This project has received funding from the European Union’s Horizon 2020 research and innovation programme under the Grant Agreement No 856691.

<table>
<thead>
<tr>
<th>Grant Agreement No</th>
<th>Acronym</th>
<th>5G-SOLUTIONS</th>
</tr>
</thead>
<tbody>
<tr>
<td>856691</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Full Title**: 5G solutions for European Citizens

**Start Date**: 01/06/2019

**Duration**: 36 months

**Project URL**: https://www.5gsolutionsproject.eu/

**Deliverable**: D3.1C (KPI Visualisation System Specifications and Design v3.0)

**Work Package**: WP3 (Technological Enablers for the Living Labs)

**Contractual due date**: M17 (31/10/2020)

**Actual submission date**: 27/10/2020

**Nature**: Report

**Dissemination Level**: Public

**Lead Beneficiary**: AppArt

**Responsible Author**: Kostis Tzanettis

**Contributions from**: Kostis Tzanettis (AppArt), Konstantinos Chalatsis (AppArt), Saman Feghhi (LMI), Ramy Mohamed (IBM), Christos Skoufis (EBOS), Martin Tolan (WIT), Takis Apostolopoulos (UoP)
### Revision history (including peer reviewing & quality control)

<table>
<thead>
<tr>
<th>Version</th>
<th>Issue Date</th>
<th>% Complete</th>
<th>Changes</th>
<th>Contributor(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>v2.8</td>
<td>15/09/2020</td>
<td>5%</td>
<td>Initial Deliverable Structure and Table of Contents</td>
<td>Kostis Tzanettis (AppArt),</td>
</tr>
<tr>
<td>v2.9</td>
<td>17/9/2020</td>
<td>15%</td>
<td>Update of the user flow on section 3</td>
<td>Kostis Tzanettis (AppArt)</td>
</tr>
<tr>
<td>v3.0</td>
<td>18/9/2020</td>
<td>30%</td>
<td>Update of the architecture on section 4</td>
<td>Kostis Tzanettis (AppArt)</td>
</tr>
<tr>
<td>v3.1</td>
<td>21/9/2020</td>
<td>40%</td>
<td>Update of the integrations on section 4</td>
<td>Konstantinos Chalatsis (AppArt)</td>
</tr>
<tr>
<td>v3.2</td>
<td>30/09/2020</td>
<td>45%</td>
<td>Contribution in section 4.4.1</td>
<td>Saman Feghhi (LMI)</td>
</tr>
<tr>
<td>v3.3</td>
<td>30/09/2020</td>
<td>50%</td>
<td>Contribution in section 4.4.1</td>
<td>Ramy Mohamed (IBM)</td>
</tr>
<tr>
<td>v3.4</td>
<td>30/09/2020</td>
<td>60%</td>
<td>Contribution to section 4.5.3.1 regarding the integration with the 5G-VINNI facility</td>
<td>Christos Skoufis (EBOS)</td>
</tr>
<tr>
<td>v3.4</td>
<td>6/10/2020</td>
<td>65%</td>
<td>Contribution in section 4.5.4 regarding the deployment architecture</td>
<td>Martin Tolan (WIT)</td>
</tr>
<tr>
<td>v3.5</td>
<td>6/10/2020</td>
<td>70%</td>
<td>Added the user equipment section 4.1.2</td>
<td>Kostis Tzanettis (AppArt)</td>
</tr>
<tr>
<td>v3.6</td>
<td>6/10/2020</td>
<td>75%</td>
<td>Contribution to section 4.6</td>
<td>Christos Skoufis (EBOS)</td>
</tr>
<tr>
<td>v3.7</td>
<td>12/10/2020</td>
<td>80%</td>
<td>Updated contribution to section 4.6</td>
<td>Christos Skoufis (EBOS)</td>
</tr>
<tr>
<td>v3.8</td>
<td>13/10/2020</td>
<td>85%</td>
<td>Updated the screenshots in section 3.4</td>
<td>Konstantinos Chalatsis (AppArt)</td>
</tr>
<tr>
<td>v3.9</td>
<td>13/10/2020</td>
<td>90%</td>
<td>Updated the 5G VINNI Patras facility integration in section 4.5.3.1</td>
<td>Takis Apostolopoulos (UoP)</td>
</tr>
<tr>
<td>v4.0</td>
<td>14/10/2020</td>
<td>95%</td>
<td>Final check before quality check</td>
<td>Kostis Tzanettis (AppArt)</td>
</tr>
<tr>
<td>v4.1</td>
<td>15/10/2020</td>
<td>100%</td>
<td>Quality check</td>
<td>Christos Skoufis (EBOS)</td>
</tr>
</tbody>
</table>

### Disclaimer

The content of the publication herein is the sole responsibility of the publishers and it does not necessarily represent the views expressed by the European Commission or its services.

While the information contained in the documents is believed to be accurate, the authors(s) or any other participant in the 5G Solutions consortium make no warranty of any kind with regard to this material including, but not limited to the implied warranties of merchantability and fitness for a particular purpose.

Neither the 5G Solutions Consortium nor any of its members, their officers, employees or agents shall be responsible or liable in negligence or otherwise howsoever in respect of any inaccuracy or omission herein.

---

1 According to 5G-SOLUTIONS Quality Assurance Process:
1 month after the Task started: Deliverable outline and structure
3 months before Deliverable’s Due Date: 50% should be complete
2 months before Deliverable’s Due Date: 80% should be complete
1 months before Deliverable’s Due Date: close to 100%. At this stage, it is sent for review by 2 peer reviewers
Submission month: All required changes by Peer Reviewers have been applied, and goes for final review by the Quality Manager, before submitted
Without derogating from the generality of the foregoing neither the 5G Solutions Consortium nor any of its members, their officers, employees or agents shall be liable for any direct or indirect or consequential loss or damage caused by or arising from any information advice or inaccuracy or omission herein.

**Copyright message**

© 5G Solutions Consortium, 2019-2022. This deliverable contains original unpublished work except where clearly indicated otherwise. Acknowledgement of previously published material and of the work of others has been made through appropriate citation, quotation or both. Reproduction is authorised provided the source is acknowledged.

---

**Table of Contents**

1 Executive Summary .......................................................................................................................... 7
2 Introduction ...................................................................................................................................... 8
   2.1 Mapping Projects’ Outputs ........................................................................................................... 8
   2.2 Deliverable Overview and Report Structure ................................................................................ 9
   2.3 Progress Since the Second Deliverable Version ......................................................................... 10
3 Usage of KPI Visualisation System ............................................................................................... 11
   3.1 Interactive Elements .................................................................................................................. 11
   3.2 Data Flow ................................................................................................................................... 11
   3.3 User Flow ................................................................................................................................... 13
   3.4 Screenshots ................................................................................................................................. 15
   3.5 Filters & KPIs .............................................................................................................................. 20
4 KPI Visualisation System and Design ........................................................................................... 23
   4.1 Management & Configuration .................................................................................................... 23
      4.1.1 KPI target values ................................................................................................................. 23
      4.1.2 User Equipment .................................................................................................................... 23
      4.1.3 User Administration ........................................................................................................... 24
   4.2 Platform Security Protocols ...................................................................................................... 24
      4.2.1 Authentication & Authorisation ......................................................................................... 24
      4.2.2 Security Measures .............................................................................................................. 25
   4.3 User Management ....................................................................................................................... 27
      4.3.1 Roles .................................................................................................................................... 27
      4.3.2 Access Rights ...................................................................................................................... 27
      4.3.3 Audit .................................................................................................................................... 29
   4.4 Analytics Engine .......................................................................................................................... 30
      4.4.1 Container based implementation .......................................................................................... 31
   4.5 Architecture ............................................................................................................................... 32
      4.5.1 High Level Architecture ..................................................................................................... 32
      4.5.2 Network & Topology Architecture ....................................................................................... 33
      4.5.3 KPI Visualisation System Integrations ............................................................................... 36
      4.5.4 Deployment Architecture ................................................................................................... 46
   4.6 Design Principles and Technologies Used .................................................................................. 54
      4.6.1 GUI Design Principles ......................................................................................................... 54
      4.6.2 Operating System and Programming Languages ................................................................ 55
List of Figures

Figure 1: KPI Visualisation System Data Flow ................................................................. 12
Figure 2: KPI Visualisation System User Flow ................................................................. 14
Figure 3: Login Page ........................................................................................................... 15
Figure 4: Landing page ....................................................................................................... 16
Figure 5: UC4.1 Dashboard ............................................................................................... 17
Figure 6: User Management .............................................................................................. 18
Figure 7: Create user ......................................................................................................... 19
Figure 8: User access to LL use cases ............................................................................... 19
Figure 9: Date / Time selection ......................................................................................... 20
Figure 10: Test ID selection .............................................................................................. 20
Figure 11: Area selection .................................................................................................. 20
Figure 12: Building selection ............................................................................................ 21
Figure 13: KPI comparison to target value ....................................................................... 23
Figure 14: Authentication & Authorization process ......................................................... 25
Figure 15: SSH public key authentication ......................................................................... 26
Figure 16: High-level ML based workflow in the KPI VS ................................................. 30
Figure 17: 5G-SOLUTIONS Containerized ML engine ....................................................... 32
Figure 18: KPI Visualisation System’s High-Level Architecture ....................................... 32
Figure 19: KPI Visualisation System’s Network Architecture ........................................... 34
Figure 20: Integration options .......................................................................................... 36
Figure 21: 5G-EVE high level framework for handle test .................................................. 38
Figure 22: 5G-VINNI Norway and VS Integration ............................................................. 39
Figure 23: LL2 Use case 2.1, test 2.1.3 dataflow .............................................................. 41
Figure 24: LL3 Use case 3.4 phase 1 of deployment dataflow .......................................... 43
Figure 25: LL3 Use case 3.4 phase 2 of deployment dataflow .......................................... 44
Figure 26: Build process example sequence diagram ...................................................... 48
Figure 27: CI / CD process ............................................................................................... 49
Figure 28: Differences between virtual machines and containers .................................... 49
Figure 29: Amazon sample costs ..................................................................................... 51
D3.1C KPI visualisation system specifications and design (v3.0)

Figure 30: Azure sample costs ................................................................. 52
Figure 31: Google Cloud sample costs ....................................................... 52
Figure 32: Rackspace sample costs ............................................................ 52
Figure 33: 5G-SOLUTIONS Roadmap ......................................................... 60

List of Tables

Table 1: Adherence to 5G-SOLUTIONS GA Deliverable & Tasks Descriptions ...........................................8
Table 2: KPI Visualisation System interactive elements .................................... 11
Table 3: Associated KPIs documented on D1.1A ........................................... 21
Table 4: User actions .............................................................................. 27
Table 5: UC3.4 metrics ........................................................................... 44
Table 6: UC4.3, UC4.4, UC4.6 metrics ...................................................... 45
Table 7: Sampling of available cloud services ............................................. 50
Table 8: Example VM options for cloud service ......................................... 51

Glossary of terms and abbreviations used

<table>
<thead>
<tr>
<th>Abbreviation / Term</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>5G</td>
<td>Fifth Generation (mobile/cellular networks)</td>
</tr>
<tr>
<td>API</td>
<td>Application Program Interface</td>
</tr>
<tr>
<td>CD</td>
<td>Continuous Deployment</td>
</tr>
<tr>
<td>CDN</td>
<td>Content Delivery Network</td>
</tr>
<tr>
<td>CDSO</td>
<td>Cross-Domain Service Orchestrator</td>
</tr>
<tr>
<td>CI</td>
<td>Continuous Integration</td>
</tr>
<tr>
<td>DA</td>
<td>Deployment Architecture</td>
</tr>
<tr>
<td>DoA</td>
<td>Description of Actions</td>
</tr>
<tr>
<td>DSM</td>
<td>Demand Side Management</td>
</tr>
<tr>
<td>GA</td>
<td>Grant Agreement</td>
</tr>
<tr>
<td>GDPR</td>
<td>General Data Protection Regulation</td>
</tr>
<tr>
<td>GPS</td>
<td>Global Positioning System</td>
</tr>
<tr>
<td>GUI</td>
<td>Graphical User Interface</td>
</tr>
<tr>
<td>IE</td>
<td>Interacting Elements</td>
</tr>
<tr>
<td>INT</td>
<td>Interactions</td>
</tr>
<tr>
<td>KPI</td>
<td>Key Performance Indicator</td>
</tr>
<tr>
<td>KPI VS</td>
<td>Key Performance Indicator Visualisation System</td>
</tr>
<tr>
<td>LL</td>
<td>Living Lab</td>
</tr>
<tr>
<td>MANO</td>
<td>Management and Orchestration</td>
</tr>
<tr>
<td>ML</td>
<td>Machine Learning</td>
</tr>
<tr>
<td>Acronym</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td>NFVI</td>
<td>Network Functions Virtualization Infrastructure</td>
</tr>
<tr>
<td>OS</td>
<td>Operating System</td>
</tr>
<tr>
<td>OSGi</td>
<td>Open Services Gateway initiative</td>
</tr>
<tr>
<td>OSM</td>
<td>Open Source MANO</td>
</tr>
<tr>
<td>RAN</td>
<td>Radio Access Node</td>
</tr>
<tr>
<td>REST</td>
<td>Representational State Transfer</td>
</tr>
<tr>
<td>UC</td>
<td>Use Case</td>
</tr>
<tr>
<td>UE</td>
<td>User Equipment</td>
</tr>
<tr>
<td>UML</td>
<td>Unified Modelling Language</td>
</tr>
<tr>
<td>VCA</td>
<td>VNF Configuration and Abstraction</td>
</tr>
<tr>
<td>VCS</td>
<td>Version Control System</td>
</tr>
<tr>
<td>VS</td>
<td>Visualisation System</td>
</tr>
<tr>
<td>VNF</td>
<td>Virtual Network Function</td>
</tr>
<tr>
<td>VM</td>
<td>Virtual Machine</td>
</tr>
<tr>
<td>WP</td>
<td>Work Package</td>
</tr>
</tbody>
</table>
1 Executive Summary

This document is the third and final version (D3.1C) of a set of Deliverables (D3.1A, B and C) called “KPI Visualisation System Specifications and Design” and describes the work performed under the respective Task (T3.1) of WP3. The KPI Visualisation System enables the automation and analysis of the set of 5G-Solutions test cases’ results, provides a graphical representation to the end user, while also including a user management module for controlling the access rights of the users to the data associated with each use case.

As the title states, this deliverable’s target is to provide a high-level description of the KPI Visualisation System along with its functionality requirements and how these are translated into detailed technical specifications.

The document is based on the initial information gathered so far (up to month M17) from the project’s activities even though there is still a lot of work in progress. For this reason, this deliverable has taken into account mainly the following documents:

- Description of Actions (DoA)
- D1.1A “Definition and analysis of the use cases / scenarios and corresponding KPIs based on LLs” v.1.0 (the latest version currently being re-written)

Based on information from the aforementioned documents, along with the activities within WP3, this last version of the requirements of the KPI Visualisation System is produced. The current version of the KPI Visualisation System design is presented with emphasis on the security measures applied within the platform, the internal user roles, along with the actions permitted per role, and the architecture of the platform. Though the integration of the KPI VS with the use case applications and the underlying 5G facilities is part of task T3.3 (and thus the series of deliverable D3.2), a high-level description of the integration with each one of the aforementioned components is also provided here as this is a crucial part of the design of the platform. Finally, a list of the technologies and libraries planned to be utilized is documented along with the implementation plan and deployment techniques followed.

The delivery of this document, in three drops, served the purpose of describing the process of designing the KPI VS as the project was ongoing. With this final version, the deliverable has reached its final version and intends to provide the reader with the design of the KPI VS. Since the capturing of the UC visualisation requirements is still ongoing and the 5G facilities used (ICT-17 and others) are still under design/development, details provided here include the information available up to end of October 2020 (M17). The intention is for the updated information to be captured in the set of D3.2 deliverables which are related to the actual implementation, integrations and testing of the KPI VS as the 5G-SOLUTIONS and ICT-17 facilities projects move forward.

Each drop of this document picks up from where the previous version left off by enriching and updating the existing content with possible updates/changes applied. The differences with the previous version are listed in section 2.3. Complementary to this deliverable, D3.2 will dive into the details of the design and implementation of the KPI VS to provide the low-level design.
2 Introduction

The aim of this deliverable is to provide the specifications and the design of the KPI Visualisation System and its functionality, on which the final implementation of the system will be based upon. It will be the main input point for task 3.2 “KPI Visualisation System components development, real-time collection and analysis of data feeds, and privacy aspects”, which has its own set of deliverables that includes the low-level design of the system.

With this final version, the reader can identify:

- the use case requirements for the KPI Visualisation System: This includes (i) the flow of the user actions to run an experiment, (ii) the filters that can be applied, (iii) the expected KPIs displayed and the outcome format (plots, tables etc.);
- the users of the platform along with the actions permitted for each role;
- the high-level architecture of the KPI Visualisation System and its integrations with the 5G facilities, the onboarded use cases and the CDSO;
- the implementation and deployment plan of the visualisation system;
- the main technologies and frameworks used.

2.1 Mapping Projects’ Outputs

The purpose of this section is to map 5G-SOLUTIONS Grant Agreement commitments, both within the formal Deliverable and Task description, against the project’s respective outputs and work performed.

<table>
<thead>
<tr>
<th>Project GA Component Title</th>
<th>Project GA Component Outline</th>
<th>Respective Document Chapter(s)</th>
<th>Justification</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Task 3.1 - KPI Visualisation System specifications and design</strong></td>
<td>This task will focus on capturing and documenting the functionality requirements of the KPI Visualisation System and translating them into detailed technical specifications, which will be used to develop the low-level design of the system.</td>
<td>Chapter 3 &amp; 4</td>
<td>This deliverable gives a first indication of all the actions that can be performed by the users (login/logout, user management, use case runs, platform configurations). Chapter 3 focuses on how the user will interact when running the use case. Chapter 4 refers to the other actions available within the platform.</td>
</tr>
<tr>
<td></td>
<td>A mapping of the descriptive list of all the use cases regarding the system functionality and requirements with the respective Unified</td>
<td>Chapter 3</td>
<td>This section provides UML diagrams of the data and user flows within the KPI Visualisation System when running a use case along with some screenshots.</td>
</tr>
</tbody>
</table>
### Deliverable Overview and Report Structure

The structure of this document follows the necessary steps to start formulating the detail design of the KPI Visualisation System. The sections defined in this final drop are:

- **Chapter 2 “Introduction”** introduces the reader into the document while providing a mapping between the requirements of task T3.1 and deliverable D3.1 and how this information is provided within the context of this document.

- **Chapter 3 “Usage of KPI Visualisation System”** offers an insight to the reader of the use cases and how they are displayed within the KPI Visualisation System. More specifically, information provided includes UMLs of the user flow within the KPI Visualisation System, some initial screenshots designed and implemented along with some foreseen filters that can be applied in the KPI visualisation.

- **Chapter 4 “KPI Visualisation System and design”** provides a high-level description of the platform in the following areas:

| Modelling Language diagrams will be provided. | The definitions of the roles available in the system alongside with the access rights of each role will be decided in this task. | Chapter 4.3 refers to all the roles of the platform and their related actions. There is a brief description of all the roles that reside within the platform and a table indicating the actions along with a mapping to the relevant roles. |
| A high-level analysis of the technical architecture of the system along with the technologies utilized will also be provided. | Chapter 4.3 refers to all the roles of the platform and their related actions. There is a brief description of all the roles that reside within the platform and a table indicating the actions along with a mapping to the relevant roles. | Chapters 0, 4.6.2 and 4.6.3 |
| DELIVERABLE | DELIVERABLE | DELIVERABLE |
| **D3.1C: KPI Visualisation System Specifications and Design** | A report containing the detailed specifications for the KPI Visualisation System | Chapter 3 contains the use case requirements from the platform including some filtering options, a reference to the required KPIs and thresholds, UML diagrams indicating the data and user flow and some indicative screenshots of the current version of the screens. |

**Chapter 4 refers to the user roles, the user actions and the mapping between them. A high-level architecture is provided along with the technologies, applications and libraries to be used.**
Section 4.1 describes the configurations and available use case management options;

Section 4.2 describes the security protocols applied in the platform to ensure data integrity and privacy;

Section 4.3 provides an insight of the user roles existing within the platform and actions permitted by each one;

Section 4.4 provides a high-level insight of the machine learning algorithms chosen and developed as part of the KPI VS along with the implementation approach;

Section 4.5 provides a high-level architecture of the KPI Visualisation System and its integrations along with the deployment architecture followed within the project;

Section 4.6 provides some principles the user interface is based upon along with the technologies and libraries used;

Section 4.7 describes the implementation plan followed throughout the project duration.

Chapter 5 “Conclusions and next actions” summarizes the results achieved within this deliverable and points out the next actions within the scope of the respective WP3.

2.3 Progress Since the Second Deliverable Version

Deliverable D3.1C, due M17, is the last of the set of three deliverables from D3.1 “KPI Visualisation System and Design”. The initial version was submitted in M6 and the second version was submitted in M11. During these 6 months between drops B and C, various updates/changes have been decided, designed and added in this final drop. Updates have been included throughout the entire deliverable. The major changes/updates are highlighted in this section.

In section 3, the flow of the user within the KPI Visualisation System has been updated. It is now separated into the user flow and the data flow. The first one covers the actions performed by the user whereas the latter describes the data ingestion in the KPI Visualisation System. The first version of screenshots in drop D3.1B have been replaced with the most recent screenshots of the user management section, the login and landing page along with the first UC dashboards.

In section 4, a high-level description of the analytics engine has been included that will be further elaborated in D3.2. Also, an update in the KPI VS architecture has been provided by updating the description of the comprising modules along with an update on the integration status with the 5G facilities, the use cases and the CDSO.

In section 4.6 the technologies and libraries have been updated based on what has been further decided to be leveraged removing also the technologies included in the previous drops that have been opted out.
3 Usage of KPI Visualisation System

This section aims to describe the data and user flows within the KPI Visualisation System. The various interacting elements are outlined and UML diagrams, as well as indicative screenshots, are provided. Additionally, the section focuses on specifying the kind of filters that will be available to users of the KPI Visualisation System along with the hierarchical organization of KPIs into use cases and living labs that will be adopted.

3.1 Interactive Elements

For all use cases, the user and data flow within the KPI Visualisation System are generally common. Of course, minor differentiation may be needed per use case as they onboard the KPI VS. The list of interactive elements covering the entire scope of the KPI Visualisation System including the data ingestions and the user interaction are referenced below in Table 2.

<table>
<thead>
<tr>
<th>Identifier</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>EU</td>
<td>End user</td>
<td>The end user of the 5G-SOLUTIONS KPI Visualisation System interested in viewing the results of a test case scenario.</td>
</tr>
<tr>
<td>KPI-VS</td>
<td>5G-SOLUTIONS platform</td>
<td>Represents the entire group of modules comprising the 5G-SOLUTIONS KPI Visualisation System.</td>
</tr>
<tr>
<td>5G Site</td>
<td>5G site facility</td>
<td>Represents the (ICT-17 projects) 5G facilities or the deployed self-contained nodes used by 5G-SOLUTIONS to deploy each use case.</td>
</tr>
<tr>
<td>UC-App</td>
<td>Use case application</td>
<td>The application running the use case. Responsible for providing the application level data for the KPI calculations</td>
</tr>
<tr>
<td>CDSO</td>
<td>CDSO</td>
<td>The Cross-Domain Service Orchestrator that will indicate the start/stop of the test cases being orchestrated</td>
</tr>
</tbody>
</table>

3.2 Data Flow

The data flow into the KPI Visualisation System can be seen from the flow in Figure 1.
Figure 1: KPI Visualisation System Data Flow

The KPI-VS is notified of a test case start through different channels depending on the UC requirements.

The KPI-VS is notified of a test case stop through different channels depending on the UC requirements.
The steps seen in Figure 1 are the below:

- **Steps 1-3**: the KPI Visualisation System is notified of a test case start. Due to the uniqueness of each UC, the KPI VS provides multiple flows for receiving this notification. The notification can be sent to the KPI VS through the CDSO, from the UC application itself or through the KPI VS portal via manual triggering.

- **Steps 4-6**: the KPI Visualisation System ingests data coming from the underlying 5G facility. The trigger for the data ingestions may come either from the 5G facility (push data) or from the KPI VS (pull data).

- **Steps 7-9**: the KPI Visualisation System ingests data coming from the UC application. Similar to the 5G facility data, the trigger for the data ingestions may come either from the UC (push data) or from the KPI VS (pull data).

- **Steps 10-12**: these steps include the correlation of the data coming in from the UC and the 5G facility, the execution of the necessary KPI calculations and the storage of results in the internal database. The correlation is based on the timestamp of the data and if necessary other information can be used such as the UE IMSI or IP (see 4.1.2).

- **Steps 13-15**: similar to steps 1-3, the KPI VS will receive notifications of the termination of a test case run through the same channel as the “start” notification.

The above flow depicts the scenario where data are ingested in the KPI Visualisation System near real time while the test case is running. There are also cases where the data are ingested offline when the test case has concluded – especially application level data. In this case the flow is similar to the above with the difference that the “stop” test case notification (steps 13-15 above) proceeds the data ingestion & calculation (steps 4-12 above).

### 3.3 User Flow

Through the KPI Visualisation System portal, the user is offered the ability to monitor and analyze the KPIs calculated from the data coming in from the UC applications and the underlying 5G facilities. These views are initially segregated on a UC level and can be further distilled on a test case level. Prerequisites for someone to view the KPIs of a UC is for that user to be authorized to enter the portal and to have access to the specific UC data. Once the user is logged in, they can choose among the UC they are permitted to access. Once the UC is selected, the user flow to view the KPIs is more or less common for all use cases. There are however some variations on the available filters and the start/stop notifications as described above. The flow is depicted below in Figure 2.
The steps captured in Figure 2 are described below:

- **Step 1-2:** the user authenticates themselves in the system. Based on the role returned, the user can perform specific actions based on their access rights granted.
- **Step 3-6:** User is presented with all the UCs he is authorized to access. From this selection, the user chooses a UC of interest.
• **Step 7-8**: each UC is segregated into one or more test cases. If the number is greater than one, the user must first select the test case they want to monitor.

• **Step 9-10**: this is an optional action where the user can notify the KPI-VS through the portal that a test case has started/stopped. It is the one of three options described in the data flow (steps 1-3). If the KPI-VS gets this notification from the CDSO or the application UC automatically, this step is not necessary.

• **Step 11-14**: this is the final step where the user selects the filters of interest. As a first step the user must choose the time window the test was executed in. Once the time window is set, a list of test case IDs (test case run unique identifier) is displayed for the user to choose. As a next step, a list of the available filters (different per test case) is presented (modems, IPs etc.). Once these have been selected too, the user can click to view the outcome. All KPIs are presented in a graphical format along with a set of statistical values (average, standard deviation, minimum, maximum, confidence level). The option for tabular data along with data export is also available for all use cases.

### 3.4 Screenshots

This section provides the reader with an indicative set of screenshots as they have been designed and implemented up to October 2020 (M16). The aim here is to provide the reader with a visual concept of the KPI Visualisation System. A more detailed list of screenshots will be provided in deliverable D3.2. These screenshots are prone to be updated as the project progresses and the feedback along with new requirements come in through the three development phases indicated in section 4.7.

The initial login page of the system for all users is depicted in Figure 3. This page is displayed to the user if they have not logged in, if their session has expired, or if they choose to logout from the portal. They cannot move forward without presenting a valid combination of username and password. If, however, the user is already logged in, this page is skipped and they are forwarded to the landing page (see Figure 4).

![Figure 3: Login Page](image-url)
Figure 4 depicts the page that displays the available use cases within the platform for the user to choose from. The use cases are grouped per living lab. This might also be considered as the landing page, the first page the user sees when logged in. Based on the user role and the respective access rights, the user might not see all the use cases supported by the system but only the ones they have access to.

Figure 4: Landing page

Figure 5 displays the dashboard of a use case (UC4.1) after the user has chosen the UC, the KPI they wish to monitor and the test case of interest to them.
On the top line are the filters already chosen by the user to filter out the output needed. Then, per KPI chosen, a plot is available along with some general statistical analysis (min, max, average, standard deviation, confidence level) and a similar analysis along with the jitter per device (modem, encoder, decoder etc.). The user can zoom in specific time periods and can also view and export the results on a tabular format.

Figure 6 displays the user administration panel. Through this panel, which is accessible to users of a certain role, users of the KPI Visualisation System, grouped per living lab, can be managed. This page will only be accessible to specific users and depending on the user role, not all users might be present in the list. Actions performed through this management section include user addition, deletion, modification, password reset and altering the permitted actions of the user within the KPI Visualisation System.
Figure 6: User Management

Figure 7 and Figure 8 display the initial onboarding of a user to the KPI Visualisation System. This is multistep action. The first step includes basic information of the user such as username, name and surname along with an email address for the system to send any potential emails. Password is not added at this point and it is up to the end user to add this in a second step (see section 4.2.1). The other group of information added here is the use cases the user will have access to in a per Living Lab manner, along with the role he will have in each use case (see below section 4.3). Once the user has been created, they can access the portal with their credentials.
Figure 7: Create user

Figure 8: User access to LL use cases
3.5 Filters & KPIs

In addition to the above designs, through a number of filters, the presented information can be filtered and focused on the visibility needs of the end user. Depending on the use cases, different filters are anticipated to be used. These are listed below:

**date-time period**: a time interval (from, to) where from and to are two timestamps (a timestamp defines year, month, year, hours, minutes and seconds). This is especially important for the end user if the results are not coming in real-time so that he can identify the exact slot the experiment ran.

**test-id**: a unique identifier of a use case run. This will enable the user to view the KPIs related to a specific use case run if there are concurrent cases running.

**area**: describes a geographical area the user can choose by dragging their mouse over the desired area. Such an indicative filter can be seen in Figure 11.
building: a list of monitored buildings (i.e. UC 2.1)

Other anticipated filters include:

- **encoders/decoders**: a list of video encoders/decoders (i.e. UC 4.1)
- **modems**: a list of modems (i.e. UC 4.3, 4.4, 4.6)
- **sensor categories**: a list of sensor categories (i.e. UC 2.1).
- **sensor groups**: a list of sensor groups (i.e. UC 2.1).
- **vehicles**: a list of monitored vehicles (i.e. UC 2.2).
- **fleets**: a list of vehicle fleets (i.e. UC 2.2).
- **charging stations**: a list of monitored charging stations (i.e. UC 2.3).
- **loading areas**: a list of monitored loading areas (i.e. UC 2.3).
- **density**: the testers density per channel (i.e. UC 4.4).

The KPI Visualisation System will provide insights based on a set of KPI requirements that arise from specific vertical domains. More specifically, the Factories of the Future vertical use cases (LL1) are associated with Industry 4.0. The Smart Energy (LL2), Smart Cities & Ports (LL3) use cases are associated with Smart Urban Environment. Finally, the Media & Entertainment (LL4) use cases are associated with the Enhanced User Experience. An additional Multi Living Lab (LL5) will validate 5G performance capabilities from the perspective of multi-vertical concurrent usage of resources across all sets of heterogeneous KPI requirements so as to prove that no performance degradation occurs.

The aforementioned hierarchical organization of KPIs into use cases that are also organized into living labs is also adopted by the KPI Visualisation System, to provide users with an intuitive view and facilitate fast user navigation in the platform. This organization is presented below with references for the KPIs of every use case as documented in the deliverable D1.1A.

<table>
<thead>
<tr>
<th>Living Lab</th>
<th>Use Case</th>
<th>Associated KPIs documented in D1.1A Section</th>
</tr>
</thead>
</table>

Table 3: Associated KPIs documented on D1.1A
| Factories of the Future Living Lab (LL1) | UC1.1: Time-critical process optimization inside digital factories | 5.1.1.3 |
| UC1.2: Non-time-critical Communication inside the Factory | 5.1.2.3 |
| UC1.3: Remotely Controlling Digital Factories | 5.1.3.3 |
| UC1.4: Connected Goods | 5.1.4.3 |
| UC1.5: Rapid deployment, Auto/re-configuration, Testing of New Robots | 5.1.5.3 |
| Smart Energy Living Lab (LL2) | UC2.1: Industrial Demand Side Management | 5.2.1.3 |
| UC2.2: Electrical Vehicle Smart Charging | 5.2.2.3 |
| UC2.3: Electricity Network Frequency Stability | 5.2.3.3 |
| Smart Cities and Ports Living Lab (LL3) | UC 3.1: Intelligent Street Lighting | 5.3.1.3 |
| UC 3.2: Smart Parking | 5.3.2.3 |
| UC 3.3: Smart City Co-Creation | 5.3.3.3 |
| UC 3.4: Smart Buildings – Smart Campus | 5.3.4.3 |
| UC 3.5: Autonomous Assets and Logistics for Smart Port | 5.3.5.3 |
| UC 3.6: Port Safety - Monitor & Detect Irregular Sounds | 5.3.6.3 |
| Media/Entertainment Living Lab (LL4) | UC4.1: Ultra-High-Fidelity media | 5.4.1.3 |
| UC4.2: Multi-CDN | 5.4.2.3 |
| UC4.3: On-site Live Event Experience | 5.4.3.3 |
| UC4.4: User & Machine Generated Content | 5.4.4.3 |
| UC4.5: Immersive and Integrated Media and Gaming | 5.4.5.3 |
| UC4.6: Cooperative Media Production | 5.4.6.3 |
| Multi Living lab (MLL) | 5.5.3 |
4 KPI Visualisation System and Design

4.1 Management & Configuration

The key functionality of the KPI visualisation platform is to provide meaningful and insightful visualisation of KPI results associated with certain use cases to users of the platform. However, the dynamic nature of certain aspects of the platform requires additional functionality with regards to configuration and management. Such functionality will be available to users with sufficient privileges in the platform, as defined by their role. Three scenarios identified that require dynamic configuration are the KPI target values, the test case user equipment (UE) and the user administration. All these scenarios are discussed thoroughly below.

4.1.1 KPI target values

Each KPI that is measured and can be visualized through the platform, is associated with a target value. This value represents the threshold that needs to be reached and will be used to evaluate firstly whether one or more of the strategic goals are met, and secondly how effectively these key objectives are achieved. More specifically, the actual outcome of the KPI value is compared with the target value, in order to provide the user with an insightful visualisation, focusing on where the test outcome stands in comparison with the target value (Figure 13). This is particularly useful as it provides a very intuitive way of assessing whether a service meets the expected standards.

![Figure 13: KPI comparison to target value](image)

The KPI visualisation platform will provide a user-friendly configuration mechanism for setting or updating the target value of every KPI offered through the platform, per use case. Users authorized with appropriate privileges will be able to access a configuration page designed for this purpose. The page will provide a hierarchical view of the KPIs based on their associated use case. In addition, the page will offer a search functionality that will allow users to quickly find their desired KPI if they choose to do so by using a search box that filters out irrelevant KPIs. Finally, having found the KPI value of interest, the user will be able to set this KPI value for that particular use case, in case it has not been previously set, or update it, in case a change is required at some point within the project. The update will be persisted in the KPI visualisation platform and will take immediate effect in visualisation reports generated after the update for all users of the platform.

4.1.2 User Equipment

During the time period for cycle 1 activities and the discussions that were held with the 5G VINNI Patras facility, some new requirements arose including the ability to filter network data based on the UE IMSI or IP address. All network data in the VINNI Patras facility are assigned to a specific IMSI and/or IP. Thus, knowing all the...
IMSI\'s and/or IPs of the equipment used in each test case will allow for the KPI VS to filter the entries relevant to the test case in question. Another objective this information will assist with is the correlation of the network data with the respective application data. Though the above has been currently discussed within the scope of the integration with the VINNI Patras facility, it is anticipated that a similar need will rise for the other facilities also.

The list of IMSIs or IPs must be provided in every test case “start” notification received by the KPI VS, whether this is received from the CDSO, through the custom REST API provided for the UC applications or through the KPI VS portal. In order to avoid the extra step of constantly needing to add a list of UE IMSIs or IPs, a separate menu will be implemented in the portal where a list of UEs will be correlated with a unique name that can be used to tie the test case to the UEs. If the portal is going to be used to start/stop the test case monitoring, the user will have a list of these unique names to choose from through a drop down presented during the start notification process.

4.1.3 User Administration

The KPI visualisation platform will allow for user management through web pages provided via the administration panel. More specifically, administrators authorized via the KPI visualisation platform, will access this page and perform all activities associated with user management, such as user addition, deletion, modification along with assigning permissions to the users.

Adding a user will be performed by completing a form with all necessary basic (username, name, surname, email, etc.) details. After the administrator submits the form, a temporary URL is created and sent to the newly created user to set a password of his own.

Updating or deleting a user will be possible through a page that offers an ordered table view of the users of the platform. This view will feature some details of the user such as the username, along with action buttons. The administrator will be able to search for the user that needs to be modified or deleted in this table by navigating through ordered pages. Alternatively, the administrator will be able to use a search box to quickly find the desired user by filtering out irrelevant ones. Having found the user, the administrator will have the option to open an edit page that will feature a form populated with the user’s current details and will allow editing. This can be particularly useful in case additional privileges need to be granted to a user. Similarly, privileges of a user can be revoked at any time. Apart from the edit functionality, the administration page will allow user deletion thus, completely revoking her/his access throughout the KPI visualisation platform, locking / unlocking platform accounts (prohibiting access but not permanently deleting the account) and resetting their passwords, if required.

4.2 Platform Security Protocols

4.2.1 Authentication & Authorisation

The user authentication within the portal will be performed by providing a set of username and password. This is the first step / page the users see on their screen and cannot move on without providing a correct combination. Once the credentials have been validated, the user is navigated to the landing page which depends on his role in the platform.

The initial registration of the user is performed in two steps. The first step is performed by the administrator of the living lab or use case or the system administrator. Once this is complete, the end user is provided with a link for him to navigate to and complete the registration by entering a password of his choice. Once the second step is complete, the user can navigate to the login page to enter the platform.
Apart from the authentication of the user, the platform also enhances security by providing authorization on all actions performed. All actions are checked whether they can be performed by the user initiating the request. A flow sequence can be seen below in Figure 14. When the user sends a successful login request, the backend authentication service creates a JWT token that is returned in the response. This token that is digitally encrypted using HMAC 512 algorithm, is session-less and has an expiration date. This token is then included in every request for the authentication service to verify the authenticity of the user, the access rights of the user and whether the token is still active, or a new authentication process is required. If for any reason the token is not valid, the user is redirected to the login page. When the user logs out from the platform, the token is blacklisted and cannot be used again.

![Figure 14: Authentication & Authorization process](image)

4.2.2 Security Measures

Since the platform will be available to end users over the internet, a number of security measures will be applied to verify data integrity and security. The mechanisms applied are:

Database Encryption

Table encryption makes it practically impossible to access the data directly from the hard disk if this is compromised. Data-at-Rest encryption will be used wherever available, minimizing the performance overhead and providing an advanced level of security.

Password Hashing

Password hashing will be used throughout the entire platform in order to ensure that no actual passwords are stored or used at any point. Passwords in all requests will be hashed and compared with the hashed version stored in the database. This will ensure that even if the request or database is hacked by a third party, the actual password will not be compromised. In order to achieve the protection of the accounts’ passwords, the
**bcrypt** password hashing function is used. This function is based on the Blowfish cipher and incorporates a salt methodology, in order to protect the passwords against rainbow table attacks. Moreover, *bcrypt* is an adaptive algorithm meaning that it is resistant to brute-force search attacks.

**SSL/TLS**

Transport Layer Security, a cryptographic protocol that provides protection over computer networks, will be used in all possible network transactions. TLS ensures the data privacy along with the data integrity during the transfer. Data remains private because all messages are encrypted using symmetric cryptography while the integrity is ensured using a message authentication code. Such an example is the use of HTTPS instead of HTTP that requires the user verification of the other side before establishing the connection.

**Access via Public Key**

Access to the remote machines will be over SSH with the use of a public key. Login requests with username and password will be forbidden. Public key authentication provides cryptographic strength that even extremely long passwords can’t offer. Also, the password is susceptible to brute-force attacks by malicious users or can be intercepted.

![SSH public key authentication](image)

**Figure 15: SSH public key authentication**

SSH key authentication is based on the asymmetric encryption using two keys, the private and the public. Both are created by the client (the one who needs authorization); one is paired with the other and none can be derived from the other. The client keeps the private key confidential in a secure place while the public key is stored in the remote server(s) to which access is required. Messages are encrypted by the public key and can only be decrypted by the private key (see Figure 15) thus allowing on the valid user to read the message.

**Whitelisted Locations**

Access to the infrastructure is only available over HTTPS and the designated ports for the portal and REST API access. Access to the VM console over SSH and the database is prohibited. However, for development and troubleshooting purposes, access is granted per request by allowing traffic from specific IP locations.

**JSON Web Token (JWT)**

JSON Web Token (JWT) is an open standard (RFC 7519) that defines a way for securely transmitting critical information between parties using the JSON format. By using either the HMAC algorithm (the information is encrypted using a secret key) either the RSA or ECDSA (public / private key pair) the information is digitally signed. Consequently, the content of the data can be verified and trusted. Signed tokens can verify the integrity of the claims contained within it, while encrypted tokens hide those claims from other parties.

---

2 [https://auth0.com/blog/hashing-in-action-understanding-bcrypt/](https://auth0.com/blog/hashing-in-action-understanding-bcrypt/)

Basic Authorization

The Basic authentication scheme will be used for authorization purposes, when each UC uploads their application data into the KPI-VS via the REST-API. The client sends HTTPs requests with an Authorization header, containing the word “Basic” followed by a base64-encrypted string, having the format “username:password”.

4.3 User Management

4.3.1 Roles

The KPI Visualisation System has various roles that segregate users into groups with different access rights assigned to each group. The platform is built in such a way that it can easily support new roles added in the future. Up to this point, the following user roles have been identified to exist in the KPI Visualisation System:

- **System Administrator**: the administrator of the KPI Visualisation System. This role is responsible for managing all the accounts of the system, can manage the platforms’ parameters and can also view all the KPIs related to all use cases.
- **Living Lab Administrator**: this user is created by the system administrator and is assigned to the owner of a particular living lab. This user can manage all use cases and their parameters placed under the corresponding living lab. Furthermore, they can manage the access to the underlying UCs for all users.
- **Use Case Administrator**: this user is created by the living lab administrator and is assigned to particular use case(s). They can manage the use case parameters (e.g. respective KPI thresholds, UEs) and can manage simple users of the use case.
- **Simple User**: a role where they can view reports on the use case(s) they are assigned to.
- **Data Collector User**: this role is used by the Data Collector module and is responsible for uploading application data into the KPI-VS, when the integration with a UC is via a REST-API.

A user account is not restricted to belong in one living lab or one UC only. One user may be an administrator in one living lab, an administrator in a use case of another living lab and a simple user in another use case.

4.3.2 Access Rights

This section provides a complete list of all the actions permitted per user role. The allowed platform actions can be seen below in Table 4.

<table>
<thead>
<tr>
<th>Action</th>
<th>Description</th>
<th>Role(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Login / logout</td>
<td>Login and logout from the KPI Visualisation System</td>
<td>- All</td>
</tr>
<tr>
<td>Create user</td>
<td>Create a new user in the platform</td>
<td>- System administrator</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Use case administrator (for assigned UCs)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Living lab administrator (for all the UCs of the living lab)</td>
</tr>
<tr>
<td>Action</td>
<td>Description</td>
<td>Permissions</td>
</tr>
<tr>
<td>------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>Delete user</td>
<td>Delete a user from the platform</td>
<td>System administrator</td>
</tr>
<tr>
<td>View user</td>
<td>View the profile of a user</td>
<td>System administrator, Use case administrator (for assigned UCs), Simple user (his own profile), Living lab administrator (for all the UCs of the living lab)</td>
</tr>
<tr>
<td>View users</td>
<td>View the list of users belonging to the platform</td>
<td>System administrator, Use case administrator (for assigned UCs), Living lab administrator (for all the UCs of the living lab)</td>
</tr>
<tr>
<td>Add role</td>
<td>Give user access to a UC</td>
<td>System administrator, Use case administrator (for assigned UCs), Living lab administrator (for all the UCs of the living lab)</td>
</tr>
<tr>
<td>Remove role</td>
<td>Remove user access from a UC</td>
<td>System administrator, Use case administrator (for assigned UCs), Living lab administrator (for all the UCs of the living lab)</td>
</tr>
<tr>
<td>Change password</td>
<td>Change own password for access to the platform</td>
<td>All</td>
</tr>
<tr>
<td>Edit user profile</td>
<td>Edit the parameters of the user’s profile</td>
<td>System administrator, Use case administrator (for assigned UCs), Simple user (his own profile), Living lab administrator (for all the UCs of the living lab)</td>
</tr>
<tr>
<td>Reset password</td>
<td>Reset the password of a user</td>
<td>System administrator, Use case administrator (for assigned UCs), Living lab administrator (for all the UCs of the living lab)</td>
</tr>
<tr>
<td>Lock account</td>
<td>Locking an account of the platform</td>
<td>System administrator</td>
</tr>
<tr>
<td>Unlock account</td>
<td>Unlocking an account of the platform</td>
<td>System administrator</td>
</tr>
<tr>
<td>View audit logs</td>
<td>View actions performed by users</td>
<td>System administrator</td>
</tr>
</tbody>
</table>

**Use Case related**
| View test case KPIs | Run reports on selected use cases, view plots and export results | System Administrator  
- Use case administrator (for assigned UCs)  
- Simple user (for assigned UCs)  
- Living lab administrator (for all the UCs of the living lab) |
| Edit KPI thresholds | Edit the thresholds of the use case KPIs | System administrator  
- Use case administrator (for assigned UCs)  
- Living lab administrator (for all the UCs of the living lab) |
| View KPI thresholds | View the thresholds of the use case KPIs | System Administrator  
- Use case administrator (for assigned UCs)  
- Simple user (for assigned UCs)  
- Living lab administrator (for all the UCs of the living lab) |
| Start/Stop UC data monitoring | Start/Stop the data monitoring of a test case run | System administrator  
- Use case administrator (for assigned UCs)  
- Living lab administrator (for all the UCs of the living lab) |
| Add/Edit/Delete UEs | Manage the UE assigned for a test case (IPs, IMSIs) | System administrator  
- Use case administrator (for assigned UCs)  
- Living lab administrator (for all the UCs of the living lab) |

4.3.3 Audit

A security trail of all actions performed (successful or not) within the KPI Visualisation System will be maintained. This log will serve the purpose of keeping track of all activities performed in the KPI Visualisation System. Each action will be associated with a timestamp (date and time the action was performed) as well as the user performing the action. In case the action was automatically performed by the platform itself (ex. Lock user after a certain period of inactivity), a relevant value will be used to indicate the originator.

The resulting log will provide easy access to documentary evidence of all actions performed, such as logging in the platform, requesting use case reports, editing KPI thresholds, etc. In addition to providing a security trail, the logs will offer meaningful insights on user preferences that could be further exploited to provide enhancements in the application. As an example, the information kept will allow for coming up with the most frequently used KPIs per use case. This could potentially lead to reconsidering certain design choices and perform adjustments to the KPI visualisation platform according to the user requirements to satisfy their needs more effectively.

Naturally, this information will only be accessible to users with administrative privileges (only the system administrators).
4.4 Analytics Engine

This feature enables the KPI Visualisation System to present insights coming from machine learning (ML)-based algorithms applied to the monitored KPIs. Planned insights refer to detection of anomalies in time-series data KPIs, correlations between different monitored KPIs, prediction of values for time-series KPIs, etc.

A generic set of ML based algorithms will be developed and will focus on a number of designated KPIs, which are the most common amongst use cases. Such KPIs will include network performance KPIs such as latency and upload/download speed; but also, some application level KPIs such as the service availability. The availability of such data spanning across 20 use cases has the potential to produce a strong model for prediction, anomaly detection as well as correlation extraction. Furthermore, a number of ML models will be built specifically for a selection of pilot use cases in order to showcase the capabilities of the ML optimisation module and the insights it can provide. The selection criteria for each UC will depend on a number of parameters such as the amount of data generated by the Use Case and collected by the KPI Visualisation System, the integration level between the ICT-17 facilities and the KPI VS, as well as the Technology Readiness Level of the use case.

The above-mentioned ML algorithms are planned to be implemented in Python or R, with more focus on Python given its widespread use of data analysis tools thanks to the large number of packages and libraries including Scikit-learn, Pandas, NumPy, PyTorch, Keras, TensorFlow, etc. Some of these libraries are further explained in Section 4.6.3

The KPI VS will incorporate the above algorithms and models and structure them in a user-friendly way. Figure 16 shows a high-level workflow of the ML-based analysis performed within the KPI VS.

![Figure 16: High-level ML based workflow in the KPI VS](image)

The correlation analyses provided as part of the KPI VS will help extract useful insights into the KPI data so as to help with the choice of ML models that are relevant to the problem and the structure of the data. The results of this step will be displayed in a visual format for the user’s convenience. A recommendation system will be implemented in the KPI VS to suggest the most suitable ML algorithms following a correlation analysis performed by the UC owner, based on a set of rules, but also from historic data about the performance of various ML algorithms applied on similarly structured KPI datasets.

The UC owner can choose to perform KPIs forecasting or monitoring by selecting the appropriate algorithm. In case of anomalies detected, the user can perform further root cause analysis by using correlation and/or clustering algorithms available. Please note that the models trained on a similar KPI from multiple UCs can be
used for further training or inferencing in subsequent runs from a different UC/LL. This will considerably improve the efficacy of the models over time as more and more features and knowledge are learned from various UCs and combined into the latest version of the model.

Analytics algorithms are planned to be offered as micro services using containers that can be readily embedded in the main visualisation platform.

4.4.1 Container based implementation

The analytics engine features a range of ML-based algorithms on monitored KPIs. These algorithms can be either generic or use case specific. The generic algorithms are designed to be data and use case agnostic, meaning that they can be applied to any of the monitored KPIs regardless of the details of their use cases. An example of such an algorithm is anomaly detection on time series data that can be applied to any monitored KPI. Algorithms mentioned above (anomaly detection and prediction of future values for time-series KPIs, correlations between KPIs) will be discussed in more detail in D3.2.

In order to facilitate this, algorithms are designed as micro-services using Docker containers (Figure 17). The containers are also planned to use REST APIs to communicate with other components within the KPI Visualisation System. This abstraction allows analytics micro services to be readily pointed at any input data (being online e.g. streaming data or offline e.g. batched data stored in a database) and their results fed back to KPI VS components (e.g. reporting, visualisation or other features) to prepare illustration of output for end-users. This also allows for ease of updates and a scalable solution for the ML engine, as well as allowing for algorithms to be integrated in any other data pipeline to act as a sub-component of a larger service (e.g., integration with the use case platform for use case specific optimisations, or for zero-touch automation purposes).

Figure 17 illustrates the proposed containers and their interaction within the Analytics Engine and with the KPI VS database (DB). The Analytics Engine will include a Data Handler container which is responsible for interfacing with the KPI VS DB. It is also responsible for pre-processing the data that will be used by ML containers. Depending on the size of data, the output of the Data Handler can either be directly provided to ML containers using REST API or can be stored back in the KPI VS DB. The Analytics Engine also contains a suite of containers dedicated to generic ML algorithms. The pre-processed input data, provided either by the Data Handler container or directly from the database, is fed to the aforementioned containers and the algorithms can be run with different parameters and techniques using a REST API while the results can be fed back to the KPI VS DB or consumed by the other components of the Analytics Engine or KPI VS through this REST API.
4.5 Architecture
4.5.1 High Level Architecture

A high-level architecture of the KPI Visualisation System is provided in Figure 18. In the left side, we see that input data will be ingested into the KPI Visualisation System from the 5G facilities (ICT-17 facilities, etc.) as well as the Vertical Industries’ Applications.

The KPI Visualisation System will employ numerous modules to process the information ingested from the 5G facilities and UC applications and make it available through the Web:

- **Data Collector Module**: module responsible for collecting the data from all sources (5G facilities, vertical applications), anonymizing them whenever needed and storing the output in the internal database. It provides interfaces for all external data sources based on the communication protocol used by the remote side. Data is always transferred over the network through a secure, encrypted manner (HTTPS, SCP, SFTP, secured SNMP, etc.).

- **Data Processing Module**: module responsible for performing near real-time processing on the collected data and extracting the valuable information that is persisted to the system’s database. Data is homogenized and transformed to predefined KPIs corresponding to field trials of different use case scenarios. This process includes formula applications, data aggregation on different levels (time and space), statistical analysis of the results along with comparison to predefined threshold values for raising the necessary alarms. There are two data collector instances running in parallel each handling different use cases.

- **Machine Learning Optimization Module**: will receive part of the processed information. This module will utilize machine learning-based methods for forecasting, prediction and fault recognition services (classical and/or deep learning algorithms) to produce measurements for sophisticated KPIs, all of which have been pre-trained offline. After the machine learning techniques have been applied on the data, the results will also be persisted in the system’s database. As the methods/algorithms that can be applied are closely

---

**Figure 17: 5G-SOLUTIONS Containerized ML engine**

**Figure 18: KPI Visualisation System’s High-Level Architecture**
dependant on the available data, the decision on the exact algorithms will be taken as soon as the kind of
data coming in from the 5G facilities and the use case applications becomes available.

- **Web-based Graphical Environment Module:** offers user-friendly, authenticated and authorized access to
the system’s data through secure protocols. It consists of two parts: the web portal, built using state-of-
the-art web technologies that provides to the Living Lab end users an intuitive web-based dashboard, and
the internal REST API for exposing data to the authenticated users in a web manageable format. Once the
end-users are authenticated to have secure access to the cloud-based KPI Visualisation System, they are
offered with a list of use cases they can choose from (depending on their user rights). After a use case is
selected, the user can choose amongst a number of filters (depending on the use case) in order to filter and
view the data in both tabular and graphical format.

### 4.5.2 Network & Topology Architecture

The design of the network architecture of the KPI Visualisation System is illustrated in Figure 19 and is split into
five distinct layers, the 5G Facility layer, the UC application layer, the data collection layer, the business layer
and the presentation layer.
The core components of the KPI Visualisation System will be deployed on the cloud to provide scalability and elasticity. This is depicted in the top part of Figure 19. The network architecture will aim to provide limited bandwidth costs. To this end, the different modules of the visualisation system will be placed – if possible – in the same cloud provider and data center so that the communication between the modules does not induce additional costs. In addition, the architecture will feature automatic resource scaling, by leveraging virtualization technologies in accordance with the workload demands.
Starting from bottom up, the architectural layers are:

**5G Facilities**

The 5G facility infrastructure will comprise of the 5G-EVE Turin facility for conducting field trials for the use cases related to the Smart Energy Living Lab, the 5G-VINNI facilities and self-contained 5G nodes for conducting field trials for the use cases related to the Factories of the Future (Ireland, Brussels, Norway), Smart Cities (Ireland), Smart Ports (Norway) and Media & Entertainment Living Labs (Patra and Norway). As shown in Figure 19, each 5G facility/node will make available its exporting data to the corresponding data collection module on the data collection layer. As the 5G facilities/nodes have different integration protocols implemented for exporting their data, various integration protocols and data loading/processing mechanisms are being implemented to support the diversity of requirements.

**Use Case Applications**

This layer consists of the applications running in each use case and the KPIs measured on their end, such as end to end latency, bandwidth etc. The nature of each use case, along with the usage of commercial applications in a number of use cases, making it difficult to adapt to external integrations, have led to a variety of integration designs with the KPI Visualisation System such as REST API, file uploading, remote database querying, SNMP queries etc. Some UC applications running on 5G VINNI Norway may populate application layer data in a PostgreSQL database offered by the VINNI Norway platform as a TaaS service. In such cases the application data are collected from the facility infrastructure.

**Data Collection Layer**

This layer consists of the data collection module. It consists of two instances each one handling different use cases. The segregation is based on the underlying 5G facilities, with one handling use cases running on 5G VINNI Patras and 5G EVE and the other one handling use cases running on 5G VINNI Norway. Both instances integrate with their respective endpoints using various communication protocols, sampling rates and handling multiple data formats. All data ingested are then homogenized and persisted in the internal KPI VS database for future KPI calculations and previewing through the portal.

This module also includes the integration with the CDSO for receiving notifications regarding a start / stop of a test case execution. For UCs with no orchestration, an alternative flow is designed by providing a REST API available both from the UC application and from the KPI VS portal for manual starting/stoping the data collection related to a test case. Finally, the data API is also located in this layer for providing the database data to the upper business layer in a secure transparent way.

In the early stage of the project, before the data volume of most UCs was known, the design also included a “proxy” layer (part of the KPI VS) located on the 5G facility / UC premises that would filter and aggregate the data before they were ingested in the cloud KPI VS. This would possible minimize the amount of data travelling towards the core KPI VS reducing the requirements for storage and network costs induced by a cloud provider. For the use case integration designed up to now, there has been no need of this extra processing layer (thus not depicted in the architecture in Figure 19).

**Business Layer**

This consists of 4 submodules, the KPI calculations, the machine learning algorithms, the user management and the internal REST API. The KPI calculations submodule performs all the necessary calculations, separately for each UC, to produce the desired KPIs based on the data ingested while the machine learning algorithms run on top of all the available data to produce insights on the KPIs. At the end, all outputted data are stored in the
internal storage. The user management component handles the management of all the users of the KPI VS and their access rights. Finally, the internal REST API works as a proxy between the data layer and the presentation layer for authorizing, fetching and transforming the data for usage from the upper presentation layer.

**Presentation Layer**

This is the graphical user interface responsible for getting the data from the internal database via the internal REST API and presenting them in all necessary formats (tabular, plots, exports) taking into account the user experience. Access to specific sections of the portal is provided through the authentication / authorization mechanism available.

### 4.5.3 KPI Visualisation System Integrations

The KPI visualisation platform will integrate with several testbed related endpoints:

- two 5G ICT-17 Platforms (5G-EVE & 5G-VINNI)
- 5G self-contained nodes
- the CDSO
- and various 5G-SOLUTIONS LL/UC applications to collect the necessary data for the KPI calculations.

This section aims to introduce the different levels of integration with the KPI Visualisation System for completeness of the system design. A more complete description can be found in deliverable D3.2 which covers the integration task. The integration with each entity will be realized with the most fitting of the options that are illustrated in Figure 20 (but not limited to) and discussed below:

**File-based Integration**

The file-based integration option considers that the source data are represented using a file that is pushed from the source to the destination. File-based integration allows data to be passed between different
databases without the need for direct connectivity. Support can be provided either for flat files, e.g., CSV format, that are used for tabular data, as well as structured files, e.g., JSON and XML.

**REST Integration**

API integration allows for connecting two systems by interpreting and translating data between them via their APIs. An API specifies the standard way for a developer to request services from another system or application. A REST (REpresentational State Transfer) API accepts HTTP requests to GET, PUT, POST or DELETE data. REST is often used to build APIs that allow users to connect and interact with cloud services. SOAP (Simple Object Access Protocol) is another protocol used to develop APIs, that uses an Envelope element (XML) that features a body and a header. In the case of API integration, both the push and pull architectures can be used, depending on the specific requirements.

**Streaming Data Integration**

The ever-increasing application data generation calls for low-latency platforms that can handle voluminous real-time data feeds. Apache Kafka provides a unified, high-throughput, low-latency platform for handling such feeds, and is ideal for persistent messaging, fast reads and writes, and numerous clients. Essentially, Kafka is very fitting for cloud-scale architectures.

Kafka uses the notion of a Kafka broker cluster, i.e., one or more servers where one or more broker processes are running. Apache Kafka is highly available as all nodes are interchangeable and there are no master nodes. Kafka uses topics which are partitioned. Producers send messages to Apache Kafka broker topics and specify the partition to use for every message they produce. Message production may be synchronous or asynchronous. Producers also specify what sort of replication guarantees they want. Consumers listen for messages on topics and process the feed of published messages and are usually asynchronous.

**Database Integration**

In cases where the data are persisted using a database management system at their source, there is the option of establishing a direct connection and issuing queries to this database. The queries must be issued from the destination side in a manner that guarantees that no duplicate information is transferred. This can be achieved by using time intervals in the queries issued. Depending on the database used, the queries may filter and project only the necessary information in an effort to reduce the overall data transfer.

**Web Socket Integration**

Web Sockets represent a standard for the communication between servers and clients. They allow a persistent TCP connection between a server and a client over which both parties can start sending data at any time. The messages are exchanged with very low latency and they are distributed with little overhead. The client establishes a Web Socket connection through a process called Web Socket handshake, which starts by sending an HTTP request to the server. This HTTP request should contain an “Upgrade” header.

**SNMP Integration**

SNMP stands for Simple Network Management Protocol and it is an application-layer protocol, defined by the Internet Architecture Board (IAB) in RFC1157. SNMP protocol is used for exchanging management information between network devices. It defines the methodology of collecting information describing the managed devices, as well as changing their behaviour. Furthermore, system variables and configurations are exposed via
the management information base file (MIB). This information can be used by applications to query and manipulate components of the system.

4.5.3.1 5G Facilities Integration

5G-EVE

5G EVE infrastructure is the facility proposed to provide field trial connectivity to the LL2 use cases. In Figure 21 below, a high-level representation of the framework used in 5G-EVE infrastructure to handle the experiment test and validation (additional detail can be provided in [3]) is shown.

The only component that will be exposed to 5G-SOLUTIONS is the 5G EVE Visualization Tool representing the 5G network KPIs. In particular, the integration plan is to embed the 5G EVE Visualization data into the 5G SOLUTIONS Visualization tool.

![Figure 21: 5G-EVE high level framework for handle test](image)

The “Experiment Automation Framework” can interact with both Traffic Generation and Monitoring Tools. These tools send information to the Metrics Broker built using the “Apache Kafka Bus”. Subscribing on a specific Kafka topic allows the reception of system KPI and performance KPIs associated with the topic. The subscription is allowed only to the module of 5G EVE platform.

The Validation Tools exploits information notified by the Broker to determine the correctness of each executed experiment, validate the test result and provide the report. This module exposes an internal API towards the other platform entities in order to provide them with operational control. This API includes a list of topics for subscription in order to select the KPI for the validation process and the information contained in the final report log file.

The topic subscription at the Metric Broker is used by the 5G EVE Visualization tools to show system performance. In order to allow external applications access to collect information on the infrastructure performance a coordination with 5G-EVE project is required asking for the exposure of an external API (for example enabling the topic subscription or the access to some log files). The 5G-EVE platform at the moment doesn’t provide external interface for monitoring purpose, and because of that the collaboration with 5G-EVE on this topic is currently in its initial phase.

5G-VINNI Norway

The 5G-VINNI integration with the Testing as a Service system (TaaS), a system for performing tests within the 5G-VINNI Norway virtualized network, has been initiated to collect measurement data and visualization service KPIs. TaaS can also be used to perform KPI measurements by sending simulated traffic between so called Hawkeye end-points (virtual probes) that can be installed both inside and outside of the 5G-VINNI network.
The integration of the 5GSOLUTIONS KPI visualization system with the 5G-VINNI TaaS system will only be partly implemented in Cycle 1.

Moreover, for Cycle 1, as shown in Figure 22, a Postgres Database has been setup and hosted in 5G VINNI Norway facility, in order to collect network and application level data from the UCs. This database has a specific structure which the UCs need to comply with, while performing the integration. Currently, only a few numbers of UCs (1.3, 1.5, 3.5 and 3.6) are aiming to integrate with this Postgres database and utilize it for sharing the relevant network and application level data. If UCs application data is collected by measurement tools managed by the TaaS, the data are stored into the Postgres database in the same way as are the network data. Then, those data will be pulled by the VS data-collector.

As the process progresses, more detailed information about this integration will be available in future releases of D3.2.

5G-VINNI Patras

The VINNI Patras 5G testbed will focus on the validation of a series of KPIs, related to developed/deployed features and selected use cases. On the generic NFV front and with respect to available MANO features, validation will focus on the latency, throughput, service deployment time KPIs. Related to employed use cases, activities will also focus on reliability (service continuity), latency (interactivity) and throughput KPIs. Additional, qualitative/quantitative assessment activities will focus on validating resource and traffic isolation.

Integration with the visualization system will be performed in both the cloud infrastructure and the Radio Access Node using Open Source technologies. KPIs will be collected during the experiments and processed by the VS in order to be evaluated and visualised.

The following data streams can be available:

- Open Source MANO provides metrics data via a collector module that gathers data from VNF’s and NFVI’s. Prometheus TSDB stores metrics and exposes them via a UI and via a REST API. Monitoring data is also available from VCA for every VNF (if the VNF has already included the support)
- Cloud infrastructure resources are monitored by NETDATA and Prometheus. Metrics are exposed via a REST API
- Radio Access Node also provides metrics regarding 5G connection quality, UE’s connected to each Node, signal strength/quality, input/output traffic per UE, frame loss in Downlink direction, and CRC
errors on uplink direction. This information will be provided via Prometheus that will be provisioned as a network service. Metrics are exposed via a REST API. The monitoring network service contains VNF's for Prometheus and NETDATA and will have to be instantiated every time an experiment is to be conducted. The provisioning of the NS can be done either via OpenSlice, or the CDSO that communicates with OpenSlice.

- 5G core related monitoring data are not currently available and will be investigated in the future

5G Self-Contained Nodes

5G Self-Contained Nodes that deployed in several locations will include the RAN and 5G-core subset of the 5G infrastructure. Those self-contained nodes will provide 5G network coverage at remote locations and inside factories using the minimal required hardware in order to be versatile and portable. Networking connectivity to the VS infrastructure might be limited by physical constraints or company policies and this also has to be taken into account. In terms of metrics data provided and exposed to the VS, UE’s connected to each node, signal strength/quality, input/output traffic per UE, frame loss in Downlink direction, and CRC errors on uplink direction will be available and accessible via Web Sockets technology directly from the 5G self-contained Node.

4.5.3.2 CDSO Integration

The KPI Visualization System will be accepting Life Cycle Notifications from CDSO, through the public internet over HTTPS (of course the source/destination hosts will be whitelisted). Once a client is registered to the CDSO notifications (once-off), it receives notifications regarding various events of the use case lifecycle. The ones of interest here are the ones that indicate the start and completion of a test regarding a specific UC. Furthermore, a start test notification sent by the CDSO will trigger various collection procedures of the corresponding UC’s network and application KPIs. The “Network Service Deploy” notification, received by the visualisation system, will define the start of an experiment and the “Network Service Terminate” notification will define the completion of an experiment.

4.5.3.3 UC Integrations

The integration of the KPI Visualisation System with the existing use cases within the 5G SOLUTIONS is an ongoing and complex process due to the different integration processes required to be designed and implemented for each use case, enforced by the different nature of each use case and available integration flows on the respective applications. This paragraph provides a high-level overview of the integration with use cases that are mature enough to have provided this information up to now. A more detailed description of integration processes for all use cases will be provided in the drops of deliverable D3.2.

Living Lab 1

Living Lab 1 includes 5 use cases designed around the idea of Factories of the Future. Currently, the integration designs for two use cases has progressed well enough to be considered for description in this deliverable; more specifically, use cases “UC1.1 - Time-critical process optimization inside digital factories” and “UC1.2 - Non-time-critical Communication inside the Factory“.

For UC1.1, the application data will be pushed from application side to the KPI-VS, by using a REST API provided by the latter. The KPI VS will receive notifications from the use case for events related to the test start and stop. Latency is one of the KPIs that are going to be measured.
For UC1.2, the application data will be pushed from the application side by consuming a REST API provided by the KPI VS. Similar to UC1.1, the KPI VS will receive test start/stop notifications determining the execution time window of the test. Some of the KPIs that are going to be measured are latency, GW packet loss and jitter.

**Living Lab 2**

All the Living Lab 2 use cases will rely on the aforementioned 5G-EVE high level framework (see Figure 21) in terms of 5G network KPI, with particular attention to throughput (data rate), latency, coverage and reliability aspects.

The vertical KPIs representation will be in part of the 5G SOLUTIONS visualization tool that will be fed by the controllers developed by Ares2T (the UC owner) in each LL2 use case. As an example, making reference to the dataflow of the use case 2.1 (Industrial Demand Response), in Figure 23 the test case 2.1.3 dataflow on energy demand side management on non-residential buildings is shown:

![Figure 23: LL2 Use case 2.1, test 2.1.3 dataflow]

With reference to the above message sequence chart, Ares2T will develop an interfacing component within the “Remote Controller” in charge of measuring vertical KPIs and feeding the 5G-Solutions “KPI Visualization” tool with the data and vertical KPIs taken from the underlying use case. For the Test case 2.1.3 of UC2.1 the application data will be pushed from the application to KPI-VS via a REST-API, provided by the KPI Visualization system. For the start and stop of the experiment, the KPI-VS will receive notifications from the UC, indicating the time frame of the experiment performed. For this test case, one KPI that is going to be measured is “disconnection time”.
**Living Lab 3**

Living Lab 3 encompasses six use cases designed around the idea of Smart Cities & Ports. These use cases can be categorised in 3 groups:

- **Use case deployed in 5G VINNI site in Norway:** this concerns UC3.1 – Intelligent Street Lighting. Nurogames will provide the application for this use case and Telenor Norway will assist in its integration with the 5G VINNI site in Norway. For the KPI VS, network related data will be collected directly from the VINNI facility while the collection points for application data will be located in the cloud infrastructure used by Nurogames. Raw application data will also be processed in this cloud before it is presented visually in the KPI VS.

- **Use cases using 5G RAN node onsite, connected to 5G VINNI facility in Norway:** this concerns UC3.2 – Smart Parking, UC3.3 – Smart City Co-creation, UC3.5 – Autonomous Assets & Logistics for Smart Harbour/Port, and UC3.6 – Port Safety: Monitor and detect irregular sounds. For these UCs, the network related KPIs will also be collected from the VINNI facility in Norway where the 5G Core will be located. The application related KPIs will be collected, depending on the UC, from either dedicated cloud platforms connected to the 5G Core through a secure VPN link over the public internet where the application will be hosted and operated, or from some dedicated servers used specifically for the purpose of the UC and connected to the 5G Core over private links.

- **Use case Self-contained 5G node (Core and RAN):** This is specific to UC3.4 – Smart Buildings/Smart Campus deployed in IBM campus in Ireland. The deployment and integration of this UC will be realised in two main phases. The first phase involves a static orchestration of the network, with preconfigured network slices by University of Patras (UoP) in Greece. During this phase, the network KPIs will be provided by the 5G node deployed in IBM, subject to the availability of metrics that can be extracted from its APIs. In the second phase however, the orchestration of the 5G network slices will be handled by the Cross-Domain Service Orchestrator (CDSO) through the 5G VINNI facility in UoP. The network KPIs will then be extracted from both the 5G node in IBM and the 5G VINNI facility in Greece. The application level KPIs will mainly be extracted from IBM Cloud where the application will be hosted. Figure 24 and Figure 25 show the data flow diagram of UC3.4 during phase 1 and 2 of deployment respectively.
Figure 24: LL3 Use case 3.4 phase 1 of deployment dataflow
The UC3.4 application will monitor a set of metrics and generate KPIs that will be stored in a log file with a csv format on IBM Cloud. These log files will then be PUSHED periodically to the KPI VS through a dedicated REST API provided by the KPI VS Platform. The following is a sample data format of the csv logs file from UC3.4 application:

Table 5: UC3.4 metrics

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Timestamp</td>
<td>datetime</td>
</tr>
<tr>
<td>UE_ID</td>
<td>string</td>
</tr>
<tr>
<td>service availability</td>
<td>float</td>
</tr>
<tr>
<td>status</td>
<td>string</td>
</tr>
</tbody>
</table>
Each run of the application from each UC will have a unique identifier that will allow the KPI VS to differentiate it from other experiments and classify the KPIs based on LL number, UC number, run number in addition to the time of the day.

**Living Lab 4**

Living Lab 4 includes six use cases around the vertical of Media/Entertainment. Currently the integration with five use cases has progressed. These are “UC4.1: Ultra-High-Fidelity media”, “UC4.3: On-site Live Event Experience”, “UC4.4: User & Machine Generated Content”, “UC4.5: Immersive and Integrated Media and Gaming” and “UC4.6: Cooperative Media Production”.

For UC4.1, the application KPIs will be retrieved through the SNMP protocol. More specifically, the start and stop of the test case will be notified by the CDSO. Within this time-window, the KPI VS will poll the SNMP server with queries regarding the required KPIs based on provided MIBs.

The remaining three use cases (UC4.3, UC4.4 and UC4.6) do not allow the option for automatically retrieving data from the applications. Thus, an option will be provided by the KPI VS to manually upload log files that will later be digested by the data collector module. For each of these three use cases, a snapshot of each modem’s status is captured every 5 seconds. An initial group of metrics to be captured is depicted in Table 6:

<table>
<thead>
<tr>
<th>Meaning</th>
<th>Potential format</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Entry Date</td>
<td>2019-07-22</td>
<td>Not necessarily synchronized to a network or another accurate clock</td>
</tr>
<tr>
<td>Entry Time</td>
<td>14:29:20.900500</td>
<td>The time part of the timestamp</td>
</tr>
<tr>
<td>Modem Num</td>
<td>1-8</td>
<td>The number of the modem for which this data line is logged</td>
</tr>
<tr>
<td>Modem IP address</td>
<td></td>
<td>The private IP of the modem</td>
</tr>
</tbody>
</table>
| potentialBW        | 0-100000         | The snapshot value of what the application “believes” the current UL bandwidth can
### D3.1C KPI visualisation system specifications and design (v3.0)

<table>
<thead>
<tr>
<th>KPI</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>loss (%)</td>
<td>0-100</td>
<td>The snapshot value of what the application “believes” the current UL loss is. Units are %, integers</td>
</tr>
<tr>
<td>extrapolated smooth</td>
<td>0-100000</td>
<td>The snapshot value of what the application “believes” the current UL extrapolated smooth upstream delay is. Units are msec, integers</td>
</tr>
<tr>
<td>upstream delay</td>
<td></td>
<td></td>
</tr>
<tr>
<td>shortest round-trip delay</td>
<td>0-100000</td>
<td>The snapshot value of what the application “believes” the current UL shortest round-trip delay is. Units are msec, integers</td>
</tr>
<tr>
<td>extrapolated smooth</td>
<td>0-100000</td>
<td>The snapshot value of what the application “believes” the current UL extrapolated smooth round trip delay is. Units are msec, integers</td>
</tr>
<tr>
<td>round trip delay</td>
<td></td>
<td></td>
</tr>
<tr>
<td>minimum smooth round trip</td>
<td>0-100000</td>
<td>The snapshot value of what the application “believes” the current UL minimum smooth round trip delay is. Units are msec, integers</td>
</tr>
<tr>
<td>trip delay</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

For UC4.5, the application data will be pushed from application side to the KPI-VS, using a REST API provided by the latter. The KPI VS will receive notifications from the use case for events related to the test start and stop. Some of the KPIs that are going to be measured are: Time to connect to game, average ping to server, average ping from server, average input lag.

### 4.5.4 Deployment Architecture

#### 4.5.4.1 Overview

The Deployment Architecture pulls together the various development strands within a single deployment architecture that facilitates the verification and validation of all implementation changes and modifications made to the various codebase in a constant and reproducible manner.

The phrase Deployment Architecture (DA) will be used to cover all aspects of the deployment design and architecture needs of the 5G-SOLUTIONS KPI system. Designing a DA is an iterative process that continues for the length of the project. The first section will discuss why a good DA is required along with some of the tools and processes that can be used to achieve the goals of the overall DA.

The goals of the DA are:

1. Standardise the deployment process,
2. Execute standard and repeatable tests,

The purpose of the above goals is to engender confidence that the application has been successfully deployed and is fully ready for its intended use/purpose.

The Deployment Architecture can be broken into two sections:

1. Processes
It is worth noting that the technologies chosen are the enabler for the defined processes.

### Processes

**Deployment Process:**

Within the 5G-SOLUTIONS project some of the goals for the deployment process are:

- Reproducibility at all stages of the deployment process;
- Support for local deployments;
- Flexible enough to recover from errors;
- Resume stopped deployments;
- Clean up failed deployments;
- Be capable of verifying successful deployments;
- Adherence to open standards;
- Full status reporting for both passed and failed deployment processes.

The deployment process will involve multiple stages:

- Setup the hardware environment (e.g., OpenStack);
- Create the Virtual Machines (VMs) on the hardware environment including the Networking topology;
- Create the containerisation environment. (e.g. Docker & Kubernetes);
- Create the support environment. (e.g. Databases, Message Brokers, Storage, etc.);
- Deploy the individual services.

Stage ‘a’ is a prerequisite and is not intended to be implemented as part of the deployment process, instead it is intended that deployment will be to an existing environment.

Stages ‘b’ to ‘d’ will be optional depending on deployment target and in most cases, will only need to be carried out once per platform. What is important here is that the DA approach taken has the option to completely tear down all allocated resources and rebuild the support infrastructure in a consistent and automated manner.

Stage ‘e’ will be implemented for each service. As a service is modified the DA, upon detecting any changes made to that service, will trigger a build of that service (including any dependencies) and upon success shall deploy to the infrastructure.

**Build Process:**

The following is an explanation of the methodology required in order to achieve the above goals. Figure 26 describes an example sequence diagram of the steps used to submit and verify code changes and the stages that might be followed throughout the lifespan of a single change.

When a change is made to the sources of a managed DA service (code change, build info change, configuration file change, etc.), the developer commits that change to the Version Control System (VCS) that’s in use, such as Git. The build machine for the project is notified of this change automatically and checks out the code for the updated service. A Docker image is created and is used as the environment to execute the unit tests for the service. If all the tests pass, the Docker image is placed in a Docker repository and tagged as having passed the unit test.

---

4 [https://www.openstack.org/software/](https://www.openstack.org/software/)
The next phase automatically carries out integration/acceptance testing, where the Docker image is tested with the rest of the system. Once it passes test, the image is retagged as having passed the integration/acceptance tests.

Following the successful integration/acceptance tests, the next step involves reviewing the reports generated by previous steps and then carrying out user acceptance tests on the service in question. Once this step is passed successfully, the image is tagged as being production ready.

The production ready service is now taken by the DevOps or Sys Admin and staged for deployment.

### 4.5.4.3 Deployment Technologies

Where possible the use of open-source tools and technologies is desirable, as they are vendor neutral and widely supported across multiple cloud vendors. These tools and technologies also tend to be community driven, with bugs and security issues addressed more responsively without the need for expensive recurring licenses.
Continuous Integration / Continuous Deployment (CI / CD)

Continuous Integration/Continuous Deployment (CI/CD) is a design practice used in software development to automate and improve the process of software delivery. To make full use of automation for the development and verification of software artifacts, the employed version control systems should have the capability of pipelining where the build process can be designed and automated and is triggered by a new check in of the software being managed. The more a software component is checked, built and verified by the build system then the sooner that the software component can be deployed to the live environments. Figure 27 displays the main stages involved in each pipeline of the CI/CD process.

![Figure 27: CI / CD process](image)

From a DevOps perspective, techniques such as automated testing, continuous integration and continuous delivery allow software to be developed to a high standard and can be easily packaged and deployed to test environments thereby resulting in the ability to rapidly, reliably and repeatedly push out enhancements and bug fixes to customers and stakeholders at low risk and with minimal manual overhead. As the processes are automated, the tasks of execution of the pipelines are completely managed by the underlying versioning control system.

Containerisation

![Figure 28: Differences between virtual machines and containers](image)
Containerisation involves bundling a service along with all the dependencies and configuration files into a single package that is then run in a configurable sandboxed environment known as a container. Containers can be allowed open paths into the sandbox to allow, for example, TCP/IP communication between them (for use of protocols such as HTTP RESTful services and SOAP). On the appropriate environment, this can occur seamlessly even if the containers are deployed onto separate hardware. This is a widely supported technology and allows services to be deployed across a wide range of operating environments and cloud vendors.

Both Docker and Kubernetes are two of the leading tools used for containerisation. While Docker is a computer application that uses the concept of containerisation, Kubernetes is a container orchestration system. Docker makes it easier to create, deploy and run applications with the use of containers. Kubernetes is used for automating deployment, scaling and management of the containers. Kubernetes solves one of the biggest problems with containerisation, how all the containers should be coordinated and scheduled.

Cloud Deployment

The KPI Visualisation System for the 5G-SOLUTIONS project has to be made available for the entire set of project partners, especially those involved in the execution and verification of the living labs use cases test runs. As the KPI Visualisation System is a distributed system that aggregates data from several sources and makes the results available to the set of stakeholders, it is important that the deployment of the platform is consistent with the needs of those actors (stakeholders, living labs, etc.). As discussed earlier, the best means of deployment for the visualisation system is for it to be hosted on a cloud platform where all of the user requirements can be met, while also providing the functional requirements such as scalability, availability, reliability, storage, etc.

Initially no decision was made regarding which public cloud service should be used to host the 5G-SOLUTIONS KPI Visualisation System, so to aid in the decision-making process an investigation was carried out amongst the most popular cloud vendors to investigate their available features and associated costs. Table 7 lists the most popular cloud services along with some of the features most relevant to the 5G-SOLUTIONS Visualisation System. The two main conclusions to be drawn from this table are:

- The deployment support across each platform is unique
- The costs across each platform are roughly similar

<table>
<thead>
<tr>
<th>Vendor</th>
<th>Build, Test &amp; Deploy Automation</th>
<th>Local Deployment</th>
<th>Command Line interface</th>
<th>Sample Compute Cost/ Month&lt;sup&gt;Table 8&lt;/sup&gt;</th>
<th>Sample Storage Cost / Month&lt;sup&gt;Table 8&lt;/sup&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>aws</td>
<td>AWS CloudBuild</td>
<td>Limited functionality available</td>
<td>AWS Command Line Interface (CLI)</td>
<td>$954&lt;sup&gt;Figure 29&lt;/sup&gt;</td>
<td>$23</td>
</tr>
<tr>
<td></td>
<td>AWS CodeDeploy</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>azure</td>
<td>Azure Pipelines</td>
<td></td>
<td>Azure CLI</td>
<td>$936&lt;sup&gt;Figure 30&lt;/sup&gt;</td>
<td>$23</td>
</tr>
<tr>
<td>google</td>
<td>Cloud Build</td>
<td></td>
<td>Gcloud CLI</td>
<td>$1041&lt;sup&gt;Figure 31&lt;/sup&gt;</td>
<td>$21</td>
</tr>
</tbody>
</table>

© 5G Solutions, 2020
Table 8 captures the template for the resources used to calculate the sample prices for each cloud service assuming a storage option of 1TB. This table is not indicative of the resources required to operate the 5G-SOLUTIONS Visualisation System, but it is an example of the average cloud cost per month for developing a small to medium sized cloud service.

<table>
<thead>
<tr>
<th>VM Type</th>
<th>RAM</th>
<th>Quantity</th>
<th>Usage</th>
<th>Storage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Small 1 – 2 vCPU</td>
<td>1-2GB</td>
<td>4</td>
<td>100%</td>
<td>1TB</td>
</tr>
<tr>
<td>Medium 2 – 4 vCPU</td>
<td>2-4GB</td>
<td>4</td>
<td>100%</td>
<td>1TB</td>
</tr>
<tr>
<td>High Compute / GPU</td>
<td>15-30GB</td>
<td>2</td>
<td>30%</td>
<td>1TB</td>
</tr>
</tbody>
</table>

Based on the template described in Table 8, the costs associated for each of the cloud vendors are presented in Figure 29, Figure 30, Figure 31 and Figure 32
### 4.5.4.4 Deployment Strategy

During the early stage of the project, it can be difficult to determine which technologies and vendors are the most appropriate to utilise for the successful delivery of the projects’ objectives. As the project progresses, certain criteria (requirements, capabilities, etc.) become clearer and over time the selection of technologies and vendors to be used is made less complex. In the case of the deployment architecture, a certain amount of investigation and proof of concepts needed to be executed prior to any final decision being made. With this in
mind, the following decisions have been made regarding the deployment strategy and the delivery of the initial demonstrator that displays the capabilities of the deployment system.

**OpenStack**

OpenStack is an open source technology that allows you to build your own cloud infrastructure. OpenStack has a very comprehensive API that allows you to configure the stack into whatever configurations you require. OpenStack leverages support from several cloud vendors and also has the ability to be deployed in a private installation make it an ideal technology base for a vendor neutral deployment platform.

It has been decided that WIT shall make its OpenStack deployment available for use by the KPI Visualisation System. The OpenStack platform has similar features available as most of the mainstream cloud vendors including availability of VM provisioning, Storage and Networking (both internal and public facing IPs). As WIT run and manage their own data centre on campus this simplifies the cloud selection decision at least until the needs of the project exceed the capabilities of the OpenStack platform. Use of the WIT OpenStack platform is more than sufficient for the initial phases of the 5G-Solutions project.

**Terraform**

Terraform is an open-source infrastructure as code software tool created by HashiCorp. It enables users to define and provision infrastructure within a datacentre using a high-level configuration language known as Hashicorp Configuration Language, or optionally JSON. Terraform enables you to safely and predictably create, change, teardown and improve the deployed infrastructure. Terraform has the capabilities to not only deploy VMs in the datacentre but also build the entire underlying platform required to support your application needs. This includes the creation of specific VM images & flavours, creation of specific networks, subnets & public interfaces while also managing the lifecycles of the VMs themselves. Terraform is virtually platform independent as it supports all of the major cloud vendors and the lesser known ones. Terraform also supports OpenStack so the decision has been made to utilise Terraform to manage the infrastructure for the deployment architecture.

**Docker and Kubernetes**

The architecture being pursued for the KPI Visualisation System is based on a micro services architecture and the use of containers is perfectly aligned with this design paradigm. Containers are a way of packaging software and provide portability (can be moved to another machine or cloud vendor). All of the code for the application, including libraries and dependencies is packed together in the container as an immutable artifact. From the usability perspective, containers help facilitate quick elasticity and separation of concerns. With this in mind, each of the individual KPI VS components can be self-contained within its own container. The use of Docker helps in the design, creation and deployment of containers that can run anywhere.

Running all of the KPI VS components as containers supports the micro services approach but there are still some issues that need to be addressed. Containers may need to be started in a particular sequence, deployed in different locations, communications between containers configured, as well as configurations for the general management of failed containers or hardware. To deal with this aspect Kubernetes is being used as the

---

5 [https://www.terraform.io/intro/index.html](https://www.terraform.io/intro/index.html)
6 [https://www.terraform.io/docs/providers/index.html](https://www.terraform.io/docs/providers/index.html)
7 [https://martinfowler.com/articles/microservices.html](https://martinfowler.com/articles/microservices.html)
8 [https://www.docker.com/](https://www.docker.com/)
9 [https://kubernetes.io/](https://kubernetes.io/)
container orchestration platform. Kubernetes is an open-source system for automating deployment, scaling, and management of containerised applications. Kubernetes can be used to run containers across different machines, scale up/down components by adding/removing containers, load balancing between containers and monitoring hardware.

**GitLab**

GitLab has been chosen as the version control system and CI/CD pipelining framework for the development and deployment of the components for the 5G-Solutions project WP3 outputs. The use of GitLab provides all of the features that support not only the versioning requirements but also the capabilities for the design and implementation to achieve the build out of the CI & CD pipelines. As a component developer (for one of the WP3 components) the complexities of building and testing of their components should be automated as much as possible while the deployment to the Kubernetes cluster should be abstracted away from the users. GitLab also provide bug tracking and feature requests workflows where all tickets can be associated to both components and developers to allow for issues to be addresses and resolved accordingly.

GitLab also features the ability to design and deploy a set of Runners that provide instances tailored for each component used to execute the build and deployment pipelines. This results is a completely decoupled versioning and build system that is compatible with any type of development language (code or scripted) and can be easily integrated into the CI/CD pipelines. This reduces on-boarding time for developers, quicker code to deployment times and provides a single consistent interface and workflow for all developers.

GitLab also offers its Container Registry for use by projects. The registry is a local repository for the storage of containers that can be used for not only the deployment of the actual KPI-VS components, but also for the storage of the containers consumed by the GitLab runner instances used to build and test the KPI-VS components themselves.

All of these features are available in the GitLab framework and they are all freely available for the use of the project for the development of all of the KPI-VS components.

### 4.6 Design Principles and Technologies Used

#### 4.6.1 GUI Design Principles

The user interface (UI) is a critical part of any software product. It is not usually noticed when it is done well, but when it’s done poorly, users cannot get past it to efficiently use a product. Interface design principles represent high-level concepts that are used to guide software design. To increase the chances of success when creating user interfaces, interface design principles were followed.

The design and implementation of the portal will follow the user interface design principles with focus on how to satisfy various types of users by providing a smooth, easy and nice user experience. The following list highlights the Graphical User Interface (GUI) Design Principles\(^\text{10}\) considered in the KPI Visualisation System’s interface design and implementation.

- **Aesthetically pleasing**: Provide visual appeal by following presentation and graphic design principles, as below:
  - Provide meaningful contrast between screen elements.
  - Create groupings.
  - Align screen elements and groups.

---

o Provide three-dimensional representation when this is necessary.
  o Use colors and graphics effectively and simply.

• **KISS (Keep It Simple and Stupid):** Users will feel most comfortable if they can use your product without putting much effort into interactions. Due to the differences in user’s habits and abilities, keeping the design simple is the best way to help users with various abilities to understand the UI. One of the best practices is to allow users to rely on their previous experience when they interact with it and this can be achieved by using familiar concepts and metaphors.

• **Clarity:** The interface is visually, conceptually and linguistically clear.

• **Comprehensibility:** The interface is easily understood, friendly and the flow can be easily followed.

• **Consistency:** The interface looks and operates in the same way throughout the portal. This means:
  o Have a similar look, and uses
  o Operate similarly
  o The same action should always yield the same result
  o The function of the elements should not change
  o The position of standard elements should not change

• **Control:** The user controls the interaction and is never interrupted for errors.

• **Efficiency:** Minimize user’s eye and hand movements. The transition between the various portal control flows are easily and freely. Also, the navigation paths are as short as possible and ensure that users never lose their work.

• **Simplicity:** Provide an interface as simple as possible and make common actions simple at the expense of uncommon actions being made harder. In addition, provide uniformity and consistency.

• **Recovery:** Immediate return to a certain point if difficulties arise and ensure that users never lose their work as a result of an error or communication problems.

The user interface is the most critical part that designers need to pay close attention to. For that reason, for all of the above GUI design principles, designers took user’s perspective into account to make users understand the graphic UI design better.

In order to achieve the cross browser and multi device compatibility, during the implementation of the KPI Visualisation System, we will utilize the latest technologies of HTML5, CSS3 and JavaScript, enabling the user to access the content of the KPI Visualisation System from anywhere. The above-mentioned technologies are the latest evolution, offering a state-of-the-art user interface with new elements, attributes, and behaviors as highlighted in the list below.

• **Offline storage:** Allow web-pages to store data locally and operate offline more efficiently.

• **Multimedia:** Usability of plug-ins allowing the handling of different multimedia content, such as videos.

• **2D/3D graphics and effects:** Allow the support of a wider range of presentation options.

• **Performance and integration:** Provide greater speed optimization, use and take advantage of the computer hardware.

• **Device access:** Allow the use and accessing from various type of devices.

• **Styling:** Allow authors to write more sophisticated themes.

### 4.6.2 Operating System and Programming Languages

Since the entire KPI Visualisation System and its integration will be based on open source standards, the operating system chosen to base the entire infrastructure on will be a UNIX operating system (CentOS, Debian, Ubuntu server, etc.). Amongst others, Linux OS requires no license for use, is reliable and needs far less hardware resources than any Windows-based OS.

The KPI Visualisation System will be built using a combination of programming languages. Based on the location the code is running, it can be separated into two categories, the client-side languages and the server-side ones.
1. **Client-Side languages:** code here is executed on browsers. Due to the low processing power available in the client-side, this code is intended to retrieve the information from the backend system and formalize the data into a user-friendly display manner. Technologies widely used in this case are:
   - HTML5 (HyperText Markup Language),
   - CSS3 (Cascading Style Sheets),
   - JavaScript,
   - AJAX (Asynchronous JavaScript and XML)

Due to their popularity, many frameworks and libraries have been designed based on the above-mentioned technologies. The KPI Visualisation System will be based on the AngularJS framework, combined with the PrimeNG UI components to design a nice, user friendly and practical graphical user interface.

2. **Server-Side languages:** code executed on the server side. Tasks such as 3rd party integrations, database integrations, data manipulation and generally tasks that require large processing power are run in the backend. Within the context of the 5G-SOLUTIONS project, the server-side coding is done in Java, Python and .NET.

The above programming languages are the ones currently used for the implementation of the KPI Visualisation System. However, the idea of using further languages is still on the table if proven to deliver better results.

### 4.6.3 Other Applications and External Libraries

A non-exhaustive list of applications, systems, tools and libraries considered for usage within the context of the KPI Visualisation System are:

- **Application Servers:**
  - **Spring boot with Tomcat:** The Spring Framework is an application framework and inversion of control container for the Java platform. The framework's core features can be used by any Java application, but there are extensions for building web applications on top of the Java EE (Enterprise Edition) platform. Spring Boot is Spring's convention-over-configuration solution for creating stand-alone, production-grade Spring-based Applications that you can "just run". It is preconfigured with the Spring team's recommended configuration and use of the Spring platform and third-party libraries to allow for getting started with minimum fuss. Spring Boot embeds Tomcat or Jetty directly, thus eliminating the need to deploy WAR files.

- **Databases:**
  - **MongoDB**\(^{11}\): MongoDB is an open-source general purpose, document-based, distributed database written in C++. MongoDB stores data in JSON-like documents offering a more expressive and powerful way of thinking about data than the traditional row / column model. MongoDB provides high availability with replica sets and scales horizontally using sharding. Support for multi-document ACID transactions was added to MongoDB with the General Availability of the 4.0 release in June 2018.
  - **MariaDB**\(^{12}\): MariaDB is a community-developed, commercially supported fork of the MySQL relational database management system (RDBMS). MariaDB is free and open-source software under the GNU General Public License. This database will be used in cases where the structured data are of interest in comparison to the document-based data provided by MongoDB.

- **Processing/Integration frameworks:**

---

\(^{11}\) https://www.mongodb.com

\(^{12}\) https://mariadb.com
Apache Spark\textsuperscript{13}: Apache Spark is a unified analytics engine for large-scale data processing that provides an interface for programming entire clusters with implicit data parallelism and fault tolerance.

Apache Camel\textsuperscript{14}: Apache Camel is an open source integration framework for message-oriented middleware with a rule-based routing and mediation engine that provides a Java object-based implementation of the Enterprise Integration Patterns using an application programming interface to configure routing and mediation rules.

Microsoft .NET Core\textsuperscript{15}: a framework utilized for the application development. .NET Core is a new version of .NET Framework, which is a free, open-source, general-purpose development platform maintained by Microsoft. It is a modular, lightweight, fast, and cross-platform framework that runs on Windows, macOS, and Linux operating systems and can be used to build different types of applications such as mobile, desktop, web, cloud, IoT, machine learning, micro-services, etc. It includes the core features that are required to run a basic .NET Core app. The main limitation with the .NET Framework is that it only runs on the Windows platform and needs to use different .NET APIs for different Windows devices such as Windows Desktop, Windows Store, Windows Phone, and Web applications. As mentioned previously, .NET Core is an open-source framework maintained by Microsoft and available on GitHub a number of GitHub repositories (i.e. .NET Core runtime\textsuperscript{16}, .NET Core SDK\textsuperscript{17} and ASP.NET Core\textsuperscript{18} repositories) under MIT\textsuperscript{19} and Apache 2\textsuperscript{20} licenses. .NET Framework supports multiple programming languages to develop .NET Core applications. The one it is used for the 5G-SOLUTIONS project is C#\textsuperscript{21}. You can use your favorite IDE, including Visual Studio 2017/2019, Visual Studio Code, Sublime Text, Vim, etc. .NET Core supports modular architecture approach using NuGet packages\textsuperscript{22}. There are different NuGet packages for various features that can be added to the .NET Core project as needed. Even the .NET Core library is provided as a NuGet package. In this way, the .NET Core application speed up the performance, reduce the memory footprint and becomes easy to maintain. The NuGet package for the default .NET Core application model, and the one that it is utilized in the project is Microsoft.NETCore.App\textsuperscript{23}. .NET Core application has a flexible deployment (deployed via user-wide or system-wide or with Docker Containers) and the one chosen to be utilized for the project is Docker Containers Deployment\textsuperscript{24}. The .NET Core Framework composed of the following parts that are currently utilized:

- CLI Tools\textsuperscript{24}: The .NET Core command-line interface (CLI) is a cross-platform toolchain for developing, building, running, and publishing .NET Core applications.
- Roslyn\textsuperscript{25}: Language compiler platform for C#
- CoreFX\textsuperscript{26}: Set of framework libraries.

\textsuperscript{13} https://spark.apache.org
\textsuperscript{14} https://camel.apache.org
\textsuperscript{15} https://github.com/dotnet/core
\textsuperscript{16} https://github.com/dotnet/runtime
\textsuperscript{17} https://github.com/dotnet/sdk
\textsuperscript{18} https://github.com/dotnet/aspnetcore
\textsuperscript{19} https://github.com/dotnet/runtime/blob/master/LICENSE.TXT
\textsuperscript{20} https://www.apache.org/licenses/LICENSE-2.0
\textsuperscript{21} https://www.nuget.org/packages
\textsuperscript{22} https://www.nuget.org/packages/Microsoft.NETCore.App
\textsuperscript{23} https://docs.microsoft.com/en-us/dotnet/core/docker/introduction
\textsuperscript{24} https://github.com/dotnet/cli
\textsuperscript{25} https://github.com/dotnet/roslyn
\textsuperscript{26} https://github.com/dotnet/corefx
• CoreCLR\(^{27}\): is the .NET execution engine, performing functions such as garbage collection and compilation to machine code.

• Machine Learning:
  o IBM Watson\(^{28}\): IBM Watson is an open, multicloud platform that lets you automate the AI lifecycle and build powerful models from scratch, or speed time-to-value with pre-built enterprise apps.
  o TensorFlow & Keras: TensorFlow is a free and open-source software library for dataflow and differentiable programming across a range of tasks. It is a symbolic math library, and is widely used for machine learning applications such as neural networks. Keras is TensorFlow’s high-level API for building and training deep learning models. It’s used for fast prototyping, state-of-the-art research, and production, with three key advantages: user-friendly, modular, and easy to extend.
  o Pytorch: is an open source machine learning library based on the Torch library. Similar to TensorFlow it is primarily used for implementation and training of neural networks. The main difference from TensorFlow is that PyTorch creates a dynamic graph which allows the model to be modified and amended as the model evolves with data. This is particularly helpful while using variable length inputs in RNNs which is the common model to use on sequential and time series data. The community support for this library has dramatically increased over the recent years with many open source implementations and tutorials now being available online.
  o Scikit-learn: is a free machine learning library for Python which features a large suit of classical machine learning algorithms in classification, regression and clustering e.g. random forests, kNN, k-means, etc. It is designed to interoperate with Python numerical and scientific libraries such as NumPy, SciPy and Pandas.
  o Numpy: is a Python library that provides a multidimensional array object, including mathematical, logical, matrix manipulation, sorting, I/O, discrete Fourier transforms, basic linear algebra, basic statistical operations, random simulation and much more.
  o Pandas: is an open source library for python programming language for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables and time series.
  o DataRobot: Data Robot is an ML lifecycle tool that automates the end-to-end process for building, deploying, and maintaining AI at scale, including in the cloud, on-premise, or as a fully-managed AI service. It includes automation for data modelling, feature selection, ways to try out a variety of algorithms to understand which ones work better on the current data and problem. It also offers flexible deployment capabilities. A full description of DataRobot capabilities is included in deliverable D2.3A [4].

• Version Control Systems:
  o Git\(^{29}\): Git is a distributed version-control system for tracking changes in source code during software development. It allows for coordinating work among programmers, but it can also be used to track changes in any set of files

• Authentication framework:
  o JSON Web Token\(^{30}\): JSON Web Token (JWT) is an open standard (RFC 7519) that defines a compact and efficient way for transmitting information between two parties in a secure manner. The data are transmitted as a JSON Object. JWT can be signed either with HMAC algorithms (using a secret key) or RSA / ECDSA (using a pair of public/private key). The JSON Web Token consists of three components, separated by dots. These three components are:
    ▪ Header (Usually contains the type of token, as well as the signing algorithm).

\(^{27}\) https://github.com/dotnet/coreclr
\(^{28}\) https://www.ibm.com/watson
\(^{29}\) https://git-scm.com
\(^{30}\) https://jwt.io/
Payload (Contains the claims, which are statements about an entity, and additional data).

Signature (This part is used to verify the message. The encoded header and payload, the secret and the algorithm specified in header are used to create the signature).

- Containers:
  - **Docker**[^1]: Docker is a tool which is used for containerizing applications. It is a tool designed to make easier and efficient the creation, deployment and execution of applications by using containers.

The KPIs will be represented in both tabular and graphical form. There are numerous JavaScript libraries, offering a large number of presentations, that can be used for the graphical representation. Some indicative ones are Highcharts[^2], D3[^3], FusionCharts[^4], MetricsGraphics[^5] and ZingChart[^6]. Among others, the aforementioned libraries offer charts such as:

- Column & Bar charts,
- Line & Area charts,
- Stacked charts,
- Pie & Donut charts,
- Bubble & Scatter charts,
- Linear scale,
- Speedometer,
- Heat maps.

Considering all the use cases designed up to now, it seems that Highcharts covers all the requirements of the KPI VS.

### 4.7 Implementation Plan

In general, the 5G-SOLUTIONS roadmap follows the 3GPP, the IMT-2020 and the 5G-PPP implementation roadmaps according to the evolution and upgrades of the 5G-EVE and 5G-VINNI ICT-17 facilities. This is achieved through the planning of 3 iterative and consequential testing cycles in Phase 3 (as presented in the following Figure 33). So, the execution of all Living Labs will be performed over 3 iterative cycles, each lasting for 6 months, followed by an evaluation period of 2 months. This will ensure a smooth and aligned evolution of the vertical industry validations in the Living Labs, and also ensures that the appropriate testing and validation is conducted in a comprehensive manner per 3GPP release.

[^1]: https://www.docker.com/
[^2]: https://www.highcharts.com
[^3]: https://d3js.org
[^4]: https://www.fusioncharts.com/
[^5]: https://metricsgraphicsjs.org
[^6]: https://www.zingchart.com
As a consequence, the implementation of the Visualisation System is planned based on the above-mentioned concept of having 3 iterative testing cycles. Apart from the testing periods, the implementation of the Visualisation System has also been identified in an agile-based process as follows:

- **Requirements collection and Designs (M3-5)**
  - Collect the requirements.
  - Have a general understanding of the volume, type of data, etc.
  - Understand who the audience is, the users and their privileges.

- **Create Wireframes and Mock-up Designs (M6-8)**
  - Prepare mock-ups and set up the basic page structure.
  - Take into account the user experience.
  - For each element that gets added to the interface, the user flow must be taken into consideration as well.
  - Prepare designs (on Photoshop or any other UI design tool) to demonstrate how the UI will actually look.
  - Get the approval from the users that everything looks fine.

- **1\textsuperscript{st} Implementation Round – Prototype Version (M9-12)**
  - Initiate the development, based on the previously approved designs.
  - Provides a first release (prototype) to the users.

- **2\textsuperscript{nd} Implementation Round – Intermediate Version (M13-18)**
  - Working on the parallel users’ feedback from the 1\textsuperscript{st} testing period.
  - Provides an intermediate release to the users.

- **3\textsuperscript{rd} Implementation Round - Final Version (M19-26)**
  - Working on the parallel users’ feedback from the 2\textsuperscript{nd} testing period.
  - Provides the final release to the users.

- **4\textsuperscript{th} Implementation Round – Adjustments (M27-34)**
  - Working on the final adjustments based on the final users’ feedback from the 3\textsuperscript{rd} testing period (executed in parallel).
  - The revised final version will be provided.

Currently, as we are now in Cycle1, we are delivering the intermediate version. We have already deployed the KPI VS in the cloud infrastructure and we are on-boarding UCs as they become available both in a matter of integration points and in a matter of users to view data and provide further feedback.
5 Conclusions and Next Actions

This deliverable tries to identify the requirements and lay out the design of the KPI Visualisation System along with the methodologies and technologies utilized for its design and implementation. Due to the fact this deliverable had three drops, it was considered a living document that was constantly updated as the design of the KPI VS progressed and new requirements came in from the use cases as they matured spanning from M3 when the WP3 started and ending on M17. Thus, the interim deliverables (A and B) include information of the requirements and the design up to the writing time. This last deliverable (C) is considered the final, complete version and supersedes all previous versions.

A data and user flow diagram is presented to indicate the integration flows and the user interactions within the KPI VS portal in a UC oriented manner. An indicative sample of screenshots and data filters have been provided to indicate the look & feel along with the various filtering needs. This flow is expected to slightly deviate per use case depending on their requirements. As this information is not currently available for all use cases, deviations will be decided and implemented during the UC on-boarding process.

The user roles within the visualisation system have been identified. Currently five different user roles have been identified, each one serving a purpose and allowing efficient user management on a per LL or UC level. Each role has been assigned specific rights within the platform allowing the isolation of use case data from unauthorized users. As the project is still half-way, the system has been designed in a modular way allowing for more roles and user actions to be added in the future if necessary.

The core architecture of the system has been designed and presented. This includes a high-level architecture of the KPI Visualisation System and a lower level modular architecture that highlights the various modules existing within the KPI VS along with the integration points necessary for the visualisation system to align with the use case applications, the underlying facilities and the Cross-Domain Service Orchestrator (CDSO). The integration points identified up to now include three 5G facilities (VINNI-Patras, VINNI-Norway, EVE), a series of self-contained nodes, the CDSO and a total of twenty use case applications. Due to the large number of external systems, some of which are commercial – thus difficult to adapt, different integration flows and protocols have been designed and are currently being implemented within the scope of the KPI VS. The deployment architecture has also been defined and already implemented on a private cloud including the deployment flows. A transition to a public cloud will be evaluated further on the project depending on the final requirements.

The selection of technologies used in the project is of significant importance. The large amount of data, the resource scaling and the security levels required enforce the use of state-of-the-art technologies to optimize performance, scaling and security. Having all the above in mind, along with the multiple integration points that require a certain level of flexibility, the appropriate technologies have been selected and documented.

The core system design is nearly complete. What is still missing at some level is the integration points with some of the use cases and the underlying 5G facilities. Currently, Cycle 1 of the use case tests is running and use cases are gradually on boarding the KPI VS. The remaining use cases are also expected to start the on-boarding process until Cycle 2. Integration with the facilities is not yet complete as the integration points were not defined until recently for 5G-VINNI and there are still ongoing discussions with 5G EVE to provide their integration points. Of course, necessary adaptations are expected to be needed once the integration points are added but the general design will not change. As this is the last drop of the D3.1 series of deliverables, any changes in the design will be updated in the next series of deliverables within this WP, namely 3.2, spanning until M30, that cover the implementation of the KPI VS and the integration process.
6 References

[1] 5G-SOLUTIONS Description of Actions (DoA).

[2] D1.1A “Definition and analysis of the use cases / scenarios and corresponding KPIs based on LLs” v.1.0.
