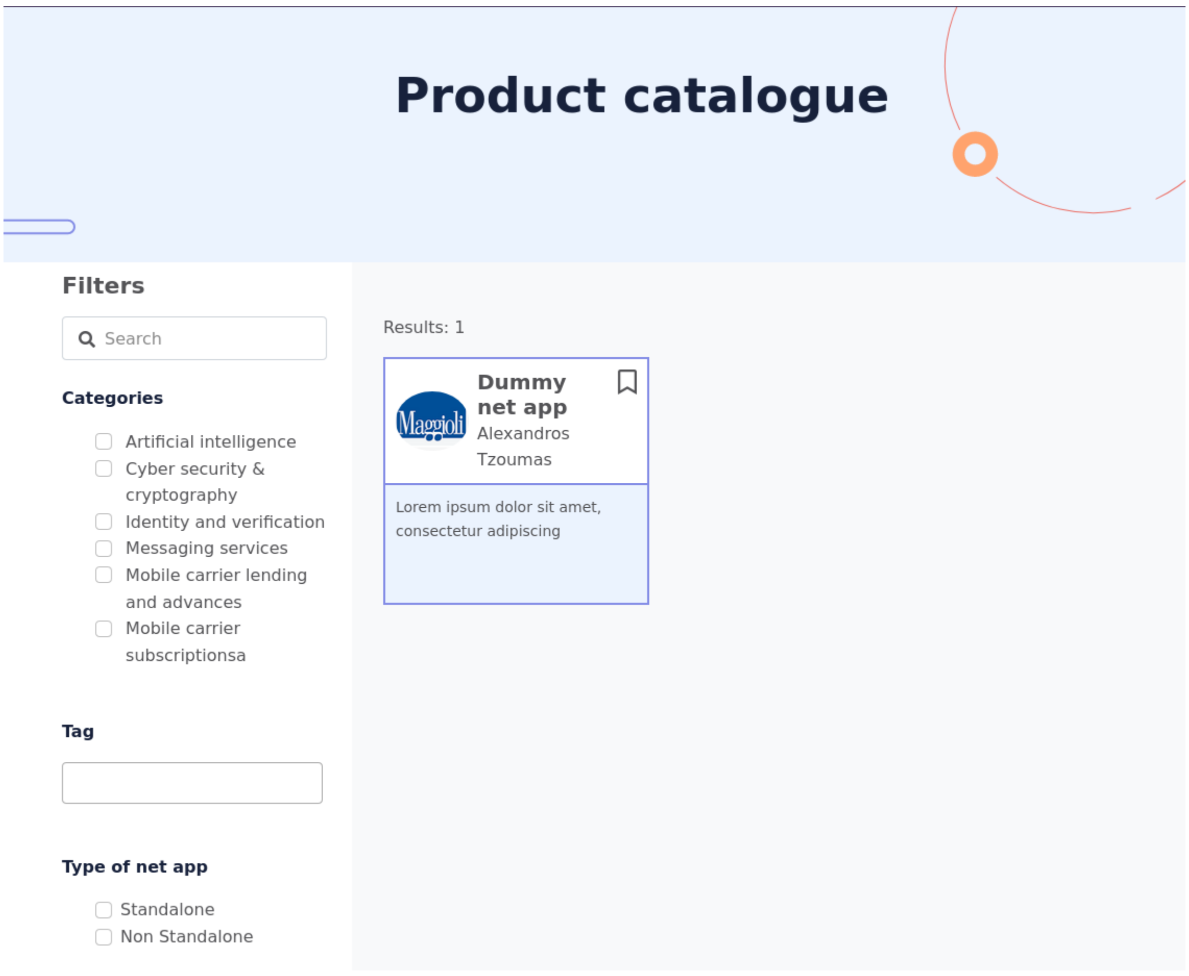
Figure 50 Step 5 - NetApp creator add pricing information - Flexible pricing scenario

Finally, a NetApp creator can save the NetApp and switch its status from "Private" to "Public". This action will make the NetApp visible to the product catalogue of the Marketplace. A landing page of the NetApp, displaying the NetApp information (basic details, tutorials and pricing info), will also be available.



Status: Private
Not Visible to the marketplace

Figure 51 Step 5 - Change status to release to the marketplace



Product catalogue

Filters

Search

Categories

- Artificial intelligence
- Cyber security & cryptography
- Identity and verification
- Messaging services
- Mobile carrier lending and advances
- Mobile carrier subscriptionsa

Tag

Type of net app

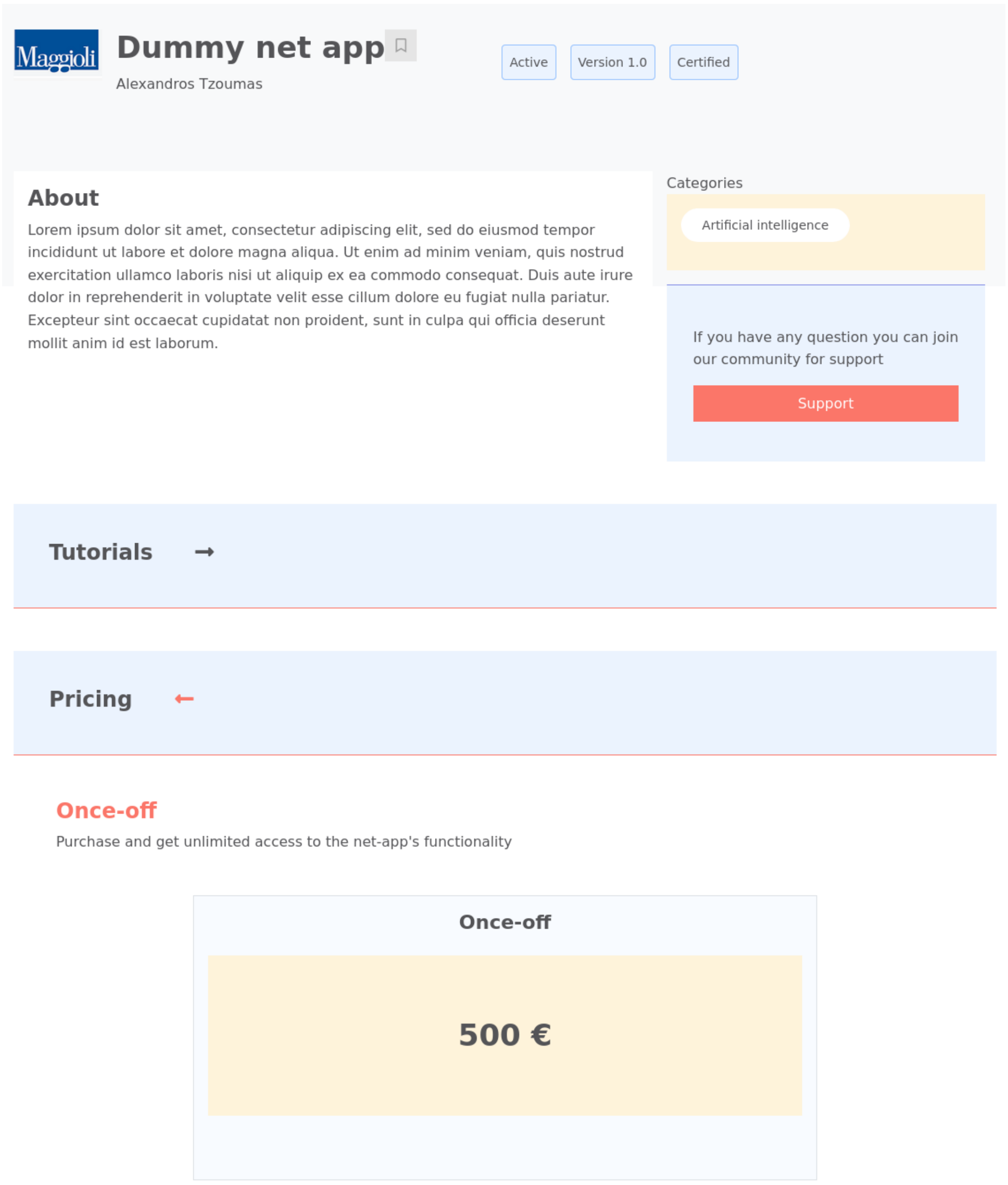
- Standalone
- Non Standalone

Results: 1

Dummy net app
Alexandros Tzoumas

Lorem ipsum dolor sit amet, consectetur adipiscing

Figure 52 NetApp is now available in the Product Catalogue



Dummy net app

Alexandros Tzoumas

Active Version 1.0 Certified

About

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Categories

Artificial intelligence

If you have any question you can join our community for support

Support

Tutorials →

Pricing ←

Once-off

Purchase and get unlimited access to the net-app's functionality

Once-off

500 €

Figure 53 A single public page exists for the NetApp

6 Marketplace and Open Repository Integration

6.1 Integration Design

The Evolved-5G Marketplace is the end destination of a NetApp, for which the development lifecycle has ended. The Marketplace exposes a dockerized image of the NetApp ensuring also that it has previously been certified.

The distribution of a NetApp through the Marketplace is realized through the implementation of the Evolved-5G Open Repository. The Evolved-5G Open Repository is based on the JFrog [3] platform. JFrog facilitates the creation of software repositories hosting artifacts of various forms, including files, packages, containers etc. Thus, the integration of the Open Repository with the Evolved-5G framework is designed in two points: (i) certification process – Open Repository and (ii) Open Repository – Marketplace.

As described in Chapter 4, the certification process ends up with a certified NetApp being uploaded to the Open Repository, in the specific catalogue of certified artifacts. At the same time, a certification report is produced by the process, which is also uploaded to the Open Repository. A unique ID of the NetApp produced by hashing the certified NetApp image is shared with the certification report. This way, a detailed certification report is ensured to refer to a specific NetApp, and in particular to a specific version of the NetApp. Any update of the NetApp requires a fresh certification process that yields a new certification report. The diagram depicted in Figure 54 illustrates the mapping process of the NetApp with its certificate.

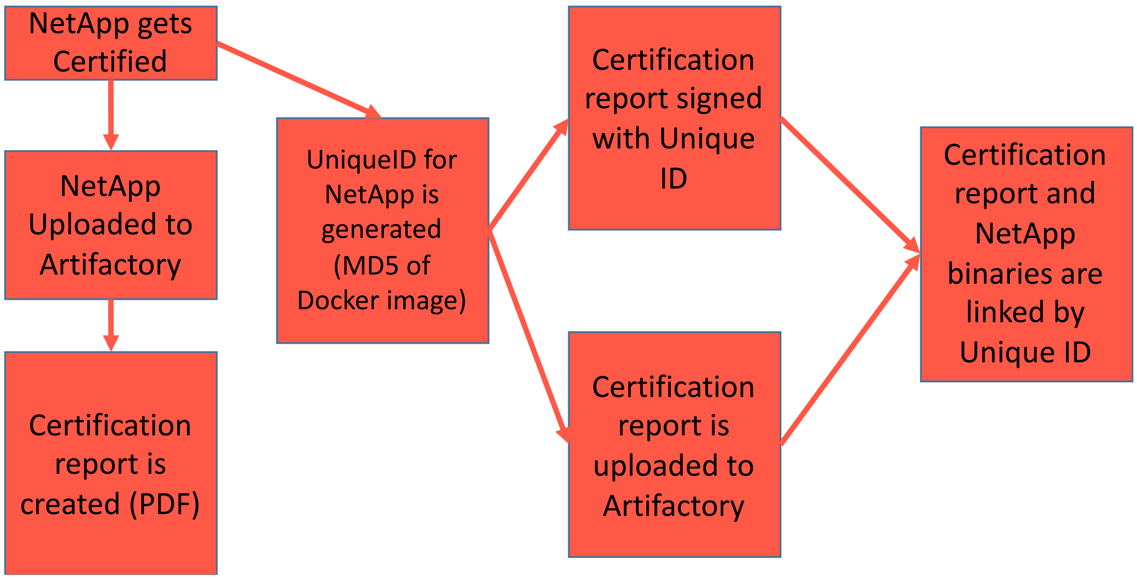

NetApp gets Certified
NetApp Uploaded to Artifactory
Certification report is created (PDF)
UniqueID for NetApp is generated (MD5 of Docker image)
Certification report signed with Unique ID
Certification report is uploaded to Artifactory
Certification report and NetApp binaries are linked by Unique ID

Figure 54 Mapping a NetApp with its certificate

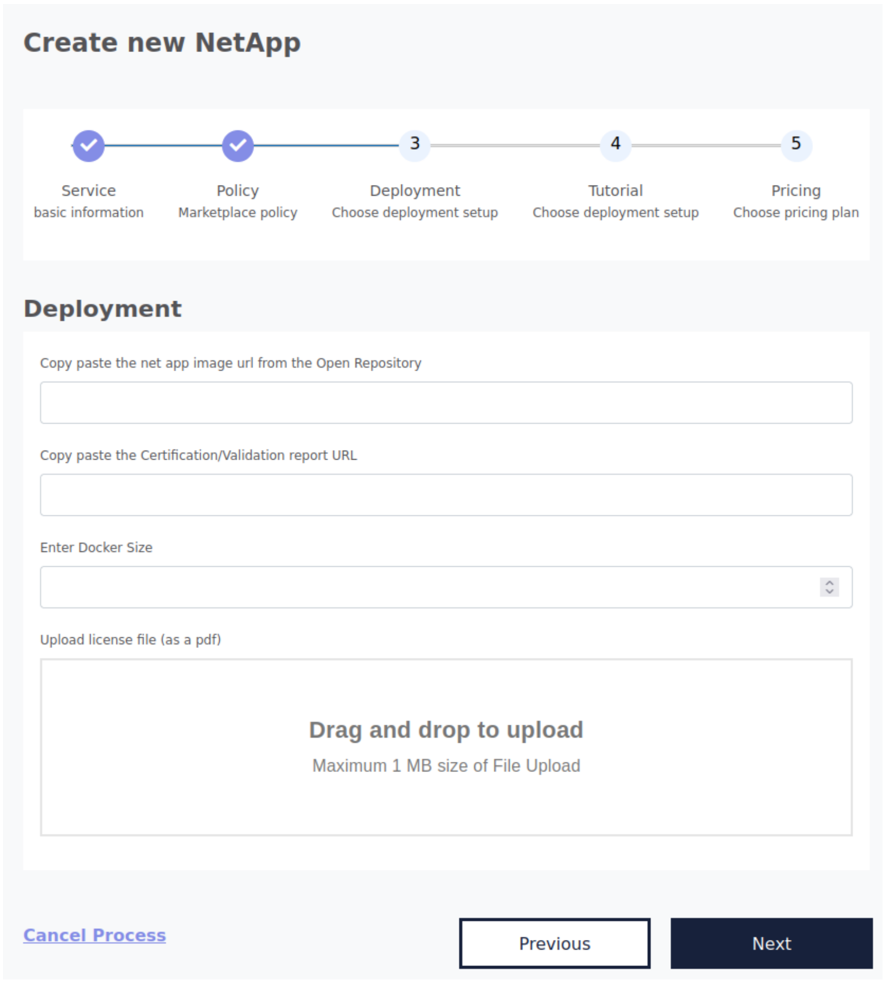
The integration of the Open Repository with the Marketplace is illustrated in Figure 54 Mapping a NetApp with its certificate. The product catalogue of the Marketplace lists all the NetApps that have been certified and uploaded to the Marketplace. Each NetApp is presented with additional descriptive information along with the relevant certification

[3] https://jfrog.com/artifactory/

report. A NetApp in the product catalogue is linked with the Open Repository with a URL corresponding to the NetApp image, as well as a URL of the related certification report. Both are provided by the NetApp developer during the NetApp creation wizard execution in the Marketplace application (Figure 47).

6.2 Implementation

In order to support the Open Repository integration, the "NetApp creation" wizard provides a field that contains the Docker Image URL as produced by the Open Repository. This URL identifies uniquely the dockerized version of the NetApp and is provided to the NetApp creator by the Open Repository environment. The NetApp creator is expected to simply copy-paste the URL at this field. The marketplace will validate the NetApp's existence at the Open Repository, as a requirement before proceeding to next steps.


Create new NetApp

Service – basic information
Policy – Marketplace policy
3 Deployment – Choose deployment setup
4 Tutorial – Choose deployment setup
5 Pricing – Choose pricing plan

Deployment

Copy paste the net app image url from the Open Repository

Copy paste the Certification/Validation report URL

Enter Docker Size

Upload license file (as a pdf)

Drag and drop to upload
Maximum 1 MB size of File Upload

Figure 55 NetApp creation wizard - Step 3 asks the user for a valid Open Repository URL that contains the docker image

7 Conclusion and next steps

This deliverable presented the work performed in the context of WP3, and more specifically, as part of task T3.4, during the first period of the project, concretely from M1 to M18.

T3.4 designed and started the development of the marketplace and the specification of certification process that NetApps will have to go through in order to be certified. A detailed description of this certification process and status of the marketplace development is provided in this deliverable as well as information about the current stage of its implementation.

Regarding next steps, it must be taken into account that some tools need to be refined in order to be integrated in the CICD and some new functionalities will be added to Marketplace as part of T3.4. Work on the certification environment and the infrastructure also continues, with the integration of new components and the improvement of existing functionality. All this work will be documented in deliverable D3.4 to be submitted in M30.

REFERENCES

[1] EVOLVED-5G, Deliverable 2.1 "Overall Framework Design and Industry 4.0 Requirements"
[2] EVOLVED-5G, Deliverable 3.4 "NetApp Certification Tools and Marketplace development"
[3] CICD perspective best practices, from https://en.wikipedia.org/wiki/CI/CD.
[4] GitHub, from https://docs.github.com/en.
[5] Docker project, from https://www.docker.com/
[6] AWS Elastic Container Registry, from https://docs.aws.amazon.com/AmazonECR/latest/userguide/Registries.html
[7] Jenkins https://www.jenkins.io/doc/
[8] JFrog Artifactory, from https://www.jfrog.com/confluence/display/JFROG/JFrog+Documentation
[9] Java Virtual Machine (JVM), from https://docs.oracle.com/javase/specs/jvms/se7/html/
[10] Robo Framework, from https://robotframework.org/robotframework/
[11] OpenShift, from https://docs.openshift.com/
[12] OpenConnect, from https://openconnect.github.io/openconnect-gui/
[13] Rescorla, E. and N. Modadugu, "Datagram Transport Layer Security Version 1.2", RFC 6347, DOI 10.17487/RFC6347, January 2012.
[14] OpenShift architecture reference, from http://uncontained.io/articles/openshift-ha-installation
[15] EVOLVED-5G, Deliverable 2.2 "Design of the NetApps development and evaluation environments"
[16] CAPIF standard specification YAMLs. https://github.com/jdegre/5GC_APIs.
[17] EVOLVED-5G, Deliverable 3.1 "Implementations and integrations towards EVOLVED-5G framework Realization (intermediate)"
[18] CAPIF implementation, from https://github.com/EVOLVED-5G/CAPIF_API_Services
[19] 3GPP TS 23.222 Common API Framework for 3GPP Northbound APIs.
[20] ETSI TS 123.222 Common API Framework for 3GPP Northbound APIs
[21] 3GPP TS 33.122 Security aspects of Common API Framework (CAPIF) for 3GPP northbound APIs.
[22] 3GPP TS 29.222 Common API Framework for 3GPP Northbound APIs
[23] EVOLVED-5G, Deliverable 4.1 "5G Exposure Capabilities for Vertical Applications"
[24] Sonarqube, from https://www.sonarqube.org/
[25] Trivy project, from https://aquasecurity.github.io/trivy/v0.28.1/
[26] Debriked project, from https://debricked.com/tools/license-compliance/
[27] NMAP, from https://nmap.org/docs.html
[28] Telnet, from https://docs.microsoft.com/en-us/windows-server/administration/windows-commands/telnet

[29] S, Shylesh, A Study of Software Development Life Cycle Process Models (June 10, 2017). Available at http://dx.doi.org/10.2139/ssrn.2988291
[30] Adobe XD, from https://www.adobe.com/products/xd.html
[31] Laravel, from https://laravel.com/
[32] Laravel, from https://laravel.com/
[33] VueJS, from https://vuejs.org/
[34] Bootstrap, from https://getbootstrap.com/docs/5.0/getting-started/introduction/
[35] Sass, from https://sass-lang.com/
[36] MySQL, from https://www.mysql.com/
[37] Discourse, from https://www.discourse.org/
[38] Community Accelerator, from https://evolved-5g.eu/community-accelerator/