Ref. Ares(2021)2243090 - 31/03/2021

pharaon

PILOTS FOR HEALTHY AND ACTIVE AGEING

Grant Agreement: 857188

D4.1 Developer guidelines & templates – first

This project has received funding from the European Union's Horizon 2020 Research and Innovation Programme under Grant Agreement No 857188.

Document Information

Deliverable number:	**D4.1**
Deliverable title:	Developer guidelines & templates – first
Deliverable version:	1.0
Work Package number:	WP4
Work Package title:	Technology Support Tools
Due Date of delivery:	M16 (March 2021)
Actual date of delivery:	M16 (March 2021)
Dissemination level:	Public (PU)
Type	Other (O)
Editor(s):	Andrej Grguric (ENT) Miran Mosmondor (ENT)
Contributor(s):	Andrej Grguric (ENT) Miran Mosmondor (ENT) Carlo La Viola (UNIFI) Zain Muhammad Bashir (UNIFI) Jorge Juan Rodríguez Vázquez (INDRA) Antonio Sanchez (MIWENERGIA) Ricardo Perez de Zabalza (MIWENERGIA) Jure Lampe (SENLAB)
Reviewer(s):	Jorge Juan Rodríguez Vázquez (INDRA) Jure Lampe (SENLAB) Rafael Maestre (CETEM) Miguel Ángel Beteta (CETEM)
Project name:	Pilots for Healthy and Active Ageing
Project Acronym	PHArA-ON
Project starting date:	01/12/2019
Project duration:	48 months
Rights:	PHArA-ON Consortium

© 2020 PHArA-ON

Document history

Version	Date	Beneficiary	Description
0.1	26.10.2020	ENT	Table of Contents proposal and initial content considerations for each chapter.
0.2	16.12.2020	ENT	Cybersecurity Tools overview input.
0.3	18.01.2021	ENT	DevSecOps lifecycle definition and visualization
0.4	26.01.2021	ENT	Pharaon technical one-stop-shop initial input. Pharaon developers definition and onboarding process.
0.5	11.02.2021	ENT	Update for Technical One-Stop-Shop chapter.
0.6	15.02.2021	ENT	Initial content in Developers Handbook. Major update for Technical One-Stop-Shop chapter. Building methodology chapter added.
0.7	19.02.2021	ENT	Developers Handbook rationality and content organization chapter. Added Considered SDLC tools in Appendix A. Updated tools inputs chapter.
0.8	23.02.2021	ENT	Added DevSecOps Lifecycle chapter and guidelines reference list.
0.9	15.03.2021	ENT	Summary, progress beyond SOTA, Introduction, conclusion. Updates of the Pharaon SDLC chapter.
0.10	19.03.2021	ENT	Introduction and structure, update of SDLC chapters, tools outcomes mapping, Developers' Handbook chapter updates with latest content structure, descriptions, and guidelines.
0.11	23.03.2021	INDRA	Review of the document.
	25.03.2021	ENT	Updated summary, updated Technical One-Stop-Shop chapter, updated tool descriptions in Appendix A and B.
0.12	26.03.2021	SENLAB, UNIFI, MIWENERGIA, ENT	Review of the document. Updated efforts and contributions table.
0.13	29.03.2021	ENT	Final proofing, updated tools descriptions in Appendix A.
1.0	31.03.2021	ENT, CETEM	Revised and finalized document.

© 2020 PHArA-ON

PM efforts per beneficiary having contributed to the deliverable

#	Partner	PM effort in D4.1*
1	Università degli Studi di Firenze (UNIFI)	0.2 PM
7	Asociacion Empresarial De Investigacion Centro Tecnologico Del Muebley La Madera De La Region De Murcia (CETEM)	0.5 PM
10	My Energia Oner Sl. (MIWENERGIA)	0.34 PM
15	Indra Soluciones Tecnologías de la Información, S.L. (INDRA)	0.4 PM
28	Ericsson Nikola Tesla d.d. (ENT)	3.5 PM
37	Senlab, družba za informacijsko tehnologijo, d.o.o. (SENLAB)	0.2 PM
TOTAL	**Pharaon Consortium**	**5.14 PM**

***Note:** Efforts are related to contributions to this deliverable report and to Pharaon Developers' Handbook on Gitlab Wiki (see Table 5.1 Pharaon guidelines reference list)*

Acknowledgement: This project has received funding from the European Union's Horizon 2020 Research and Innovation Programme under Grant Agreement No 857188.

Disclaimer: The content of this publication is the sole responsibility of the authors, and in no way represents the view of the European Commission or its services.

Executive Summary

This deliverable reports on the work carried out within the task T4.1 “Developer guidelines & templates” of the fourth work package (WP4) of the PHArA-ON (in further text “Pharaon”) project.

The focus of the work reported in this deliverable is to give initial considerations and information of the initial work done with regards to the Technology Support Tools and more concretely with regards to the process and tools related to Software Developing Life Cycle (SDLC) within Pharaon. This document reports on the work being done and residing within Pharaon GitLab which is, alongside with the established overall Pharaon SDLC methodology and processes, the main achievement. More precisely, the main outcome of this task is Developers’ Handbook that contains instructions, guidelines, recommended open-source tools and best practices needed by developers. It is realized using Gitlab Wiki and available to registered developers through Pharaon Gitlab at:

https://gitlab.com/pharaongroup/developers-handbook [1]

[1] Access to Pharaon Developer's Handbook is available to registered Pharaon developers. Contact project coordinator or WP4 leader for more information.

Progress beyond the state of the art

This deliverable reports on the holistic approach taken in Pharaon with respect to overall technical management and developments. Going beyond only WP4 the activities reported here were extensively discussed and agreed with all relevant technical workpackages (mostly WP3 and WP5 but also WP7) and task leaders to establish a common understanding and clearer division of work and responsibilities. Apart from reporting on the extensive survey of the tools applicable to all Pharaon SDLC phases, and more, the shortlisted tools were identified. Pharaon Technical One-Stop-Shop is set up and the collection of all relevant content (including guidelines, links, software, personnel contacts etc.) has started and will be continued continuously in the coming period. Developers' Handbook set up as a wiki page within Pharaon Technical One-Stop-Shop aimed at filtering and summarizing all relevant technical information curtail for the ongoing and future technical activities.

© 2020 PHArA-ON

Contents

© 2020 PHArA-ON

Figures

Tables

© 2020 PHArA-ON

Acronyms & Abbreviations

Term	Description
AIOTES	ACTIVAGE IoT Ecosystem Suite
CMS	Content Management Systems
DAST	Dynamic application security testing
SAST	Static application security testing
SDLC	Software Development Life Cycle
WP	Work Package
WP3	Pharaon WP3 Secure Interoperability Solution
WP4	Pharaon WP4 Technology Support Tools
WP5	Pharaon WP5 Technology Ecosystem Integration
WP6	Pharaon WP6 Ecosystem Evolution
WP7	Pharaon WP7 Pilot Deployment, User Validation, Optimization and Evaluation

© 2020 PHArA-ON

1 Introduction

1.1 Overview

This is a technical document giving the status of the progress in task T4.1 within the WP4 done for its primary consumers: Pharaon internal as well as external technical personnel. The document aims to answer the questions on “what”, “how”, “where” for the technical work of integrating different technologies into the Pharaon ecosystem.

1.2 Relation to other tasks and deliverables

The main inputs to this deliverable come from WP2, WP3, WP5, WP10 technical activities.

The following figure (Figure 1.1) gives an overview of the primary completed deliverables that serve as an input to the work reported in this deliverable as well as main activities that will build upon the results reported here.

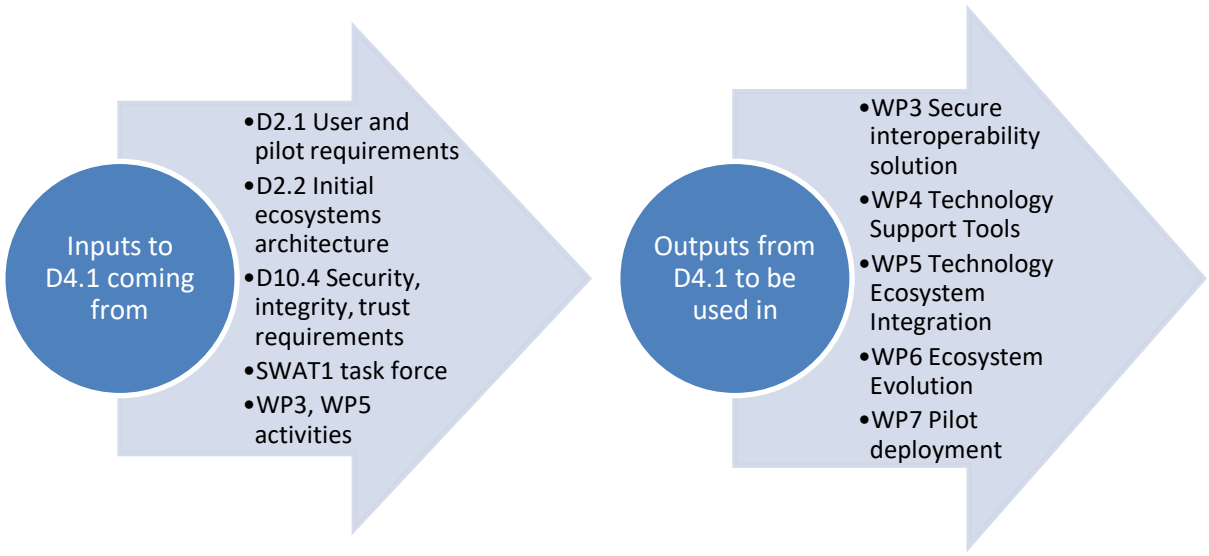

Figure 1.1 Inputs and outputs of D4.1 in relation to other project work

1.3 Structure of the deliverable

After the executive summary and acronyms used in the document, an Introduction of the document is given, including the relation to other project tasks and deliverables.

Chapter 2 presents Pharaon Software Development Life Cycle with emphasis on the initial selection of tools that can be used in each development phase. Chapter 3 describes main target stakeholders of this deliverable, while main outcomes of the deliverable are described in Chapter 4 Technical One-Stop-Shop and most importantly in Chapter 5 that describes Developers’ Handbook. Conclusion and next steps are provided in Chapter 7, while appendixes provide a list of considered tools.

Actual outcome of this deliverable is provided as Gitlab Wiki and available at:

https://gitlab.com/pharaongroup/developers-handbook

© 2020 PHArA-ON

2 Pharaon Software Development Life Cycle (SDLC)

Software Development Life Cycle (SDLC), often referred to as Software Development Process methodology considers a formal control, long time frame, many users, documentation, integrity, and security as vital. All these aspects are suitable for a Pharaon project.

There are different SDLC methodologies that include Agile, Lean, Iterative, Spiral, Waterfall and DevOps. The waterfall model is defined as a sequential process in the development of a system or software that follows a top-down approach and it is considered as a straightforward and linear model[2]. Agile methodology is a practice that promotes continuous iteration of development and testing throughout the software development life cycle of the project where both development and testing activities are concurrent.[2]

DevOps is a set of practices that combines software development and IT operations with aim to shorten the systems development life cycle and provide continuous delivery with high software quality[3]. Some DevOps aspects come from the Agile methodology and DevOps is considered as complementary with Agile software development. Amazon Web Services describes DevOps like this: "DevOps is the combination of cultural philosophies, practices, and tools that increases an organization's ability to deliver applications and services at high velocity: evolving and improving products at a faster pace than organizations using traditional software development and infrastructure management processes."[4] They identify benefits of DevOps as:

- Speed,
- Rapid delivery
- Scale
- Improved collaboration
- Security.

DevSecOps is an evolution of the DevOps to inherently include security aspects in all phases and aims to build awareness of the importance of a continuous security. In short, it evolves DevOps to ensure security remains an essential part of the process.

For Pharaon DevSecOps as the youngest SDLC methodology seems most suitable, as it gives emphasis on cybersecurity. It emerged from applications of lean and agile practices to operations work and emphasizes strong collaboration between development and operations teams that work as one team with shared responsibilities.

2.1 DevSecOps Lifecycle

Typical SDLC process includes phases such as:

[2] DevSecOps pipelines and tools: What you need to know, https://opensource.com/article/19/10/devsecops-pipeline-and-tools

[3] *Mala, D.J. (2019). Integrating the Internet of Things into Software Engineering Practices. Advances in Systems Analysis, Software Engineering, and High Performance Computing. IGI Global. p. 16. ISBN 978-1-5225-7791-1. Retrieved 4 April 2019.*

[4] https://aws.amazon.com/devops/what-is-devops/ [Accessed 27 Oct 2020]

- **Plan** - This phase is related to everything that happens before the developers start writing code and it usually involves requirements and feedback gathering from stakeholders and users. It is the first phase of the DevSecOps lifecycle that involves understanding the vision of the project and envisioning a software based on those perceptions[5].
- **Code** - Related to developing the source code for the system.
- **Build** - Related to automated process which builds the codebase and runs a series of end-to-end, integration and unit tests to identify any regressions.
- **Test** - Once a build succeeds, it is automatically deployed to a staging environment for deeper, out-of-band manual and automated testing in the Test phase. Manual testing can be traditional User Acceptance Testing (UAT) where people use the application as the customer would to highlight any issues or refinements that should be addressed before deploying into production[6].
- **Release** - The point at which developers can say a build is ready for deployment into the production environment. By this stage, each code change has passed a series of manual and automated tests, and the operations team can be confident that breaking issues and regressions are unlikely[6]. The organization then manually or automatically deploys any build that makes it to this stage of the pipeline.
- **Deploy** - In this phase a build is ready and released into production. There are several tools and processes that can automate the release process to make releases reliable with no outage window. In the operation phase the deployed system in the production environment is being managed and making sure that everything is running smoothly. This also includes receiving feedback from users.
- **Operate** - The 'final' phase is to monitor the environment by collecting data and providing analytics on user behavior, performance, errors and more.

These usual SDLC phases are also part of DevSecOps cycle with the emphasis on automation and continuity. In fact, often DevSecOps is associated with the term **Continuous Everything** — Continuous Integration, Continuous Delivery, Continuous Deployment and more. Different projects use different continuous phases, depending on the project. In Pharaon DevSecOps focus is on:

1. Continuous Integration
2. Continuous Delivery
3. Continuous Feedback
4. Continuous Security

[5] https://www.cuelogic.com/blog/devops-lifecycle

[6] https://medium.com/taptuit/the-eight-phases-of-a-devops-pipeline-fda53ec9bba

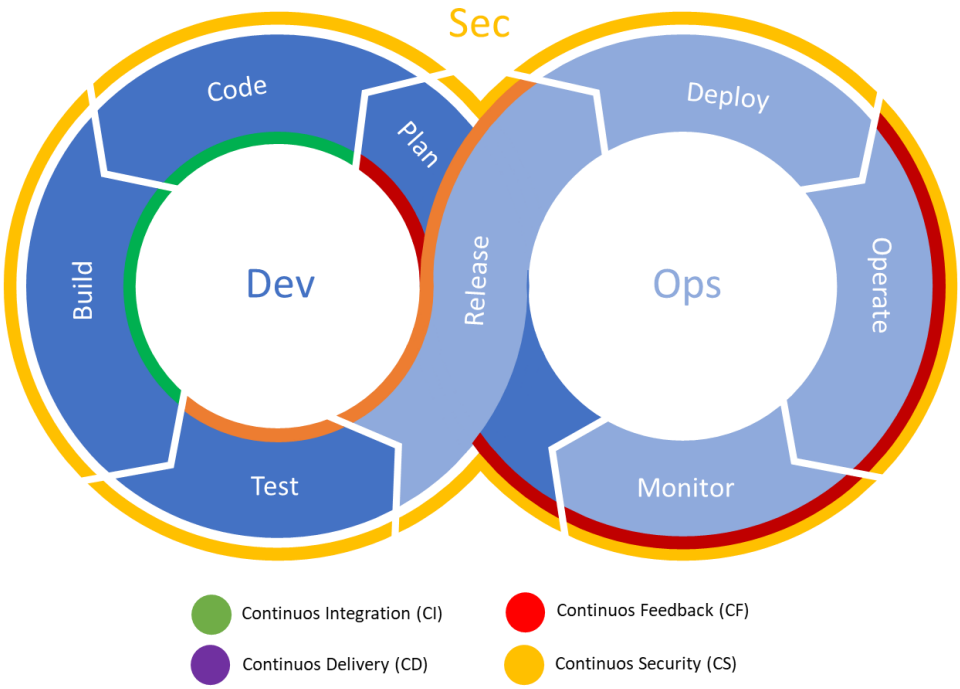

Figure 2.1 Pharaon DevSecOps lifecycle

2.1.1 Continuous Integration

Continuous integration aligns with the Code and Build phases of the DevSecOps pipeline. It's the practice of regularly merging a developer's code into the centralized codebase and conducting automated testing to ensure that no new issues have been introduced. By merging smaller changes more regularly, these issues become smaller and easier to manage, improving overall productivity and sanity.[6]

2.1.2 Continuous Delivery

Continuous Delivery aligns with the Test and Release phases of the pipeline and allows organizations to manually trigger the release of new builds as regularly as they choose.

Continuous Delivery is an extension of Continuous Integration which automates the process of deploying a new build into production. The goals of Continuous Delivery are to[6]:

- Perform automated testing on each new build to verify builds that are ready for release into production and fail those which are not.
- Manage the automatic provisioning and configuration of deployment environments, as well as testing of these environments for stability, performance and security compliance.
- Deploy a new release into production when approved and manually triggered by the organization.

Continuous Deployment is a more advanced version of Continuous Delivery (which makes the reuse of the 'CD' abbreviation more acceptable). The goals are the same, but the manual step of approving new releases into production is removed. In a Continuous Deployment model, each build which passes all of the checks and balances of the pipeline are automatically deployed into production. Due to

© 2020 PHArA-ON

complexity of the Pharaon technical ecosystem, ethical approvals and the fact that large part of user acceptance testing needs to be done manually, the continuous deployment cannot be used in Pharaon project, only Continuous Delivery.[6]

2.1.3 Continuous Feedback

Continuous feedback is fundamental when it comes to continuous delivery in DevOps. An application delivered faster with weekly releases and with higher quality does not guarantee business outcomes nor user satisfaction. Continuous feedback is essential to application release and deployment because it evaluates the effect of each release on the user experience and then reports that evaluation back to the DevOps team to improve future releases.

2.1.4 Continuous Security

Continuous Security is a process of making security as a part of CI/CD process. Most of the organizations especially high performing ones have already implemented CI/CD pipelines to make software release more agile, building software automatically using Continuous Integration tools, and then packaging them in docker images and then containers are run in production using Continuous Deployment tools such as Jenkins, Kubernetes.[7]

In DevSecOps, specific security checks are applied in each

- **Plan**: Execute security analysis and create a test plan to determine scenarios for where, how, and when testing will be done.
- **Code**: Deploy linting tools[8] and Git controls to secure passwords and API keys.
- **Build**: While building code for execution, incorporate static application security testing (SAST) tools to track down flaws in code before deploying to production. These tools are specific to programming languages.
- **Test**: Use dynamic application security testing (DAST) tools to test your application while in runtime. These tools can detect errors associated with user authentication, authorization, SQL injection, and API-related endpoints.
- **Release**: Just before releasing the application, employ security analysis tools to perform thorough penetration testing and vulnerability scanning.
- **Deploy**: After completing the above tests in runtime, send a secure build to production for final deployment.

[7] https://www.xenonstack.com/insights/continuous-security/

[8] Linting is the automated checking of source code for programmatic and stylistic errors. A lint tool is a basic static code analyser

2.2 Pharaon building methodology

For Pharaon a twelve-factor methodology *"for building distributed applications that run in the cloud and are delivered as a service"* will be used in WP4 where it will make sense, as it is suggested for other Pharaon technical workpackages. More info on the methodology can be found on https://12factor.net/. Sometimes not all 12 factors will make sense, but developers will benefit with even applying some of them.

The benefits of twelve-factor design include well defined practices around version control (in Pharaon Git and GitLab are selected), environment configuration, isolated dependencies and their explicit declaration, and others as shown in the following figure (Figure 2.2). On a business level it can be said benefits are scalability, resiliency, CD, maintainability, information security.

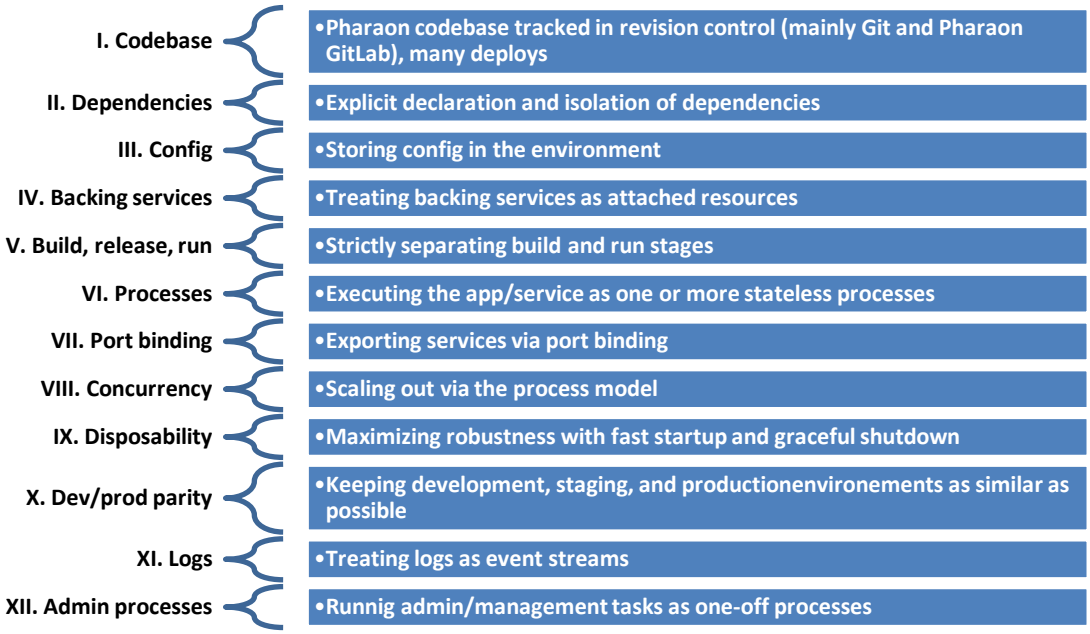

Figure 2.2 Pharaon Twelve-factor apps methodology

With respect to Pharaon DevSecOps it can be said 12 factor app principles are mainly focusing on the build phase but also cover some aspects of the others (e.g. deploy, operate, monitor).

2.3 Input considerations for Pharaon tools

The inputs towards the definition of the Technology support tools are summarized in the following figure (Figure 2.3).

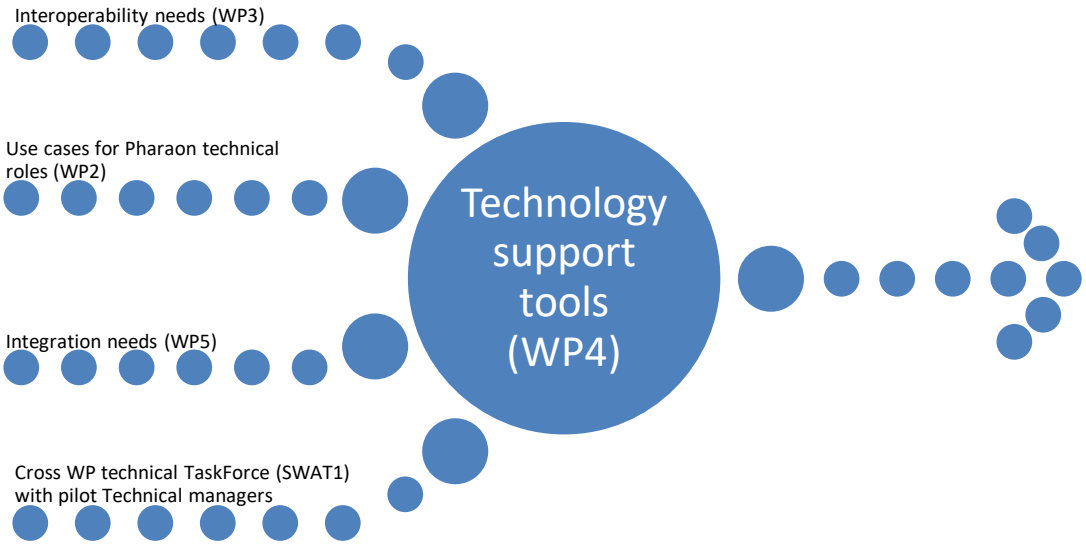

Figure 2.3 Inputs for definition of Technology Support Tools

2.3.1 ACTIVAGE IoT Ecosystem Tools Suite (AIOTES) workshop

Another valuable input for the selection of proper tools for Pharaon SDLC come from the AITOES workshop organised in December 2020 by WP4. AIOTES is ACTIVAGE IoT Ecosystem Suite consisting of a set of Techniques, Tools and Methodologies for interoperability at different layers between heterogeneous IoT Platforms and an Open Framework. ACTIVAGE was an EU project with main objective to build the first European IoT ecosystem across nine deployment sites, reusing and scaling up underlying open and proprietary IoT platforms, technologies and standards, and integrating new interfaces needed to provide interoperability across these heterogeneous platforms[9]. There are some similarities with the objectives of Pharaon project, thus this workshop provided useful insights about the project, ACTIVAGE IoT Ecosystem Suite and various development and deployment tools they used in the project.

[9] https://www.activageproject.eu/activage-project/

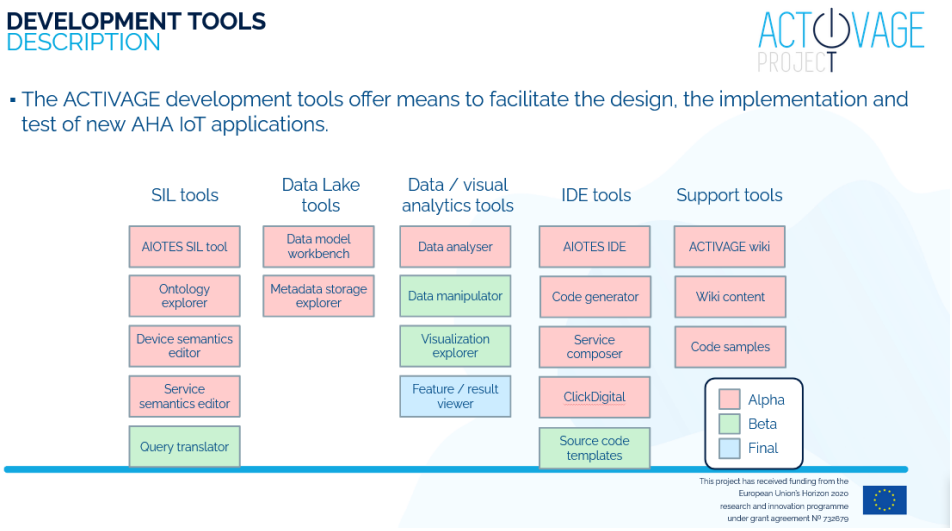

Figure 2.4 Snapshot from AITOES workshop presentation

2.4 Planned Pharaon tools outcomes mapped to DevSecOps phases

Table 2.1 presents mapping of the main Pharaon tools outcomes with related WP to the different DevSecOps phases.

Table 2.1 Pharaon tools outcomes mapped to DevSecOps phases

DevSecOps phase	Short Desc	Related WP	Pharaon outcome
Plan	Involves requirements and feedback gathering from stakeholders and users. It is the first phase of the DevSecOps lifecycle that involves understanding the scope and purpose of the project.	WP1, WP2	Documentation, Wiki,
Design	The purpose of the Design Definition process is to provide sufficient detailed data and information about the system and its elements to enable the implementation consistent with architectural entities as defined in models and views of the system architecture." [ISO/IEC/IEEE 12207:2017]	WP2, WP3	Architecture, program languages, UIs, platforms, Samples, tutorials, discussion forum
Code	Related to developing the source code for system	WP3, WP4	Code standards per programming language, code documentation, code gen, version management Code samples; Pharaon adapters

Build	Related to an automated process which builds the codebase and runs a series of end-to-end, integration and unit tests to identify any regressions.	WP3, WP4	Build management, Continuous Integration
Test	Manual (i.e. UAT) or automated tests. Automated tests might run security scanning against the application, check for changes to the infrastructure and compliance with hardening best-practices, test the performance of the application or run load testing.	WP5	Debug management, Test management, unit test, white box, black box testing, integration test, stress test
Release	The point at which developers can say a build is ready for deployment into the production environment. By this stage, each code change has passed a series of manual and automated tests, and the operations team can be confident that breaking issues and regressions are unlikely. The organization then manually or automatically deploys any build that makes it to this stage of the pipeline.	WP5	Release management
Deploy	In this phase a build is ready and released into production. There are several tools and processes that can automate the release process to make releases reliable with no outage window.	WP7	Config management, installation, instantiation, provisioning; device management, service management
Operate	In the operation phase the deployed system in the production environment is being managed and making sure that everything is running smoothly. This also includes receiving feedback from users	WP7	Update management; data analysis
Monitor	The 'final' phase is to monitor the environment by collecting data and providing analytics on user behavior, performance, errors and more.	WP7	Log management, notifications, alerting, 1st, 2nd, 3rd lines of support

2.5 DevSecOps tools survey

The tools survey conducted was focused on open, largely open-source tools with the survey framework summarized as shown in the following table (Table 2.2).

Table 2.2 Template for tools survey

SDLC phase	Tool category	Tool name	Short Description	License/ Price	Link	Selected for use on Pharaon project	Comment

Tool categories as mapped to SDLC phase are:

- Code:
  - VSC (Version Control System)
  - SCM (Source Code Management)
  - IDEs (Integrated Development Environments)
  - Source code editors
- Build, package
  - Build automation tool
  - Automate dependency updates
  - Package (Binaries) repository
  - General purpose repository
- Test
  - Code test frameworks
  - Code quality tools
  - Data generators
  - Load test tools
  - API testing tools
  - Security testing tools
  - Penetration testing
- Release
  - CI/CD tools
- Deploy
  - Containers
  - Web application servers
  - Configuration management tools
- Operate
  - Secret Management
  - Service Discovery
  - IAM (Identity and Access Management tools)
  - Antivirus
  - Network analyzers
  - Network defense
  - Password security auditing and recovery
  - Web vulnerabilities scanning
  - Network Intrusion detection and prevention
- Monitor
  - Logging tools
  - IT infrastructure Monitoring

© 2020 PHArA-ON

- APM (Application Program Monitoring)

- Other
  - Anonymization tools
  - Interoperability tools

The results were collected in spreadsheet, meant to be continuously updates and were also shared with other healthcare projects as a part of the "Health and care cluster WG4 architectures, standardization and reusable components" lead by ENT from Pharaon side, supported by CSA OpenDei[10].

The reasons for focusing mostly on open source include also the survey results on trends in professional development teams as shown in the following figure (Figure 2.5) confirming open source helps developers get more work done, more quickly and cost-effectively.

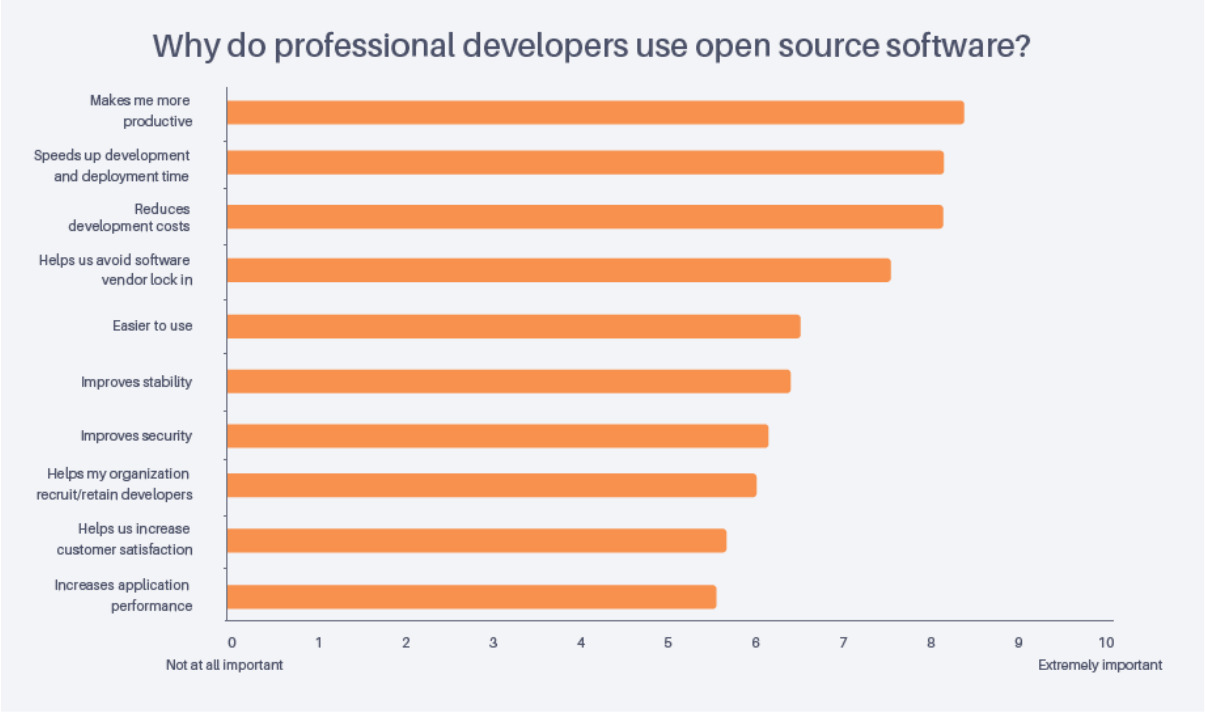

Figure 2.5 Why do professional developers use open source software? [image from [11]]

Additional reasons for focusing on open source additionally include sustainability, transparent governance, available documentation, code quality, avoidance of vendor lock-in and legal issues.

[10] https://www.opendei.eu/healthcare-sector/

[11] Survey report: Key open source usage trends in professional development teams, Tidelift, April 2019

2.6 Shortlisted DevSecOps tools for Pharaon

The following figure (Figure 2.6) gives an overview of the shortlisted tools already used or in close consideration to be used in Pharaon with regards to different SDLC phases.

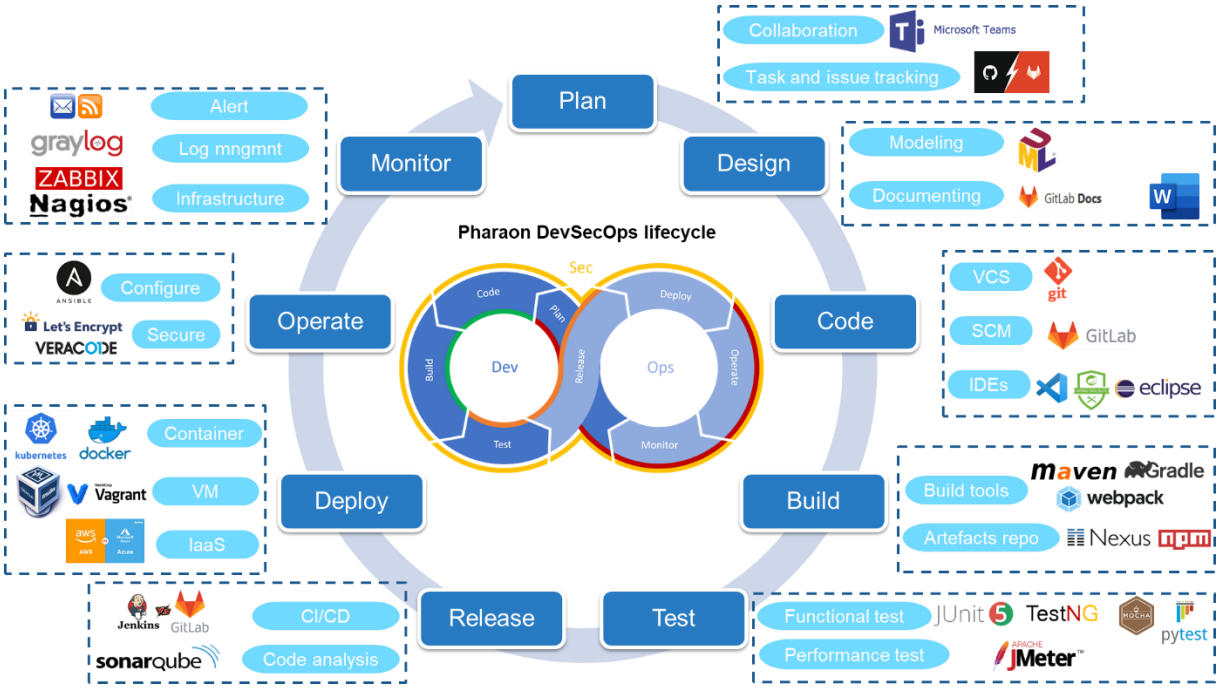

Figure 2.6 Selection of tools mapped to Pharaon DevSecOps lifecycle phases

Since cybersecurity is one of the Pharaon priorities special attention was given to the cybersecurity tools that will be further analyzed for the use in Pharaon in coordination with T3.4 focusing on cybersecurity are given in Appendix B: Cybersecurity tools.

3 Pharaon developers

When considering interaction with Pharaon technical environment and services we can distinguish Pharaon internal developers (relating to the technical user roles coming from different Pharaon partners) and Pharaon external developers (relating to the technical user roles coming from other organizations not participating in the Pharaon consortium and onboarded by WP6 open calls to bring additional value to Pharaon pilots).

Under the umbrella term "developer" we mean all technical roles serving all phases of the DevSecOps lifecycle from planning and coding, through integration, to deployment and monitoring. Furthermore, we do not differentiate among *junior developer, senior developer, non-technical developer, service developer, application developer, software engineer, ML engineer, researcher.*

3.1 Pharaon internal developer

In order to secure the Pharaon developers onboarding following process (shown in the Figure 3.1) shall be applied.

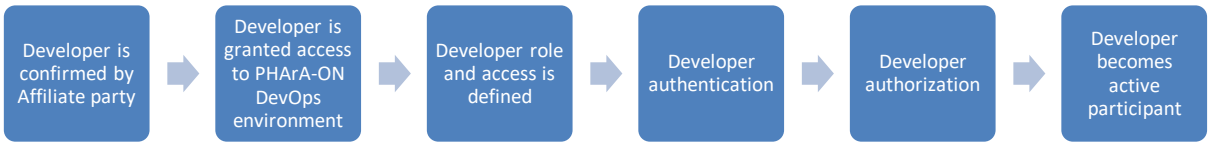

Figure 3.1 Securing Pharaon developer onboarding process [from D10.4]

3.2 Pharaon external developer

Pharaon external developers onboarding will follow the process as shown in the following figure (Figure 3.2).

Figure 3.2 Pharaon external developer onboarding process

© 2020 PHArA-ON

4 Pharaon Technical One-Stop-Shop

Pharaon Technical One-Stop-Shop is considered as a main entry point through which developers can access all the documentation, guidelines, different repositories, and most important technical information needed to work on the Pharaon technical ecosystem. The central part of Technical One-Stop-Shop is Developers' Handbook that contains instructions, guidelines, recommended open-source tools and best practices needed by developers and it is considered as the main explicit output of task T4.1 Developer guidelines & templates, described in this deliverable report.

In addition, Technical One-Stop-Shop also contains resources that are outputs of other technical tasks on Pharaon, such as code and code repositories being produced by WP3 Secure interoperability solution and tools being envisioned by other WP4 tasks. In this way the Technical One-Stop-Shop is considered as a placeholder for various technical (development) tasks outputs on Pharaon with Developers' Handbook for providing technical guidelines and best practices.

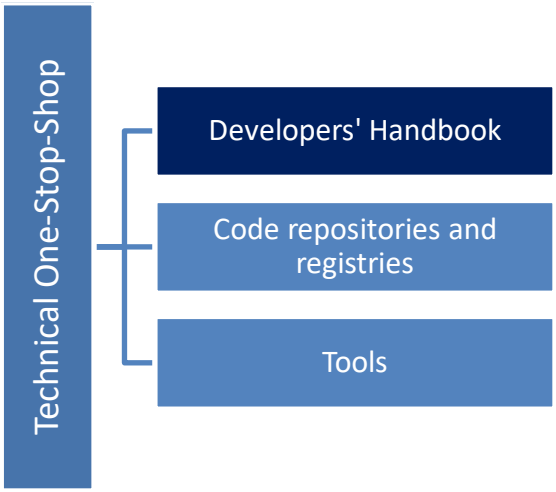

Figure 4.1 Main components of Pharaon Technical One-Stop-Shop

4.1 Infrastructure

Initial version of the Pharaon Technical One-Stop-Shop for developers is set up on Gitlab.com as a private group accessible at https://gitlab.com/pharaongroup. The group contains access to various repositories with resources for developers (both internal and external recruited through the open calls in WP6).

The free version of the Gitlab is used and it is hosted by Gitlab on their servers as SaaS. The Gitlab is used by 100 000+ organizations around the globe[12] as a web-based DevOps lifecycle tool that provides a Git-repository manager but also provides wiki, issue-tracking, continuous integration, deployment pipeline features and many more tools that are part of DevSecOps cycle. Gitlab features mapped to DevOps phases can be found on the following link: https://about.gitlab.com/features/ [accessed 15 Feb 2021].

[12] https://about.gitlab.com/

4.2 Content organization

Pharaon private Gitlab group (https://gitlab.com/pharaongroup) is administrated by Pharaon WP4 leaders to serve the needs of the Pharaon technical tasks and developers.

Initial organization of the content is meant to evolve following the needs of the Pharaon work. The snapshot of the content taken in March 2021, presented with Figure 4.2, shows the main Pharaon group with subgroups and projects following the principle of granting the minimal access rights.

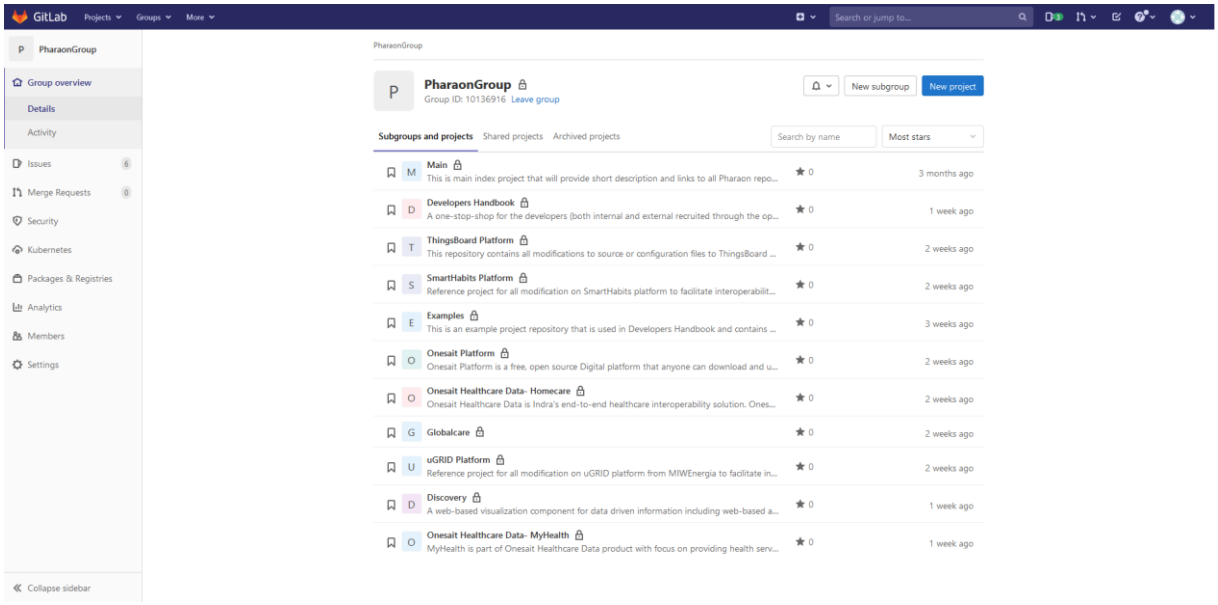

Figure 4.2 Pharaon main group with list of repositories

There are three main parts of Technical One-Stop-Shop for developers (as presented in Figure 4.1):

1. Developers' Handbook
2. Code repositories and registries
3. Tools

The Developers' Handbook is the main output of this task and described in detail in separate chapter (Chapter 5). It provides all relevant information, instructions, guidelines, recommended open-source tools and best practices needed by internal and external developers.

Code repositories and registries are primarily used by developers on WP3 and provide output of development work on adaptations of the Pharaon platforms within that work package. This WP3 output includes technological implementation extension and gateways (T3.1), adaptations of the platforms to implement the services and interfaces (T3.2), evolved semantic components oriented to seamless interoperability (T3.3), secure intra-platform (T3.5) and inter-platform interoperability components (T3.6). Every developer and task leader can create their own repository; however, initial repositories structure is proposed within this deliverable. Following figure (Figure 4.3) shows the main index with different Pharaon repositories to serve Pharaon technical work packages.

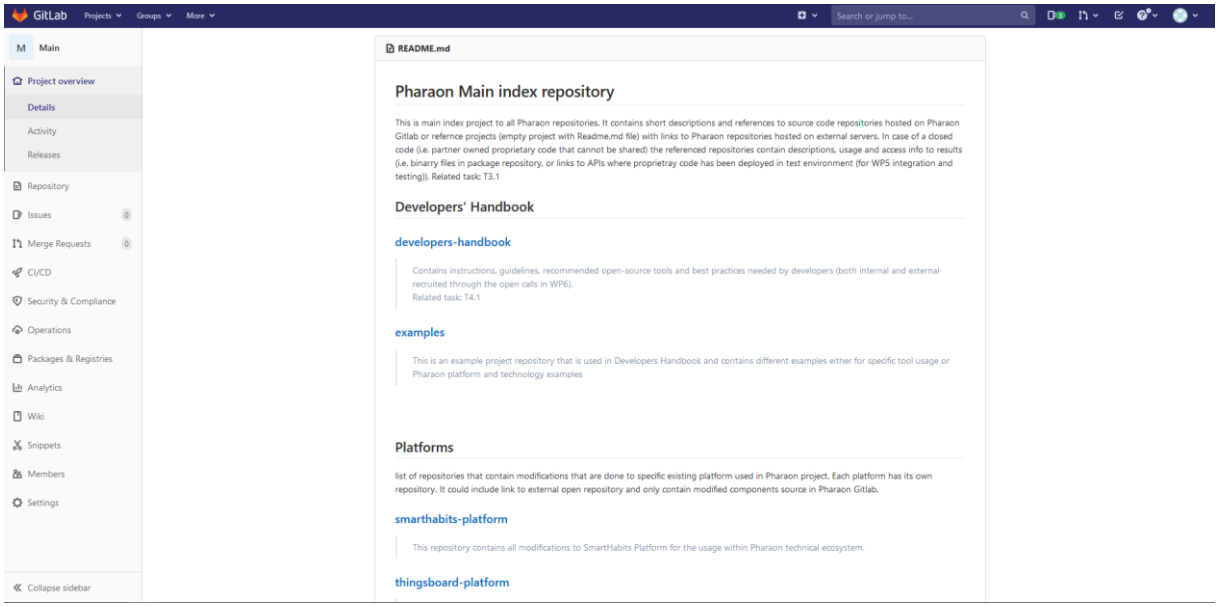

Figure 4.3 Pharaon main index with different repositories

The main index of Pharaon repositories (https://gitlab.com/pharaongroup/main) contains short descriptions and links to source code repositories hosted on Pharaon Gitlab and references with links to Pharaon repositories hosted on external servers. In case of a closed code (i.e. partner owned proprietary code that cannot be shared) this index file contains reference repositories with descriptions, usage and access info to APIs where the proprietary component has been deployed (i.e. in test environment for WP5 integration). Currently, the code repositories accessible from main index repository are organized as follows:

- *Developers' Handbook*
  - **developers-handbook:** Contains instructions, guidelines, recommended open-source tools and best practices needed by developers (both internal and external recruited through the open calls in WP6). Related task: T4.1
  - **examples:** This is an example project repository that is used in Developers Handbook and contains different examples either for specific tool usage or Pharaon platform and technology examples
- *Platforms*: list of repositories that contain modifications that are done to specific existing platform used in Pharaon project. Each platform has its own repository. It could include links to external open repositories and only contain modified components source in Pharaon Gitlab.
  - **smarthabits-platform**: This repository contains all modifications to the SmartHabits Platform for the usage within the Pharaon technical ecosystem.
  - **thingsboard-platform**: This repository contains all modifications to the ThingsBoard source or configuration files for the usage within the Pharaon technical ecosystem.
  - **onesait-platform**: This repository contains all modifications to the Onesait platform source for the usage within the Pharaon technical ecosystem.
  - **onesait-healthcare-data**: Onesait Healthcare Data is Indra's end-to-end healthcare interoperability solution. Onesait Healthcare Data is based on FHIR (Fast Healthcare

Interoperability Resources), the latest interoperability standard developed and promoted by HL7.
    - **globalcare**: Reference project for all modifications on Globalcare's Platform to facilitate interoperability with the Pharaon ecosystem.
    - **ugrid-platform**: Reference project for all modification on the uGRID platform from MIWEnergia to facilitate interoperability with the Pharaon ecosystem.
    - **iotool-platform**: Project for modifications on the IoTool Platform to support interoperability with the Pharaon ecosystem and additional modules needed in the Pharaon ecosystem.
    - **Iochat-platform**: Project for modifications on the IoChat Platform to support interoperability with the Pharaon ecosystem and additional modules needed in the Pharaon ecosystem.
- *Other technologies*: all repositories that contain modifications to other existing technologies used in Pharaon that are not considered platforms. For example, applications (frontends), devices or ML components used for achieving user/device interoperability. Related task: T3.2
    - **onesait-healthcare-data-myhealth**: MyHealth is part of Onesait Healthcare Data product with focus on providing health services to patients and their caregivers. It facilitates access to patients to their health data and the exchange of information with the health system.
    - **discovery**: A web-based visualization component for data driven information including web-based analytics.
- *Pharaon interoperability*: all components that ensure inter platform interoperability but are not specific to any input/existing platform or technology. Also includes semantic components oriented to seamless interoperability of applications within the system (T3.3). Related tasks: T3.1, T3.3, T3.5
- *Pharaon security*: all components that are not specific to any platform and are used to implement privacy and security aspects of Pharaon such as identify and access management, consent management, and data security. Related task: T3.4
- *Tools:* repositories for service orchestration, composition and customization support tools. Related tasks: T4.2, T4.3

The existing tools for developers are also considered as a resource to help developers in their work but are not direct outputs of this task. This task provides recommendations, instructions, and best practices on their usage, but actual deployment and maintenance is beyond of scope of this task. The survey of these tools for development has been described in Chapter 2.

5 Pharaon Developers' Handbook

The Developers' Handbook contains instructions, guidelines, recommended open-source tools and best practices needed by developers and it is considered as main output of task T4.1 Developer guidelines & templates. It is realized using Gitlab Wiki pages and available through Pharaon Gitlab at:

https://gitlab.com/pharaongroup/developers-handbook

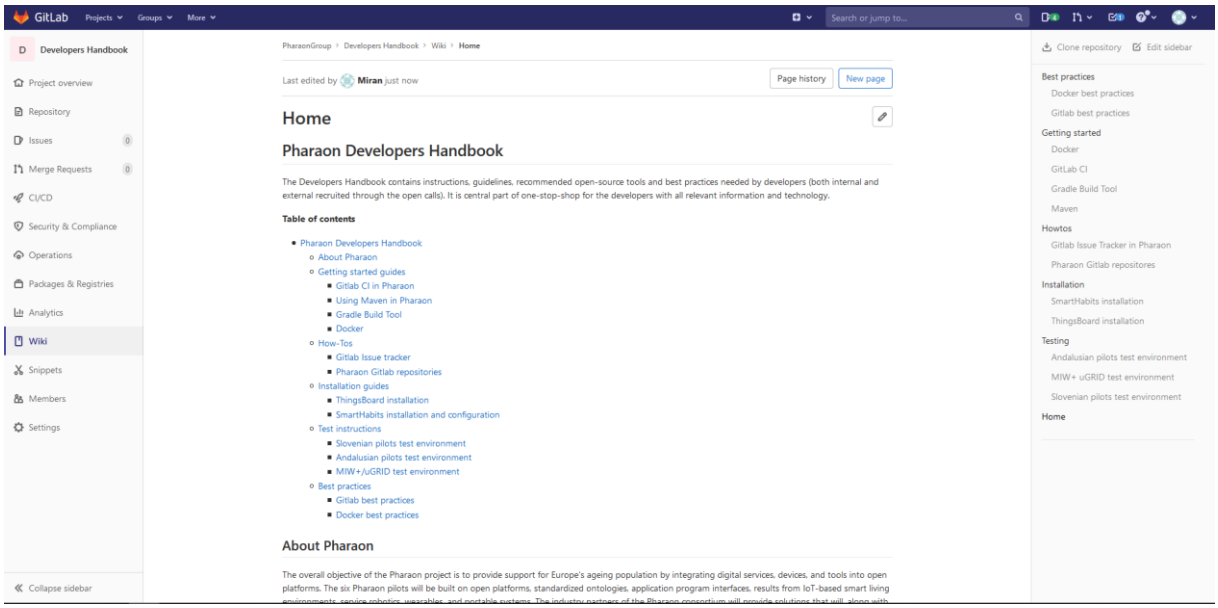

Figure 5.1 Pharaon Developers' Handbook home page (March 2021)

5.1 Rationality for using Gitlab Wiki as tool for Developers Handbook

For managing the creation and modification of digital content on a website usually content management systems (CMS) are used. CMS is typically a software or a set of related programs that helps building a website without the need for specialized technical knowledge. Joomla!, Drupal, SPIP and Wordpress are among the most well-known CMS tools. Content in a CMS is stored in a database and a CMS workflow system allows editorial teams to work on the website content, which is then validated by a publication manager. CMS focuses on standardized publishing information and usually has a limited group of editors. It is useful for relatively static content where emphasis is on style and presentation.

Besides CMS, wikis are also often used as a tool for documentation and are especially convenient for technical content that is to be provided in Pharaon Developers' Handbook. A wiki is classically presented as a set of alterable web pages by all users who have been granted permission to do so. It allows the collaborative creation of content (including text, images, videos), as well as the creation of

links between different sets of content. Collaborative needs are becoming more prominent in the modern-day working environment. [13]

The advantages of using wiki are that they are much more focused on collaboratively improving each topic (i.e. adding hyperlinks to other topics and websites counts as improving the topic) and are generally much more open to every developer on the project. Another advantage is that wikis focus on content and it is easier to find and update information. Wikis that are available together with code repositories are very convenient since they enable separation of concerns by not keeping documentation in the same repository as code, but still allowing access from the same place. For all these reasons Gitlab Wiki was chosen as a tool for creating and managing content for Pharaon Developers' Handbook.

Gitlab Wiki is a separate system for documentation built into GitLab. Gitlab Wiki pages can be created through the web interface or locally using Git since every Wiki is a separate Git repository.

5.2 Content organization

Pharaon Developers Handbook as a part of the One-Stop-Shop act as a placeholder for all relevant technical information for getting-started, understanding most important Pharaon technical innerworkings, installation and configuration guides and integration possibilities. It requires authentication and authorization so only the approved developers can access it from https://gitlab.com/pharaongroup/developers-handbook/-/wikis/home

Pharaon Developers' Handbook currently contains five different categories of documents (topics):

- Getting started guidelines
- How-tos
- Best practices
- Installation guides
- Usage instructions

Figure 5.2 Pharaon Developers' Handbook content organization

[13] https://xwiki.com/en/Alternatives/xwiki-vs-cms

5.2.1 Getting started guidelines

In general guidelines help guide developers through a certain process or task. They provide general recommendations of how to perform a task or advice for how to proceed in a situation. Guidelines usually provide a general overview and may be used in situations where no specific policy or standard applies. The guidelines are not mandatory and are typically not enforced, but instead attempt to streamline a process based around a sound practice. [14]

Getting started guidelines are considered as hands-on introduction to different Pharaon technologies and tools for developers. They include step-by-step guides that cover key tasks and operations and common problems and typically involve a guide through specific examples.

Figure 5.3 shows example of getting started guides created in Pharaon Developers' Handbook. It shows how to get started with using GitLab CI/CD in the Pharaon project and it is intended for developers on WP3, WP4 and WP5.

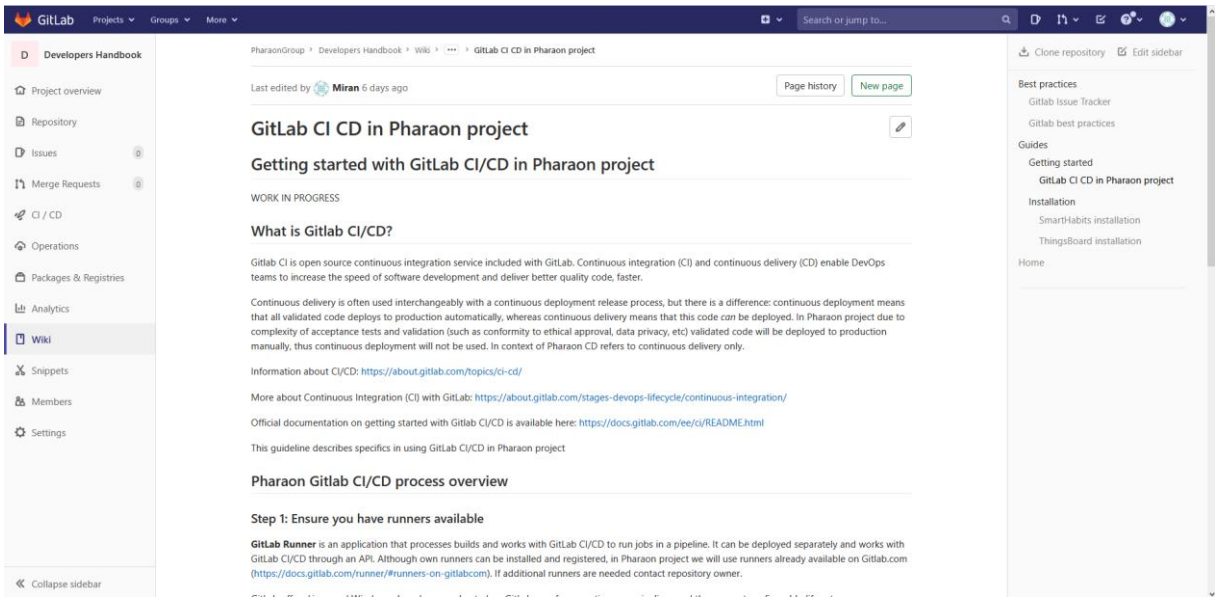

Figure 5.3 Example of getting started guideline - Pharaon Gitlab CI/CD

5.2.2 How-tos

How-tos are very similar to getting started guides but are related to specific Pharaon processes, such as using Issue Tracker or Gitlab repositories. Example on Figure 5.4 shows how to get started in using Issue Tracker on Pharaon and it is intended for developers on W3, WP4, WP5 and validation partners on WP7.

[14] https://www.powerdms.com/blog/guidelines-vs-policies/

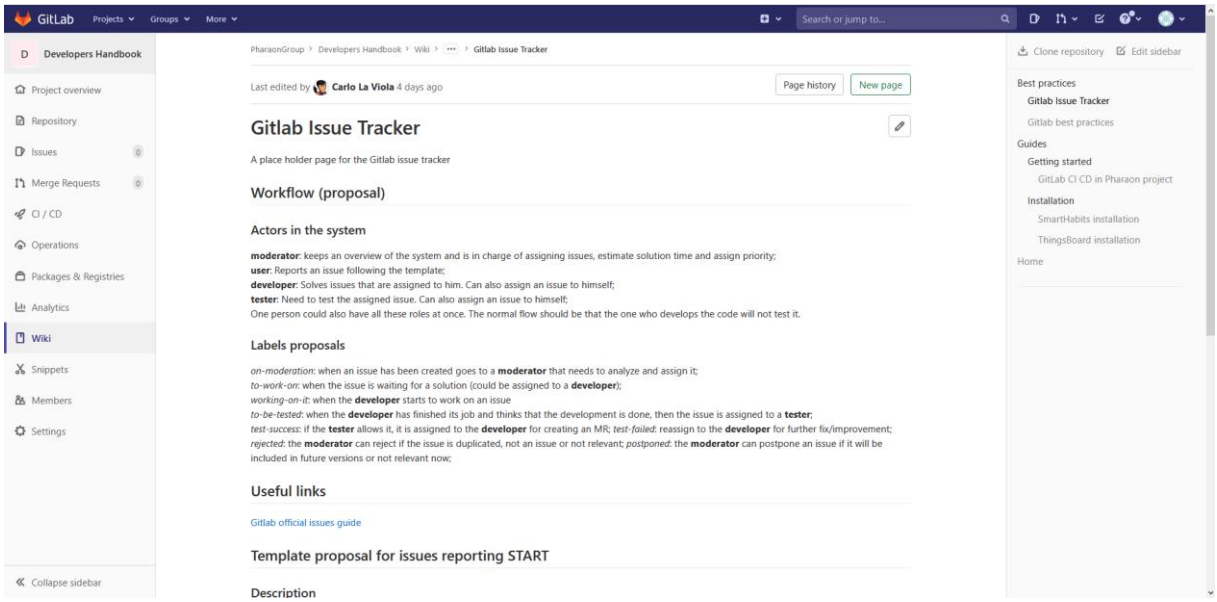

Figure 5.4 Example of how-to guideline - Pharaon Gitlab Issue Tracker

5.2.3 Best practices

Best practice can be described as proven technique that has been shown to produce positive results, compared to others. The word 'best' is a baseline, as even better methods for practices are developed.[15] Just as guidelines, best practices are not mandatory and are not enforceable however they usually do not include specific examples.

Figure 5.5 illustrates a best practice guide on how to use Git and Gitlab on the Pharaon project.

[15] https://www.josieahlquist.com/2013/09/30/policyguidebestpractice/

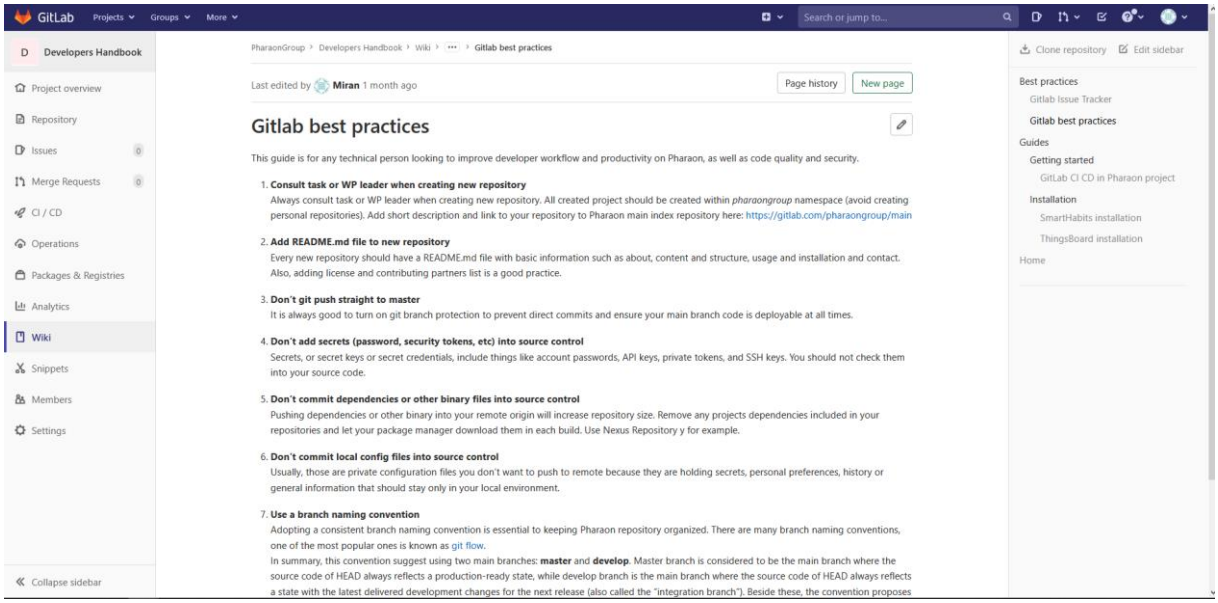

Figure 5.5 The example of best practices guide on Pharaon

5.2.4 Installation guides

An installation manual or installation guide is a technical communication document intended to instruct people how to install a particular product. An installation manual is usually written by a technical writer or other technical staff. Installation is the act of putting something in place so that it is ready for use.

The example of installation and configuration guide is presented on Figure 5.6.

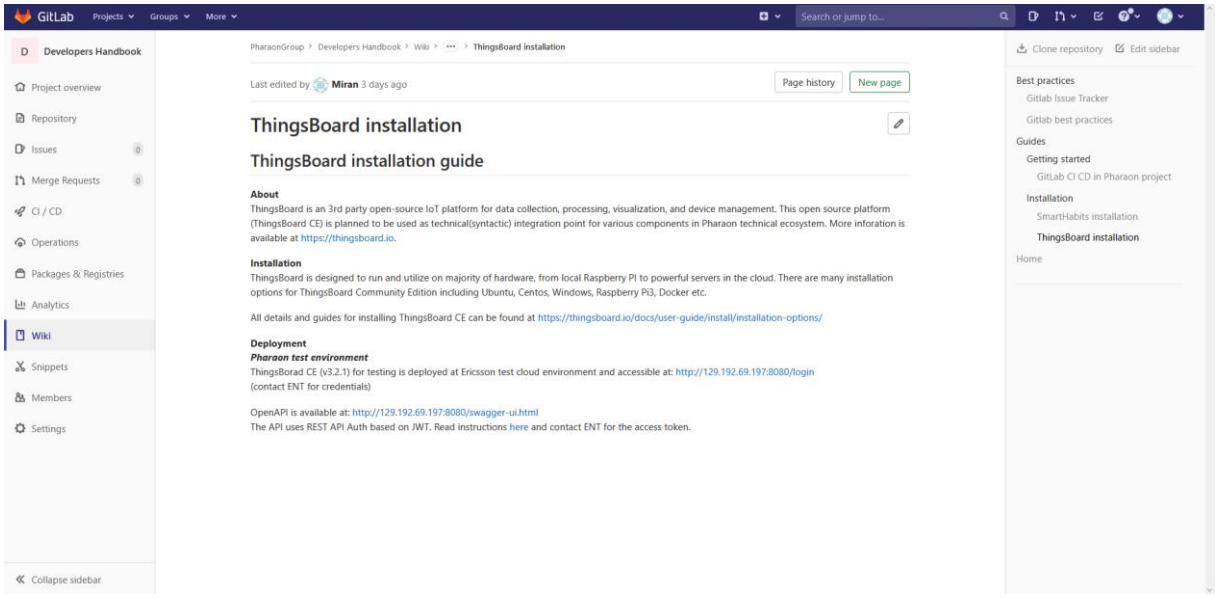

Figure 5.6 Example of installation and configuration guide

5.2.5 Usage instructions

Usage instructions contain documents which describe how to deploy certain Pharaon technology or platform to specific environment (i.e. to test environment on WP5 or production/pilot environment on WP7). It contains a set of directions provided by the developers of a Pharaon technical ecosystem and includes access information, test and validation instructions and usage guide. It could also contain brief explanations and clarification on key Pharaon technical topics.

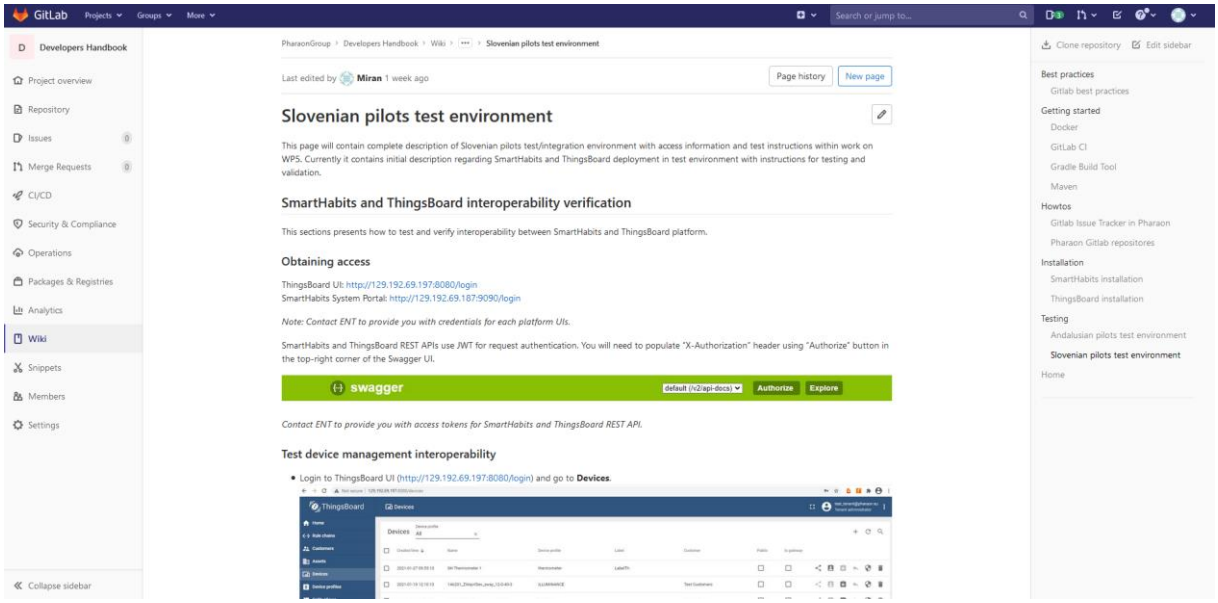

Figure 5.7 Example of test environment usage instructions

5.3 Guidelines reference list

Table 5.1 lists all guidelines that are currently available in Developers' Handbook. The status is referring to the current situation. Based on developers feedback it is expected that all guidelines will be updated in the next version of this deliverable, including the ones with current status 'Completed'.

In addition, the latest version of Developers' Handbook Gitlab project has been released using Gitlab Releases[16] and is available at (Figure 5.8):

https://gitlab.com/pharaongroup/developers-handbook/-/releases/1.0.0

Since every Gitlab Wiki is a separate Git repository, the latest version of Developers' Handbook Wiki pages has been tagged with label 1.0.0. and can be accesses by cloning following repository:

git@gitlab.com:pharaongroup/developers-handbook.wiki.git

Table 5.1 Pharaon guidelines reference list

Topic name	Short description	Type	Contrib. partners	Status	Link
Home page	Main page of Developers Handbook with brief introduction and references to all guidelines.	About	ENT	Completed	https://gitlab.com/pharaongroup/developers-handbook/-/wikis/home
Gitlab best practices	This guide is for any technical person looking to improve developer workflow and productivity on Pharaon, as well as code quality and security.	Best practices	ENT	Completed	https://gitlab.com/pharaongroup/developers-handbook/-/wikis/Best-practices/Gitlab-best-practices
GitLab CI/CD in Pharaon project	Getting started guideline about using GitLab CI/CD in Pharaon project.	How-to	ENT	In progress	https://gitlab.com/pharaongroup/developers-handbook/-/wikis/Getting-started/GitLab-CI
Pharaon Gitlab Issue Tracker	Getting started guideline about using GitLab Issue Tracker in Pharaon project.	Getting started	UNIFI	Completed	https://gitlab.com/pharaongroup/developers-handbook/-/wikis/Best-practices/Gitlab-Issue-Tracker
SmartHabits installation	Installation and configuration of 3rd party software components that are prerequisites for	Installation instr.	ENT	Completed	https://gitlab.com/pharaongroup/developers-handbook/-/wikis/Installation/SmartHabits-installation

[16] https://docs.gitlab.com/ee/user/project/releases/

	SmartHabits deployment				
ThingsBoard installation	Reference for ThingsBoard installation.	Installation instr.	ENT	Completed	https://gitlab.com/pharaongroup/developers-handbook/-/wikis/Installation/ThingsBoard-installation
Using Maven in Pharaon	Installation and usage of Maven in Pharaon.	Getting started	UNIFI	Completed	https://gitlab.com/pharaongroup/developers-handbook/-/wikis/Getting-started/Maven
Pharaon Gitlab repositories	Instructions on how to use Pharaon Gitlab repositories to share results of implementation work	How-to	ENT, INDRA	Completed	https://gitlab.com/pharaongroup/developers-handbook/-/wikis/Howtos/Pharaon-Gitlab-repositores
Gradle Build Tool	Getting started with Gradle build tool.	Getting started	ENT	in progress	https://gitlab.com/pharaongroup/developers-handbook/-/wikis/Getting-started/Gradle-Build-Tool
Getting started with Docker in Pharaon	Getting started with Docker in Pharaon.	Getting started	ENT	In progress	https://gitlab.com/pharaongroup/developers-handbook/-/wikis/Getting-started/Docker
Slovenian pilots test environment	Description of Slovenian pilot test/integration environment with access information and test instructions.	Usage instr.	ENT	Completed	https://gitlab.com/pharaongroup/developers-handbook/-/wikis/Testing/Slovenian-pilots-test-environment
Andalusian pilots test environment	Description of Andalusian pilot test/integration environment with access information and test instructions.	Usage instr.	INDRA	Completed	https://gitlab.com/pharaongroup/developers-handbook/-/wikis/Testing/Andalusian-pilots-test-environment
Docker best practices	This guide is for any technical person looking to improve security and overall creation of Dockerfiles.	Best practices	ENT	In progress	https://gitlab.com/pharaongroup/developers-handbook/-/wikis/Best-practices/Docker-best-practices
MIW+/uGRID test environment	Description of how to use MIW+/uGRID technology	Usage instr.	MIWENERGIA	Planned	https://gitlab.com/pharaongroup/developers-handbook/-/wikis/Testing/MIW--uGRID-test-environment

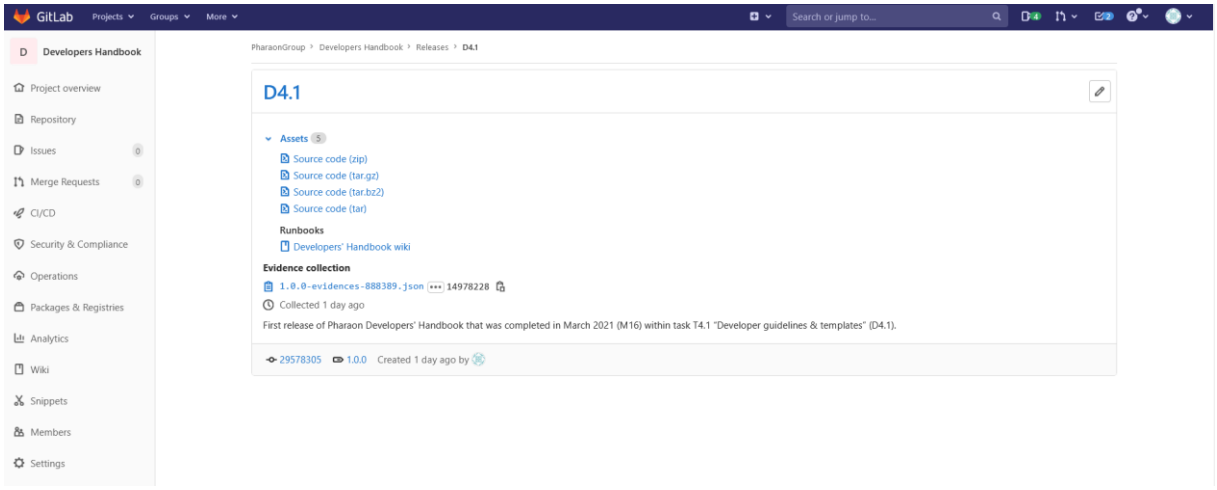

Figure 5.8 First release of Developers' Handbook Gitlab project available through Gitlab *Releases*

6 Conclusions and next steps

Considering many different partners and organizations it is of vital importance to give the right information, guidelines and tools related to all technical activities to the right people. Taking into consideration all the common and best practices in the field the activities that were initiated aim to streamline and make the technical work within Pharaon as effective and efficient as possible. Not only for Pharaon internal developers but also for the external ones that are to be onboarded in later stages within Pharaon (WP6) open calls. The future activities will be based on the extensive analysis of the best value for the technical development and will consider limited timeframe, finite effort pool in terms of personnel engagement and will aim to provide the best possible added value for the main stakeholders, in our case being the technical personnel within and outside Pharaon.

© 2020 PHArA-ON

Appendix A: Considered SDLC tools

Table A.0.1 SDLC tools - Code phase

Tool category	Tool name	Short Description	License/ Price	Link	Status
SCM (source control mngmnt)	Git	Git is one of the most popular DevOps tools, widely used across the software industry. It's a distributed SCM (source code management) tool, loved by remote teams and open source contributors	GPLv2 & LGPL v2.1	https://git-scm.com/	Selected
	Subversion	Open-source revision control and software versioning system that provides interactive conflict resolution, merge tracking, file locking, and more.	Apache 2.0	https://subversion.apache.org/	Considered
	CVS	CVS operates as a front end to Revision Control System (RCS), an older version control system that manages individual files but not whole projects. It expands upon RCS by adding support for repository-level change tracking, and a client-server model	GNU	https://savannah.nongnu.org/projects/cvs	Dismissed
	Vesta	Vesta is a portable SCM system targeted at supporting development of software systems of almost any size, from fairly small to very large	LGPL	http://www.vestasys.org/	Dismissed
	Mercurial	Mercurial is a free, distributed source control management tool. It efficiently handles projects of any size and offers an easy and intuitive interface.	GNU GPL v2+	https://www.mercurial-scm.org/	Considered
Git hosting service	Bitbucket	Bitbucket has been around for many years. In some ways, it could serve as a looking glass into the future of GitHub. Bitbucket was acquired by a larger corporation (Atlassian) eight years ago and has already been through some of that change-over process		https://bitbucket.org/	Considered
	GitHub	It's the largest community website for software development, and it still has some of the best tools for issue tracking, code review, continuous integration, and general code management.		https://github.com/	Considered
	GitLab	GitLab is probably the leading contender when it comes to alternative code platforms. It's fully open source. You can host your code right on GitLab's site much like you would on GitHub, but you can also choose to self-host a GitLab instance of your own on your own server and have full control over who has access to everything there and how things are managed.		http://gitlab.com/	Selected
	Microsoft Azure DevOps	Provides version control, reporting, requirements management, project management, management capabilities. It covers the entire application lifecycle and enables DevOps capabilities.		https://azure.microsoft.com/en-us/services/devops/	Considered
	Amazon AWS CodeCommit	A Git based version control service hosted by Amazon Web Services that can be used to privately store and manage assets (such		https://aws.amazon.com/codecommit/	Considered

		as documents, source code, and binary files) in the cloud.			
	Google Cloud Source Repositories	Fully featured, private Git repositories hosted on Google Cloud.		https://cloud.google.com/source-repositories	Considered
	Beanstalk	Provides a complete workflow to write, review & deploy code.		http://beanstalkapp.com/	Considered
	Codebase	Code hosting and project management tool for developers		http://www.codebasehq.com/	Considered
	Fog Creek Kiln	Source control management system with tightly integrated code review		https://www.fogcreek.com/kiln/	Considered
	Launchpad	Softwre collaboration platform that also provides code hosting using Bazaar and Git	GNU	https://launchpad.net/	Considered
	Planio	Online project management platform that includes Git based repositories.		https://plan.io/	Considered
	Perforce	Perforce is an enterprise version management system in which users connect to a shared file repository.		http://www.perforce.com/	Considered
	RhodeCode	Enterprise code management for Hg, Git and SVN		https://rhodecode.com/	Considered
	SourceForge	One of the most popular open source code repository site.		https://sourceforge.net/	Dismissed
	Assembla	Multirepository platform offering secure Git, SVN and Perforce		https://www.assembla.com/home?affiliate=fournova	Considered
	Gogs	A painless self-hosted Git service. Gogs is 100% open source and free of charge.	MIT license	https://gogs.io/	Considered
Desktop IDE	Eclipse	IDE for developing applications using the Java programming language and other programming languages such as C/C++, Python, PERL, Ruby etc	Eclipse Public License	https://www.eclipse.org/ide/	Selected
Cloud IDE	GitPod	Gitpod does to Dev Environments what Docker did to Servers	Free/ Comm.	https://www.gitpod.io/	Considered
	Theia	Eclipse Theia is an extensible platform to develop multi-language Cloud & Desktop IDEs with state-of-the-art web technologies.		https://theia-ide.org/	Considered
Source code editors	Visual Studio Code	Visual Studio Code is a free source-code editor made by Microsoft for Windows, Linux and macOS.[7] Features include support for debugging, syntax highlighting, intelligent code completion, snippets, code refactoring, and embedded Git	MIT	https://code.visualstudio.com/	Selected
	Atom	Atom is a free and open source desktop text editor	Open source	https://atom.io/	Considered
	Sublime Text	Sublime Text is a sophisticated text editor for code, markup and prose. It natively supports many programming languages and markup languages, and functions can be added by users with plugins, typically community-built and maintained under free-software licenses.		https://www.sublimetext.com/	Considered
	SandDance for VSCode	Visually explore, understand, and present your data.	free	https://marketplace.visualstudio.com/items?itemName=msrvida.vsc	Considered

				ode-sanddance	
	Data Preview for VSCode	Data Preview extension for importing, viewing, slicing, dicing, charting & exporting large .json array .arrow .avro data files, .config .env .properties .ini .yml configurations files, .csv/.tsv & .xlsx/.xlsb Excel files and .md markdown tables with Perspective - streaming data analytics WebAssembly library.	Apache 2.0	https://marketplace.visualstudio.com/items?itemName=RandomFractalsInc.vscode-data-preview	Considered
	GitLab Workflow VSCode integration	Adding a new GitLab sidebar where you can find issues and merge requests created by you or assigned to you. It also extends VS Code command palette and status bar to provide more information about your project.		https://marketplace.visualstudio.com/items?itemName=GitLab.gitlab-workflow	Considered
	GitLab-CI Templates VSCode	Create .gitlab-ci.yml file from GitLab-CI templates		https://marketplace.visualstudio.com/items?itemName=jgsqware.gitlab-ci-templates	Considered
	OpenAPI (Swagger) Editor VSCOde	SwaggerUI and ReDoc preview,IntelliSense, linting, schema enforcement, code navigation, definition links, snippets, static security analysis, and more!		https://marketplace.visualstudio.com/items?itemName=42Crunch.vscode-openapi	Considered

Table A.0.2 SDLC tools - Build, package phase

Tool category	Tool name	Short Description	License /Price	Related lang.	Link	Status
Build automation tool	Maven	Apache Maven is a software project management and comprehension tool. Based on the concept of a project object model (POM), Maven can manage a project's build, reporting and documentation from a central piece of information.	Apache 2.0	Java	https://maven.apache.org/gradle	Selected
	Ant	Apache Ant is a Java library and command-line tool whose mission is to drive processes described in build files as targets and extension points dependent upon each other.	Apache 2.0	Java	https://ant.apache.org/	Considered
	Gradle	Gradle is automated build tool which allows you to write your code in Java, C++, Python, or other languages. Gradle is also supported by popular IDEs such as Netbeans, Eclipse, and IntelliJ IDEA and it as the official build tool for Android Studio.	Apache 2.0	Java	https://gradle.org/	Selected
	Bazel	Bazel is an open-source build and test tool similar to Make, Maven,	Apache 2.0	Java	https://bazel.build/	Considered

		and Gradle. It uses a human-readable, high-level build language. Bazel supports projects in multiple languages and builds outputs for multiple platforms. Bazel supports large codebases across multiple repositories, and large numbers of users.				
	Make	Make is a build automation tool that automatically builds executable programs and libraries	GNU	N/A	https://www.gnu.org/software/make/	Considered
	Grunt	JavaScript task runner	MIT	JavaScript	https://gruntjs.com/	Considered
	Gulp	A toolkit to automate & enhance workflow	MIT	JavaScript	https://gulpjs.com/	Considered
	Buildr	Apache Buildr is a build system for Java-based applications, including support for Scala, Groovy and a growing number of JVM languages and tools.	Apache	Ruby	https://buildr.apache.org/	Considered
	Rake	Rake is a Make-like program implemented in Ruby	MIT	Ruby	https://github.com/ruby/rake	Considered
	A-A-P	A-A-P makes it easy to locate, download, build and install software.	GNU	Python	http://www.a-a-p.org/tools_build.html	Considered
	Scons	Open Source software construction tool.	MIT	Python	https://scons.org/	Considered
	BitBake	BitBake is a make-like build tool with the special focus of distributions and packages for embedded Linux cross compilation	GPLv2	Python		Considered
	Cake	free and open source cross-platform build automation system with a C# DSL	MIT	C#	https://cakebuild.net/	Considered
	ASDF	Manage multiple runtime versions with a single CLI tool	MIT	LISP	https://asdf-vm.com/	Considered
	Cabal	A system for building and packaging Haskell libraries and programs	BSD	Haskell	https://www.haskell.org/cabal/	Considered
Automate dependency updates	Renovate	handles the automation of package updates. Has both GitLab and GitHub app	AGPL3.0	Multi-platform and multi-language	https://github.com/renovatebot/renovate	Considered
	Dependabot	Dependabot creates pull requests to keep your dependencies secure and up-to-date.	Zero Prosperity Public License	Multi-platform and multi-language	https://dependabot.com/	Considered
Package (Binaries) repository	Sonatype Nexus OSS	Repository manager		for npm, Docker, Bower, Maven, Go, PyPi	https://www.sonatype.com/nexus/repository-oss	Considered
General purpose repostitory	Zenodo	Zenodo is a general-purpose open-access repository developed under the European OpenAIRE program and operated by CERN. It allows researchers to deposit research papers, data sets, research			https://zenodo.org/	Selected

		software, reports, and any other research related digital artifacts.				

Table A.0.3 SDLC tools - Test phase

Tool category	Tool name	Short Description	License /Price	Related language	Link	Status
Code test framework	Junit	JUnit is a simple framework to write repeatable tests. *JUnit 5* is the next generation of JUnit. The goal is to create an up-to-date foundation for developer-side testing on the JVM. This includes focusing on Java 8 and above, as well as enabling many different styles of testing.	Eclipse Public (EPL)	Java	https://junit.org/junit5/	Selected
	TestNG	testing framework which, inspired by JUnit and NUnit, but introducing many new innovative functionalities like dependency testing, grouping concept to make testing more powerful and easier to do	Apache 2.0	Java	https://testng.org	Considered
	EasyMock	EasyMock is an open-source testing framework for Java released under the Apache License. The framework allows the creation of test double objects for the purpose of Test-driven Development or Behavior Driven Development.	Apache	Java	https://easymock.org/	Considered
	Mockito	Tasty mocking framework for unit tests in Java	MIT	Java	https://site.mockito.org/	Considered
	PowerMock	PowerMock is a framework that extends other mock libraries such as EasyMock with more powerful capabilities.	Apache 2.0	Java	https://github.com/powermock/powermock	Considered
	Pytest	The pytest framework makes it easy to write small tests, yet scales to support complex functional testing for applications and libraries.	MIT	Python	https://docs.pytest.org/en/stable/index.html	Considered
	Hypothesis	Modern implementation of property-based testing	Mozilla	Python	https://hypothesis.works/	Considered
	Tox	Command line driven CI frontend and development task automation too	MIT	Python	https://pypi.org/project/tox/	Considered
	QuerySurge	QuerySurge is the smart data testing solution that is the first-of-its-kind full DevOps solution for continuous data testing.	Commercial	data testing		Dismissed
Code quality	Cobertura	Free Java tool that calculates the percentage of code accessed by tests.	GNU	Java	https://cobertura.github.io/cobertura/	Considered
	CodeCover	Open source glass testing tool.	Eclipse Public (EPL)	Java	http://codecover.org/	Considered

	Coverage.py	A tool for measuring code coverage of Python programs	Apache 2.0	Python	https://coverage.readthedocs.io/en/coverage-5.5/	Considered
	Emma	Open-source toolkit for measuring and reporting Java code coverage.	Common Public License	Java	http://emma.sourceforge.net/	Considered
	JaCoCo	Free code coverage library for Java	Eclipse Public License	Java	https://www.eclemma.org/jacoco/	Considered
	Hypothesis	Python library for creating unit tests which are simpler to write and more powerful when run	Mozilla	Python	https://pypi.org/project/hypothesis/	Considered
	Tox	A generic virtualenv management and test command line tool	MIT	Python	https://pypi.org/project/tox/	Considered
	Jasmine	QA testing tool for JavaScript	MIT	JavaScript		Considered
	Karma	A simple tool that allows executing JavaScript code in multiple real browsers.	MIT	JavaScript	https://github.com/karma-runner/karma	Considered
	Mocha	Feature-rich JavaScript test framework running on Node.js and in the browser,	MIT	JavaScript	https://mochajs.org/	Considered
	Jest	JavaScript Testing Framework with a focus on simplicity	MIT	JavaScript	https://jestjs.io/	Considered
Data generator	Faker	Faker is a Python package that generates fake data for you. Whether you need to bootstrap your database, create good-looking XML documents, fill-in your persistence to stress test it, or anonymize data taken from a production service, Faker is for you.	MIT	Python	https://github.com/joke2k/faker	Considered
Load test	Apache JMeter	load testing to analyze and measure the efficiency and performance of the services especially the services are web applications	Apache 2.0	Java	https://jmeter.apache.org/	Considered
	Gatling	Gatling is an open-source load testing tool for web applications, designed for DevOps and Continuous Integration. Gatling includes a web recorder and colorful reports.	Apache		https://gatling.io/open-source/	Considered
API testing	REST-Assured	REST-Assured is a fluent Java library you can use to test HTTP-based REST services. It's designed with testing in mind, and it integrates with any existing Java-based automation framework.			https://rest-assured.io/	Considered
	Postman	Postman is an easy-to-use REST client, and you can get started with it quickly by leveraging its Chrome plug-in. There are native versions for both Mac and Windows.			https://www.getpostman.com/	Considered
	SoapUI	An open-source web service testing application for Simple	EUPL		https://www.soapui.org/do	Considered

		Object Access Protocol and representational state transfers.			wnloads/soapui.html	
	Karate	Karate offers API testing, API testing doubles, and API performance testing all in one framework.	MIT	Java	https://github.com/intuit/karate	Considered
	Fiddler	Fiddler lets you monitor, manipulate, and reuse HTTP requests. . It does many things that allow you to debug website issues, and, with one of its many extensions, you can accomplish even more.			https://www.telerik.com/fiddler	Considered
	Citrus Framework	The Citrus Framework can help you automate integration tests for virtually any messaging protocol or data format.			https://citrusframework.org/	Considered
	Insomnia	Insomnia allows you to create HTTP requests, view response details, organize your tests, reuse values, generate code snippets			https://insomnia.rest/	Considered
	Dredd	Dredd is a language-agnostic command-line tool for validating API description document against backend implementation of the API. It is a response verifier that takes examples from the OpenAPI definition of the API, invokes the API, and then compares the received responses to what is declared in the OpenAPI specification.	MIT	Go, Node.js (JavaScript), Perl, PHP, Python, Ruby, Rust	https://dredd.org/en/latest/	Considered
	Taurus	Taurus is an automation-friendly framework for continuous testing. Because you can use it with JMeter, it can handle API testing.	Apache 2.0		https://gettaurus.org/	Considered

Table A.0.4 SDLC tools - Release phase

Tool category	Tool name	Short Description	License	Link	Status
CI/CD tools	Jenkins	An open source, Java-based CI/CD tool based on the MIT License, is the tool that popularized the DevOps movement and has become the de facto standard.	Creative Commons and MIT	https://github.com/jenkinsci/jenkins	Considered

	Bamboo	Bamboo is Atlassian's CI/CD server solution that has many similar features to Jenkins. Bamboo has many pre-built functionalities that you have to set up manually in Jenkins.	Free/Enterprise	https://www.atlassian.com/software/bamboo	Considered
	Travis CI	Hosted continuous integration service used to build and test software projects hosted on GitHub and Bitbucket.	MIT	https://travis-ci.org/	Considered
	TeamCity	Build management and continuous integration server from JetBrains	Free/Enterprise	https://www.jetbrains.com/teamcity/	Considered
	CruiseControl	Java-based framework for a continuous build process.	BSD	http://cruisecontrol.sourceforge.net/	Considered
	Buildbot	Software development continuous integration tool which automates the compile or test cycle required to validate changes to the project code base	GPL	https://buildbot.net/	Considered
	Apache Gump	Open source continuous integration system, which aims to build and test all the open source Java projects	Apache 2.0	https://gump.apache.org/	Considered
	Go CD	Free & open source ci/cd server	Open Source		Considered
	Gitlab CI/CD Community	GitLab is a web-based DevOps lifecycle tool that provides a Git-repository manager providing wiki, issue-tracking and continuous integration and deployment pipeline	Free		Selected
	Github Actions	Allows building continuous integration and continuous deployment pipelines for testing, releasing and deploying software	Free/Enterprise		Considered

Table A.0.5 SDLC tools - Operate phase

Tool category	Tool name	Short Description	License/Price	Link	Status
Secret Management	Vault	Secure, store and tightly control access to tokens, passwords, certificates, encryption keys for protecting secrets and other sensitive data using a UI, CLI, or HTTP API.	Open Source & Enterprise	https://www.vaultproject.io/	Considered
Service Discovery	etcd	A distributed, reliable key-value store for the most critical data of a distributed system		https://etcd.io/ https://github.com/etcd-io/etcd	Considered
	Consul	Consul is a tool for service discovery, monitoring, and configuration. It uses Serf to form dynamic clusters and a peer-to-peer data store, based on the Serf library. Consul is a highly distributed service discovery tool.		https://www.consul.io/ https://github.com/hashicorp/consul	Considered

Table A.0.6 SDLC tools - Deploy phase

Tool category	Tool name	Short Description	License /Price	Link	Status
Containers	Docker	Docker has been the number one container platform since its launch. Docker has made containerization popular in the tech world, mainly because it makes distributed development possible and automates the deployment of apps.	Apache 2.0	https://www.docker.com/	Selected
	Portainer Community Edition	Portainer simplifies container management in Docker, Swarm, Kubernetes, ACI and Edge environments. It's used by software engineers to speed up software deployments, troubleshoot problems and simplify migrations.		https://www.portainer.io/	Considered
	DockerHub	World's largest library and community for container images.		https://hub.docker.com	Selected
	Kubeflow	Machine learning toolkit for Kubernetes	Apache 2.0		Considered
	Kubernetes	open-source container-orchestration system for automating computer application deployment, scaling, and management	Apache 2.0	https://kubernetes.io	Considered
Configuration management tools	Ansible	"Ansible is a radically simple IT automation engine that automates cloud provisioning, configuration management, application deployment, intra-service orchestration, and many other IT needs." IT automation and orchestration engine.	GNU Public	www.ansible.com/home	Considered
	SaltStack	Python-based, open-source software for event-driven IT automation, remote task execution, and configuration management	Apache 2.0	https://www.saltstack.com/	Considered
	Chef	Configuration management tool written in Ruby and Erlang. It uses a pure-Ruby, domain-specific language (DSL) for writing system configuration "recipes".	Apache 2.0	https://www.chef.io/configuration-management	Considered
	Puppet	The puppet is a system management tool that helps in automating and centralizing the configuration management process. It also used for software deployment.	Apache or GPL	https://puppet.com/	Considered
Web application server	Tomcat	Open-source implementation of the Java Servlet, JavaServer Pages, Java Expression Language and WebSocket technologies	Apache 2.0	http://tomcat.apache.org/	Considered
	Jetty	Java web server and Java Servlet container from Eclipse	Apache 2.0, EPL	https://www.eclipse.org/jetty/	Considered
	WildFly	An application server written in Java and implements the Java Platform, Enterprise Edition (Java EE) specification.	LGPL	https://www.wildfly.org/	Considered
	GlassFish	Reference implementation of Jakarta EE and as such supports EJB, JPA, JSF, JMS, RMI, JSP, servlets, etc.	EPL or GPL+Classpath exc.	https://javaee.github.io/glassfish/	Considered

	Django	Python-based free and open-source web framework that follows the model-template-views (MTV) architectural pattern	3-clause BSD	https://www.djangoproject.com/	Considered
	Tornado	a scalable, non-blocking web server and web application framework written in Python.	Apache 2.0	https://www.tornadoweb.org/en/stable/	Considered
	Gunicorn	Python WSGI HTTP Server for UNIX	MIT	https://gunicorn.org/	Considered
	Python Paste	Set of utilities for web development in Python	MIT	https://pypi.org/project/Paste/	Considered
	Rails	Server-side web application framework written in Ruby under the MIT License. Rails is a model–view–controller (MVC) framework.	MIT	https://rubyonrails.org/	Considered
	Node.js	Open-source, cross-platform, back-end JavaScript runtime environment that runs on the V8 engine	MIT	https://github.com/nodejs/node	Considered

Table A.0.7 SDLC tools - Monitor phase

Tool category	Tool name	Short Description	License	Link	Status
Logging	ELK Stack	Elasticsearch, Logstash, and Kibana. Elasticsearch is a search and analytics engine. Logstash is a server-side data processing pipeline that ingests data from multiple sources simultaneously, transforms it, and then sends it to a "stash" like Elasticsearch. Kibana lets users visualize data with charts and graphs in Elasticsearch.	Open Source & Enterprise	https://www.elastic.co/what-is/elk-stack	Considered
	Fluentd	Cross platform open-source data collection software	Open Source	https://www.fluentd.org/	Considered
	Graylog	An open source log management platform.	Open Source & Enterprise	https://www.graylog.org/	Considered
	logz.io	A cloud observability platform providing Log Management built on ELK, Infrastructure Monitoring based on Prometheus, and an ELK-based Cloud	Enterprise	https://logz.io/	Considered
	Splunk	Captures, indexes and correlates real-time data in a searchable repository from which it can generate graphs, reports, alerts, dashboards and visualizations.	Enterprise	https://www.splunk.com/	Considered
	Syslog-ng	Free and open-source implementation of the syslog protocol for Unix and Unix-like systems.	Enterprise	https://www.syslog-ng.com/	Considered
IT infrastructure Monitoring	Prometheus	An open-source system monitoring and alerting toolkit. t records real-time metrics in a time series database (allowing for high dimensionality) built using a HTTP	Open Source	https://prometheus.io/	Considered

		pull model, with flexible queries and real-time alerting			
	Sensu	The Observability Pipeline that delivers monitoring as code on any cloud	Open Source & Enterprise	https://sensu.io/	Considered
	Riemann	Riemann aggregates events from servers and applications with a powerful stream processing language.	Open Source	https://riemann.io/	Considered
	Nagios	Free and open-source computer-software application that monitors systems, networks and infrastructure.	Free and Enterprise	https://www.nagios.org/	Considered
	Zabbix	Monitoring software tool for diverse IT components, including networks, servers, virtual machines and cloud services.	Open Source & Enterprise	https://www.zabbix.com/	Considered
	Data Dog	A monitoring service for cloud-scale applications, providing monitoring of servers, databases, tools, and services, through a SaaS-based data analytics platform.	Enterprise	https://www.datadoghq.com/	Considered
	New Relic	Collects all telemetry data in one place to deliver full-stack observability and power AI-driven insights	Enterprise	https://newrelic.com/	Considered
	App Dynamics	Application performance management and IT operations analytics software	Enterprise	https://www.appdynamics.com/	Considered
	Sumologic	provides log management and analytics services that leverage machine-generated big data to deliver real-time IT insights.	Enterprise	https://www.sumologic.com/	Considered
APM	Apache SkyWalking	Application performance monitor tool for distributed systems, especially designed for microservices, cloud native and container-based (Docker, Kubernetes, Mesos) architectures.	Open Source	https://skywalking.apache.org/	Considered
	Dynatrace	Kubernetes monitoring simplified	Enterprise	https://www.dynatrace.com/technologies/kubernetes-monitoring/	Considered

Table A.0.8 SDLC tools - Other

Tool category	Tool name	Short Description	License	Link	Selected
Anonymization	ARX	ARX is a comprehensive open source software for anonymizing sensitive personal data. It supports a wide variety of (1) privacy and risk models, (2) methods for transforming data and (3) methods for analyzing the usefulness of output data.	Apache 2.0	https://arx.deidentifier.org/	Considered
	OnFHIR	Enables sensitive patient data to be processed in compliance with GDPR	proprietary	https://onfhir.io/technology.html	Considered

Interoperability	HAPI FHIR	HAPI FHIR is a complete implementation of the HL7 FHIR standard for healthcare interoperability in Java	Apache 2.0	https://hapifhir.io/	Considered
	openmhealth.org	making patient-generated data accessible through an open data standard and community	Apache 2.0	https://www.openmhealth.org/	Considered
	Express Gateway	Microservices and Serverless API Gateway Built on Express.JS.	Apache 2.0	https://www.express-gateway.io/	Considered
	NodeRed	Node-RED is a flow-based development tool for visual programming developed originally by IBM for wiring together hardware devices, APIs and online services as part of the Internet of Things.	Apache 2.0	https://nodered.org/ https://github.com/node-red/node-red	Considered
Data exploration and visualization	Apache Superset	open-source data exploration and visualization platform designed to be visual, intuitive, and interactive. It enables users to analyze data using its SQL editor, and easily build charts and dashboards. Top level Apache project since 21.2.2021	Apache 2.0	https://superset.apache.org/	Considered

Appendix B: Cybersecurity tools

Table B.0.1 Open source cybsersecurity tools

Tool category	Tool name	Tool features	Web link
Identity management (secure authentication and authorization or users and devices)	OpenIAM Community Edition	• Single Sign On - SAML 2, OpenID Connect • Self-service portal - Profile Management, Forgot password • Automated provisioning to many applications (AD, Google, Office365) • User and Group Management • Flexible Authentication / Authorization	https://www.openiam.com/
	Keycloak	• of-the-box user authentication and federation • standard protocols • centralized management • password policies	https://www.keycloak.org/index.html
	Apache Syncope	• identity lifecycle management, • identity storage, • provisioning engines, • access management capabilities	https://syncope.apache.org/
	Gluu	• authorization server for web & API access management. • directory for identity data storage, • authentication middleware for inbound identities, • two-factor authentication, • directory integration	https://www.gluu.org/
Antivirus	Avast Free Antivirus	• virus and malware protection — including anti-rootkit and anti-spyware capabilities • access to threat detection networks, • blocking malicious URLs, and stopping auto-downloads • monitoring app activity to watch for any suspicious issues • machine-learning antivirus tool alongside Wifi and Browser Security	https://www.avast.com/en-gb/free-antivirus-download
	AVG AntiVirus Free	• Stop viruses, spyware, ransomware & other malware • block suspicious links, • prevent suspect downloads, • keep suspicious email attachments from being used	https://www.avg.com/en-us/free-antivirus-download
	Bitdefender Antivirus Free Edition	• real-time virus shield blocks malicious URLs and uses behavior-based detection to protect against potential threats • active application monitoring, • anti-fraud and phishing options • excellent malware blocking and removal, • anti-rootkit	https://www.bitdefender.co.uk/solutions/free.html
Penetration testing	Kali Linux	• Debian-based Linux distribution aimed at advanced Penetration Testing and Security Auditing. Kali Linux contains several hundred tools which are geared towards various information security tasks. • Penetration Testing, • Security research, • Computer Forensics • Reverse Engineering.	https://www.kali.org https://tools.kali.org/tools-listing

		• information gathering • vulnerabitly analysis, • wifi attacks, • web apps tools • exploitation • stress test • sniffing and spoofing, • password attacks, • maintating attacks • reporting tools • hardware hacking	
Network analyzers	Wireshark	• analyzes network protocols and sniffs the network in real-time to assess the presence of vulnerabilities • scrutinizing all details related to network traffic at different levels, ranging from the connection level to all pieces of data packets	https://www.wireshark.org/
Password security auditing and recovery	John the Ripper	• it can run on fifteen different platforms	https://www.openwall.com/john/
Network defense	Netstumbler	• identify open ports on a network on Windows	https://www.netstumbler.com/
	Aircrack-ng	• Aircrack-ng contains a comprehensive set of utilities used to analyze the weaknesses of Wi-Fi network security	https://www.aircrack-ng.org/
	KisMAC	• wireless network security in the MAC OS X operating system	https://kismac-ng.org
Web vulnerabilities scanning	Nmap	• scan networks and IT systems to identify existing security vulnerabilities • mapping out potential attack surfaces on a network and monitoring service or host uptime	https://nmap.org
	Nikto	• web server scanner which performs comprehensive tests against web servers for multiple items, including over 6700 potentially dangerous files/programs, checks for outdated versions of over 1250 servers, and version specific problems on over 270 servers	https://cirt.net/Nikto2
	Wapiti	• File disclosure (Local and remote include/require, fopen, readfile...) • Database Injection (PHP/JSP/ASP SQL Injections and XPath Injections) • XSS (Cross Site Scripting) injection (reflected and permanent) • Command Execution detection (eval(), system(), passtru()...) • CRLF Injection (HTTP Response Splitting, session fixation...) • XXE (XML External Entity) injection • SSRF (Server Side Request Forgery) • Use of know potentially dangerous files (thanks to the Nikto database) • Weak .htaccess configurations that can be bypassed • Presence of backup files giving sensitive information (source code disclosure) • Shellshock (aka Bash bug) • Open Redirects • Uncommon HTTP methods that can be allowed (PUT)	https://wapiti.sourceforge.io/

		• Generates vulnerability reports in various formats (HTML, XML, JSON, TXT...) • Can suspend and resume a scan or an attack (session mechanism using sqlite3 databases) • Can give you colors in the terminal to highlight vulnerabilities	
	OWASP ZAP (Zed Attack Proxy)	• penetration testing tool for web apps • ZAP Desktop UI • ZAP will proceed to crawl the web application with its spider and passively scan each page it finds. Then ZAP will use the active scanner to attack all of the discovered pages, functionality, and parameters • Intercepting Proxy • Active and Passive Scanners • Traditional and Ajax Spiders • Brute Force Scanner • Port Scanner • Web Sockets	https://www.zaproxy.org/ https://www.zaproxy.org/addons/ https://www.zaproxy.org/docs/api/
	Vega	• GUI based, written in Java, and runs on Linux, OS X, and Windows • automated scanner automatically crawls websites, extracting links, processing • intercepting proxy allows for detailed analysis of browser-application interaction	https://subgraph.com/vega/
Network Intrusion detection and prevention	Snort	• uses a series of rules that help define malicious network activity and uses those rules to find packets that match against them and generates alerts for users • protocol analysis, content searching and matching. The program can also be used to detect probes or attacks, including, but not limited to, operating system fingerprinting attempts, semantic URL attacks, buffer overflows, server message block probes, and stealth port scans.	https://www.snort.org/