FORESIGHT

ADVANCED CYBER-SECURITY SIMULATION PLATFORM FOR PREPAREDNESS TRAINING IN AVIATION, NAVAL AND POWER-GRID ENVIRONMENTS

Grant Agreement: 833673

D9.1

Gamification and visualisation modules (I)

This project has received funding from the European Union's Horizon 2020 Research and Innovation Programme under Grant Agreement No 833673.

Document information

Deliverable number:	**D9.1**
Deliverable title:	Gamification and visualisation modules (I)
Deliverable version:	V1.0
Work Package number:	WP9
Work Package title:	FORESIGHT toolkit development
Due Date of delivery:	31/03/2021
Actual date of delivery:	22/04/2021
Dissemination level:	Public (PU)
Type	Other
Editor(s):	Nicholas Kolokotronis, George Lepouras, Nikos Platis (UOP)
Contributor(s):	Nicholas Kolokotronis, George Lepouras, Nikos Platis, Panagiotis-Iasonas Diakoumakos, Michalis Tsoukalos, Costas Vassilakis, Evangelos Chaskos (UOP) Peng Zhao, Stavros Shiaeles (UOPHEC) Vasiliki-Georgia Bilali, Eleni Darra (KEMEA)
Reviewer(s):	Dimitrios Kavallieros (KEMEA) Christos Iliou (CERTH)
Project name:	Advanced cyber-security simulation platform for preparedness training in Aviation, Naval and Power-grid environments
Project Acronym	FORESIGHT
Project starting date:	1/10/2019
Project duration:	42 months
Rights:	FORESIGHT Consortium

Document history

Version	Date	Beneficiary	Description
0.1	01.10.2020	UOP	Tentative table of contents has been prepared
0.3	18.02.2021	UOP	Information about the architecture and design/state-of-the-art of the modules has been added
0.5	15.03.2021	UOP	Information about the requirements/functionalities covered and interfaces of the modules has been added
0.6	31.03.2021	UOP	Unit testing information has been added
0.7	12.04.2021	UOP	Complete version, with data messages obtained from other modules, and also introduction/conclusions
0.8	13.04.2021	UOP	Deliverable has been sent for review
0.9	15.04.2021	KEMEA, CERTH	Review comments and quality review
1.0	16.04.2021	UOP	Final version for submission

Acknowledgement: This project has received funding from the European Union's Horizon 2020 Research and Innovation Programme under Grant Agreement No 833673.

Executive summary

Amongst the primary goals of WP9 is to increase trainees' engagement so as to maximise the training outcomes of security professionals at all levels and also implement advanced visualisations to explore information about simulation and training. This report presents the first version of the Gamification (GAME) and the Cyber Security Visualisation (CSV) modules that offer users the ability to be engaged in the training process to interact with the FORESIGHT platform and with each other.

The deliverable describes a high-level overview of the modules and how they have proceeded to satisfy the requirements and expected functionalities. State-of-the-art aspects are also included in the report along with the design details for each module (regarding the technologies used, the database (DB) schema, etc.). The deliverable also gives a detailed description of the modules' architecture, as well as, the Application Programming Interfaces (APIs) that are being developed to provide the envisaged functionalities and interact with other FORESIGHT modules (e.g. to obtain information about the available scenarios, or to get the evaluation results of the trainees in various tasks). The deliverable also describes the approach taken for testing, along with a number of test cases, to verify that the proper operation of the GAME and CSV software modules.

The deliverable provides an analysis of the tools and methods used in the process of developing the GAME and CSV modules of FORESIGHT; it is therefore technical by nature. We believe that readers with a technical background will find the presentation comprehensive and the analysis accurate and complete. Non-technical readers might have to skip more technical parts, especially during the first reading of the document.

Table of contents

List of figures

List of tables

Acronyms and abbreviations

Term	Description
API	Application programming interface
CR	Cyber range
CSV	Cyber-security visualisation module
CTC	Cyber-team collaboration module
DB	Database
DBMS	Database management system
DSG	Dynamic scenario generation module
EXP	Experience points
GAME	Gamification module
HCI	Human computer interaction
IC	Innovative curricula
LMS	Learning management system
RDBMS	Relational DBMS
TE	Training evaluation module
UI	User interface
URI	Uniform Resource Identifier

1 Introduction

1.1 Overview

The FORESIGHT project (see Figure 1) aims to develop a federated cyber-range solution to enhance the preparedness of cyber-security professionals at all levels and considerably advance their skills towards preventing, detecting, reacting and mitigating sophisticated cyber-attacks. This deliverable presents the work carried out in the context of the development for the Gamification (GAME) module and the Cyber Security Visualisation (CSV) module.

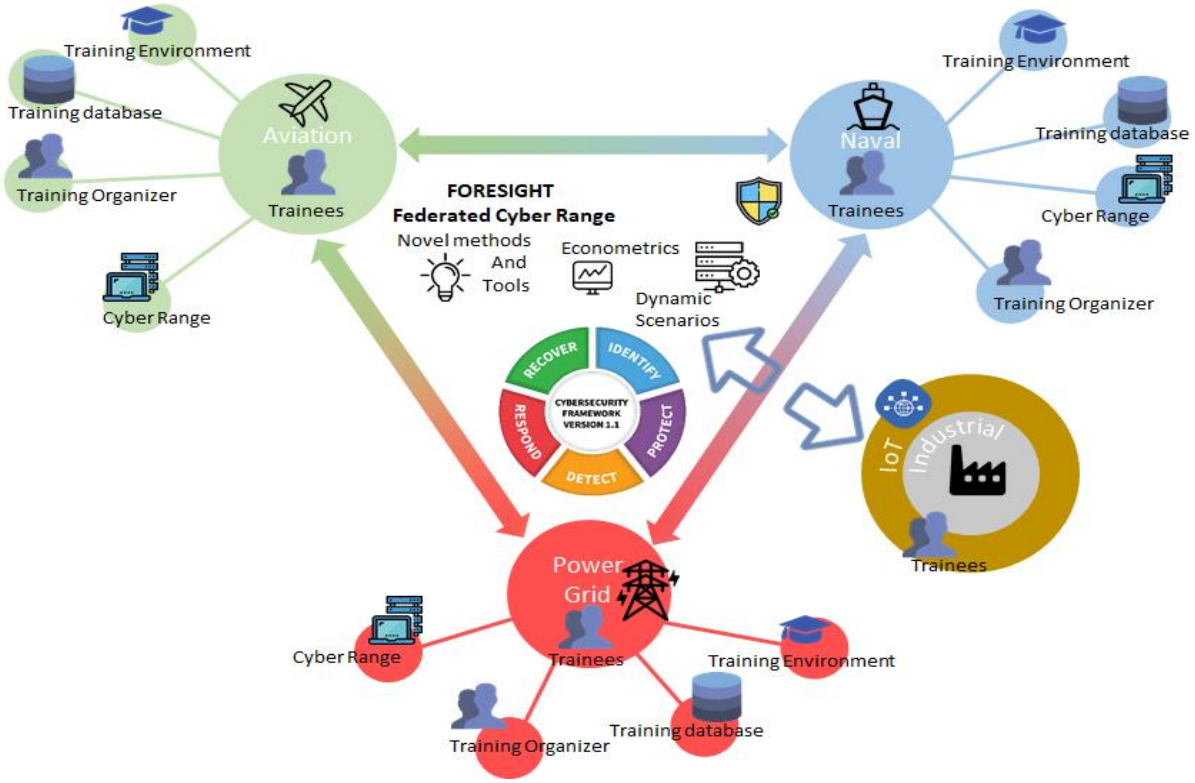

Figure 1. FORESIGHT project overview

The GAME module incorporates the necessary elements to enable the training of users and security professionals and experts on cyber-security aspects by taking a more sophisticated approach that relies on gamification. It aims at enhancing the experience of the user when interacting with the platform (e.g. to select from a list of training scenarios, view scenarios in different categories either by difficulty level or by training area) and introduce formal game elements. For example, the trainers will be able to allocate points to training scenarios, thus creating close coupling between learning and fun. The GAME module keeps a record of trainees' achievements and supports the creation of accomplishment schemes, e.g. it caters for the accumulation of predefined experience points, badges, etc.

The CSV module presents specific data that are made available in the context of training from other FORESIGHT components and cyber-ranges via easy comprehensible graphical representations and interactive visualisations so as to enhance data interpretation and manipulation. The modules provide

established visualisations, such as user's rankings and network topology, but also historical/statistical data are offered.

1.2 Relation tasks and deliverables

This deliverable is related to the FORESIGHT deliverables presented in the following table:

Table 1. Relation to submitted deliverables

Deliverable	Deliverable Title	Relation
D2.3	FORESIGHT Cyber-Range Requirement Report	The deliverable D2.3 "FORESIGHT Cyber-Range Requirement Report" presented all the requirements that relate to the components described in this deliverable.
D2.4	FORESIGHT Architecture report	The deliverable D2.4 "FORESIGHT Architecture report" presented the platform's high-level architecture and use cases for the components given in this deliverable.
D10.2	Federated User Interface	The deliverable D10.2 "Federated User Interface" presented an early version of the UI (in the form of mock-ups) that GAME and CSV components have.

In addition to those presented in Table 1, there are also interactions with the rest of WP9 deliverables, mostly with D9.2 "Training evaluation and scenario creation modules (I)" and D9.4 "Collaboration modules (I)". This deliverable is also related to the forthcoming FORESIGHT deliverables presented in the following table in order of submission deadline.

Table 2. Relation to forthcoming deliverables

Deliverable	Deliverable Title	Relation
D9.5	Gamification and visualisation modules (II)	The deliverable D9.5 "Gamification and visualisation modules (II)" will report the revised version of the GAME and CSV modules.
D10.4	Integration & Testing (I)	The deliverable D10.4 "Integration and Testing (I)" will document further work on the GAME and CSV modules towards integration with other FORESIGHT modules.

1.3 Deliverable's structure

The subsequent sections of the deliverable are structured as follows.

- Section 2 describes in detail the GAME module. The section first provides a high-level overview of GAME and how its expected requirements/functionalities are being developed along with some state-of-the-art and design aspects of the module (regarding its technologies, the DB schema, etc.). It proceeds with a more detailed description of its architecture and the available interfaces of other components that are being utilised.

- Section 3 provides the current state of the development of the CSV module. Likewise, the section presents a high-level overview and design aspects of the module as well as a more detailed description of its architecture and available interfaces. Moreover, it gives the details about the asynchronous messages expected to be exchanged with the cyber-ranges that are connected to the FORESIGHT platform.
- Section 4 describes the approach taken for unit testing so as to verify that the individual artefacts comprising GAME and CSV software modules operate as expected. These artefacts included units of source code, sets of one or more computer programs together with associated control data, as well as usage/operating procedures. Particular test cases are also presented.

Finally, Section 5 provides the deliverable's conclusions and information about future steps.

2 Gamification module

Users should feel engaged during their interaction with the FORESIGHT platform, which should provide them an instant way to monitor their performance and progress. As a result, the GAME module should be able to provide the users with all necessary information, through rewards and through the overall gamification content applied to the platform's components. In the subsequent sections the necessary tools to achieve such a goal are being described.

2.1 High-level overview

The gamification module applies gamification schemes and mechanisms in FORESIGHT platform, thus providing to trainers the ability to apply and/or create/modify the corresponding gamified elements and to trainees the way to monitor their progress through their actions. Since the gamification module augments the learning content, which is developed in Moodle[1] learning management system (LMS), the GAME module is designed to be a Moodle plugin for better performance and interaction amongst those FORESIGHT modules.

2.1.1 Objectives

The gamification module is one of the core components regarding the user-system interaction. Its purpose is not only to extend the system's functionality but also to provide the ability for users to monitor their own progression and to engage them during their interaction with the platform. As a result, the gamification's objectives are listed:

- Augment the content to engage the users interacting with FORESIGHT platform by rewarding their actions performed in maintaining and extending the content of FORESIGHT platform (e.g. about exercises, modules, etc.) and in communicating with other platform users (i.e. related to social aspects). Those rewards attained by users should be maintained
- Provide gamification schemes and tools for the trainees to apply/ create/ modify those schemes in the variety of the IC's content. Those schemes should be able to differentiated based on needs
- Provide different types of rewards based on the trainee's performance, in particular Badges and Achievements
- Provide leader-boards to users for competing with each other and monitor their performance
- Provide the ability to the users to add/ delete/ modify friend lists

To achieve all aforementioned and aiming at engaging and attracting the users enlisted in FORESIGHT platform, the GAME module is implemented in respect to the different gamification types of users noted under different content, either game specific and/ or industry-based applications. Furthermore, the GAME module is to extend the Moodle's functionality, serving as the Innovative Curricula (IC) module, through a carefully designed plugin that will interact with it and receive further input from the Training Evaluation (TE) module for the GAME module to accurately award users.

[1] https://moodle.org

2.1.2 Functionality coverage

The GAME module is described by a variety of requirements to be fulfilled and the aspects to be implemented in order to serve its purpose. The next subsections will describe the system's architecture and tools to be created in order to address all necessary requirements.

Related requirements

Table 3 lists the requirements related to the GAME module and the provisions made to support the fulfilment of these requirements.

Table 3. Requirements of GAME module and use-case references

REF_ID	Description of implementation	Use Cases
REQ-105	**Requirement:** Maximize the engagement and re-engagement of users. Engagement loops need to be user-centric as different users might require different engagement mechanisms. **Implementation:** The GAME module is consisted of several gamification schemes and diverse types of rewards originated by various sources, such as lesson modules, hands-on exercises and social interaction. The progress a user is performing regarding the gamification content has an on-growing difficulty.	UC-B-01
REQ-129	**Requirement:** The GAME module should allow users to manage their profile in respect to trainees' badges and achievements. **Implementation:** As the user progresses through the gamification content new rewards will be distributed such as Badges and/ or images to be displayed in their profile. Subsequently the users can choose to edit their profile accordingly.	UC-B-01
REQ-130	**Requirement:** The GAME component should assign reward scheme to scenario. **Implementation:** A trainer can select through different gamification schemes provided by the module and connect it to a specific content.	UC-B-03
REQ-131	**Requirement:** The GAME component should maintain/ present scores per trainee. **Implementation:** An internal database is implemented to store the progress of a trainee. In addition, they can view their progress and scores through their personal profile.	UC-B-01
REQ-132	**Requirement:** The GAME component should support multiple reward scheme per scenario **Implementation:** A trainer can apply a gamification scheme to a specific content. A Module in IC is consisted of different and multiple sub-modules, which can be independently connected with different gamification schemes	UC-B-03
REQ-133	**Requirement:** The GAME module should support new reward schemes. **Implementation:** A trainer can modify an existing gamification scheme to further match the needs of a content; thus, creating a new scheme.	UC-B-03

REQ-134	**Requirement:** The gamification modules must provide Achievements and track the user's progress on these. **Implementation:** A different number of Achievements have been implemented with different goals each. Each earned achievement by a trainee is stored in the game's database.	UC-B-01
REQ-135	**Requirement:** The GAME module must provide Badges. **Implementation:** A different number of Badges have been implemented with different goals each and can be earned by the user upon accomplishing those goals.	UC-B-01
REQ-136	**Requirement:** The GAME module must be able to utilize the user's performance and provide awards accordingly. **Implementation:** The GAME module receives input from IC and TE. Those two components can monitor the user's progress and performance and send the results to the GAME module. Based on these results the GAME will decide whether the user should be awarded and with which reward.	UC-B-03
REQ-137	**Requirement:** The GAME module must be able to utilize the user's time-to-completion and provide awards accordingly **Implementation:** The GAME module receives input from IC and TE. Those two components can monitor the user's progress and performance and send the results to the GAME module. Based on these results the GAME will decide whether the user should be awarded and with which reward.	UC-B-03
REQ-138	**Requirement:** The GAME module must keep a record of trainees' performance and social elements **Implementation:** The internal database of the gamification module stores the information regarding a user's progress for each Achievement implemented by the GAME module.	UC-B-03
REQ-139	**Requirement:** The GAME module must enable trainers to add formal game elements to basic training scenarios **Implementation:** The trainers of FORESIGHT platform can connect an existing gamification scheme to a specific Module implemented by IC.	UC-B-03
REQ-140	**Requirement:** The GAME module must introduce formal game elements in the scenarios **Implementation:** A trainee can be awarded through their progress and performance under different scenarios existing in the IC components and that are related to them.	UC-B-03
REQ-141	**Requirement:** The gamification module should be able to award a user based on his social networking. **Implementation:** Different awards have been implemented with their goals being described under social elements, received by the CTC module. Those elements are being evaluated in respect to the user's actions in regards to forums and comments, friends and teaming.	UC-B-02
REQ-142	**Requirement:** The Gamification module should be able to provide time limited events.	UC-B-04

	Implementation: Upon connecting a module with a gamification scheme, the trainer can select to specify a bonus period, meaning that users participating/ completing this module for the corresponding period will earn better rewards.	
REQ-143	**Requirement:** The Gamification module should provide leader boards. **Implementation:** A variety of leader-boards have been implemented by the gamification module, such as user leader-boards, team leader-boards, country leader-boards and friend leader-boards, listing the corresponding information.	UC-B-04

Related use cases

Table 4 lists the use cases related to GAME module and the provisions made to support their fulfilment them.

Table 4. Use-cases related to the GAME module

REF_ID	Description of implementation
UC-B-01	**Use case:** Single user at his/ her profile **Implementation:** Providing a list of different Badges, Achievements and awards in general to the users will engage them with the content. The users are able to gain experience points (EXP) and level up. Different levels indicate the level of user's growth and abilities.
UC-B-02	**Use case:** Teams at teams' profile - space **Implementation:** Part of the awards provided by the gamification module are being described as Social awards. The goals that the user needs to fulfil in order to obtain them are based on his actions on forums, such as posting, voting and teaming.
UC-B-03	**Use case:** System **Implementation:** The internal database of Gamification module stores all necessary information regarding the trainee's progress through the gamified content. In addition, it stores all the schemes that a trainer can connect to a specific content.
UC-B-04	**Use case:** Challenge – Labs - Exercises **Implementation:** Regarding challenges a dedicated page lists all active challenges made by IC module that the trainee can access. In addition, numerous leader-boards are implemented for the users to monitor and compare their performance against other platform users.

2.2 Design details

The content of gamification derives from games by introducing formal game-design elements in a non-game application or scenario. This technique, if used correctly can have the users to feel more engaged and increase their engagement and interaction with the platform [6]. There are many different elements that one can introduce to a non-game application to attract the users; with the most

common to be a point system, badges, rewards and leaderboard. Gamification has been used broadly in social or learning platforms[2].

The main aspect of the GAME module is to engage the users enlisted in FORESIGHT platform. The trainees should be able to monitor their progress and receive awards based on their actions. Comparison measures should be provided to them, such as leader-boards. However, different trainees need to be handled under diverse ways, since their personality differ for each one of them.

These personality types are being described as "gamification player types" and for each different type, the platform should provide a different tool for it to fulfil the trainees' needs. The most seen player types are being described under Bartles Player's taxonomy and are distinct in four main categories[3]:

- Killer
- Achiever
- Socializer
- Explorer

The above are being furthered described and analyzed in deliverable *D4.2 Report for Learning/ Training objectives, methodology and evaluation*. The gamification module is being implemented in a way to address all those different player types by providing different tools, awarding schemes and rewards for those trainees to use and chase for.

Examples of gamification mechanism in cyber security can be seen to have a great impact in the education aspect. From simple games, such as Game of Threats [1] and PenQuest [2], addressing a certain number of attacks and vulnerabilities from the defending side, quiz games or games that are more attack–centric [3], to more complex scenarios and games such as Project ARES [4] which includes a complete virtual world, with the ability for the user to navigate in it. Other professional training scenarios with gamification mechanisms include the University of California, Santa Barbara and the US National Security Agency's Cyber Defense Exercise [5].

Regardless the platform and the game implementation there are 7 core mechanics that can be used in all areas [1]–[6]:

- Badges: Each badge may describe a different single or multi goal completion. Having such a mechanism will engage both long-term and short-term users and regarding the former, users may be able to compete.
- Levelling system: Provides the ability to the users to level-up through earning experience points, distributed based on their actions. Such mechanism will enable long-term users to progress through their interaction and prove their progress, addressing different player-types.
- Leader-boards: This mechanism may offer a competitive style between the players and provoke them on studying harder while competing.
- Progress Bar: This mechanism can enable the feedback that a user needs to interact with the platform but also to be able to determine their progression.
- Virtual Currency: Another way to motivate users to constantly being engaged with the platform. If this mechanism is affected by the user's performance, may result in them trying harder to accomplish the specified goals.

[2] https://www.gamify.com/what-is-gamification
[3] https://www.interaction-design.org/literature/article/bartle-s-player-types-for-gamification

- Awards: Providing awards based on users' actions and performance may result in their greater motivation and engagement. The awards can either be pre-specified or known to the users or can be hidden.
- Challenges: Provides the ability to the users to compete through scenarios.

Having a platform hosting participant with different background and goals, those elements can be implemented to engage the most users with each one serving a unique purpose.

Gamification is an on-growing trend to be used in cyber-security systems and positive impact is noticed upon the trainee's performance [6] [7] and his/ her engagement [8]. KYPO is platform offering a scoring system, which is connected with the different weights appointed to the scenario's objectives [9] and the availability status of each one during the scenario; however, in the Red Team's score is assigned manually. Project ARES, being a large-scale progress, base its evaluation mechanism on the availability of the user to predict, detect and prevent a threat [10].

Apart from CR platforms, many other lightweight platforms are built, mostly for individual learners, such as HackTheBox[4], OverTheWire[5], TryHackMe[6], and CYWARIA[7]; which base their scoring system based on different metrics, with the latter evaluating the successfully stolen flags from another team, the number of successful defends when attacks occur and finally based on the team's ranking; the first team will get more points than the next to achieve the objective. Those platforms also offer leader-boards for the users to monitor the progress in regard to others and compete with each other, while also awarding badges to the users to indicate their knowledge or participating in a specific content.

In the FORESIGHT platform there are two types of users: long-term and short-term. Short-term users are those to participate in a training program for a few days. The gamification module's goal is to engage both types of users and if possible, to attract the short-terms to interact further with the platform after they have finished their organizational training.

To achieve such goal, the GAME module needs to introduce certain mechanics to address the users based on their player type and also engage them based on their needs. As a result, the gamification schema implemented is consisted of:

- Badges: Describing a single goal task for the user to acquire
- Achievements: Describing multi goal tasks for the user to acquire, that may be originated from multiple sources
- Leader-boards: To introduce competency amongst the users and a way for them to simultaneously communicate regarding each other's performance
- Awards: For the users to be rewarded based on their actions, which are consisted of experience points and/or profile pictures
- Levelling System: For the users to progress and monitor their performance.

The levelling system and the overall gamification scheme are made so that they are able to address both types of users. The levelling system consists of 25 total levels, with the starting levels to be easier to acquire in order to engage the short-term users, while the latter levels require much more effort, so that the long-term users feel challenged. It is worth mentioning that the number of levels can be further extended to address later needs, by adding new ones to the overall scheme. The user can earn

[4] https://www.hackthebox.eu/
[5] https://overthewire.org/wargames/
[6] https://tryhackme.com/
[7] https://www.soteria-int.com/product-cywaria/

experience points based on his/ her actions. The experience points that can be earned by a user are presented in Table 5.

Table 5. Experience points that a user may earn.

Points	CR	Social	Modules
Easy	5 (challenges)	2	5
Medium	10 (hands – on)	5	7
Hard	15 (hands – on)	–	10
Extra Hard	25 (hands – on)	–	14

By performing actions and completing content or by performing social actions the user is able to be awarded with the corresponding experience points. The required total exp for each level is presented in Table 6.

Table 6. The total experience points required for each level

LEVEL	EXPERIENCE	LEVEL	EXPERIENCE
1	7	14	212
2	9	15	276
3	12	16	358
4	15	17	466
5	20	18	605
6	26	19	787
7	34	20	1023
8	44	21	1330
9	57	22	1729
10	74	23	2248
11	97	24	2922
12	125	25	3799
13	163		

From this table it clearly derives that the first levels are easy to be acquired and the difficulty is rising significantly. The above classification derives from the formula bellow based on geometric progression

$$exp = C \cdot A^{level-1}$$

$$C = 7 \text{ and } A = 1.3$$

The above metrics are selected so that the GAME module can engage both types of users, short-term and long-term. Short-term users can progress even with a few days of training and gain the first 4-6 levels based on the training content and their progress and interaction with the platform. Furthermore,

long-term users will need to further interact with the platform and participate in more than one domain to earn the corresponding experience points for the higher levels.

The GAME module has three main sources from which the user can earn EXP: Lessons, Exercises, and Social actions. All of these may award users with a number of EXP, based on their actions. However, the GAME module is designed to distinct the source of EXP, and thus except from the experience points that a user hold in a particular time, he/ she also hold an amount of CR_{score}. The latter, indicates the amount of the total EXP, that is earned from Cyber-Ranges only, meaning the score originating from exercises. This implementation is to distinct users based on their actual skills and experience, with those with high total EXP but rather low hands-on training.

Regarding the badges and achievements, they will be rewarded based on the user's actions. Different awards will be created for each domain and category, so that they mirror the user's knowledge and progress in each one of them. Each badge, will indicate the user's knowledge upon a specific area. Example of achievements are presented in Figure 2.

Figure 2. Indicative images for achievements

The GAME module is being implemented as a Moodle plugin. Since IC is made under Moodle domain, those two components should coexist and interact with the GAME module to be able to monitor the trainee's performance. Moodle uses a MySQL database[8] as an internal storage and thus the same logic is adapted by the GAME module. In addition to the aforementioned, having an internal database will increase efficiency and serves for better and faster communication between the components of IC and GAME. Among the good practices when using containers in application development has to do with the storage of applications under a single container[9]. Since GAME relies on Moodle platform, it is preferable to be implemented in the same container; the fact that it is also a Moodle plugin, will assist in retrieving all necessary information faster and more efficiently.

2.3 Architectural aspects

2.3.1 Application architecture

Moodle is the main platform where all IC content will be stored and users will interact with. As a result, GAME component is implemented as a Moodle plugin for higher efficiency. Moodle, at its core aspects,

[8] https://docs.moodle.org/310/en/MySQL
[9] https://geekflare.com/container-best-practices/

uses PHP, MySQL and JavaScript. Those programming languages are being used in the GAME module, extending the functionality of the IC module.

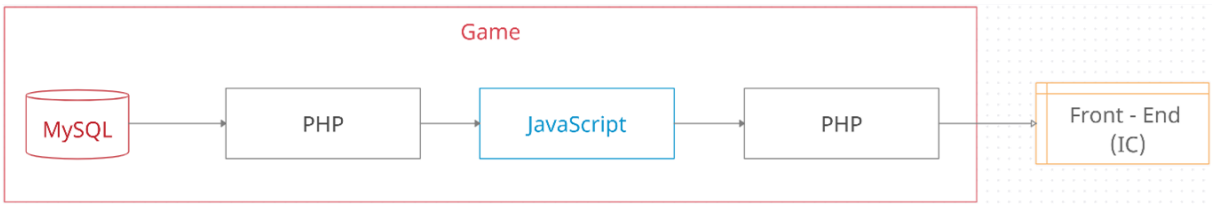

Figure 3. Programming languages are used by the GAME module

To implement the front-end of the application, PHP will handle all requests in regard to the user interface and perform the corresponding queries to retrieve from the database the information needed to be presented. JavaScript will handle all dynamic presentation of this data for the user to interact with. The game-logic is stored in a MySQL database, extending the core database of Moodle, while also storing all user progression.

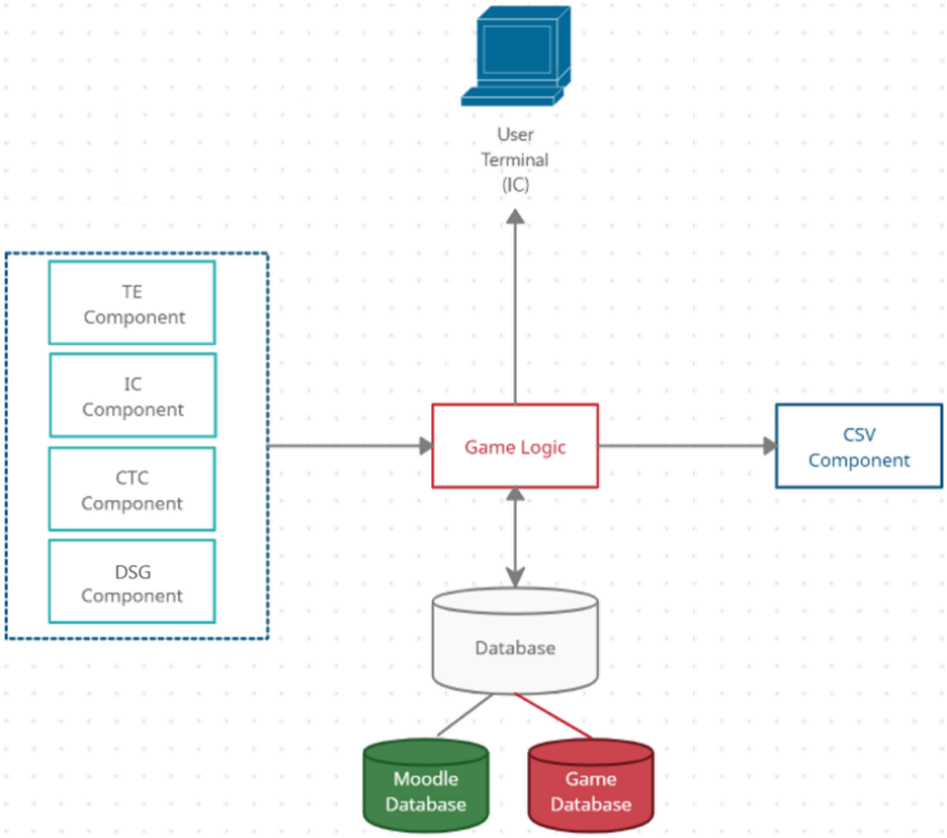

Figure 4. High-level architecture of the GAME module

High-level Architecture

The GAME module exists as a Moodle plugin having its own database storing all necessary data and PHP pages in regards to the interface. Subsequently, it communicates with IC component (Moodle) and TE to receive information regarding users. The CTC module also sends data to GAME module

regarding the trainee's actions in social aspects such as forums. In addition, it sends data to the CSV component for it to generate the corresponding graphs. Figure 4 and Figure 5 present the high-level architecture of the GAME module.

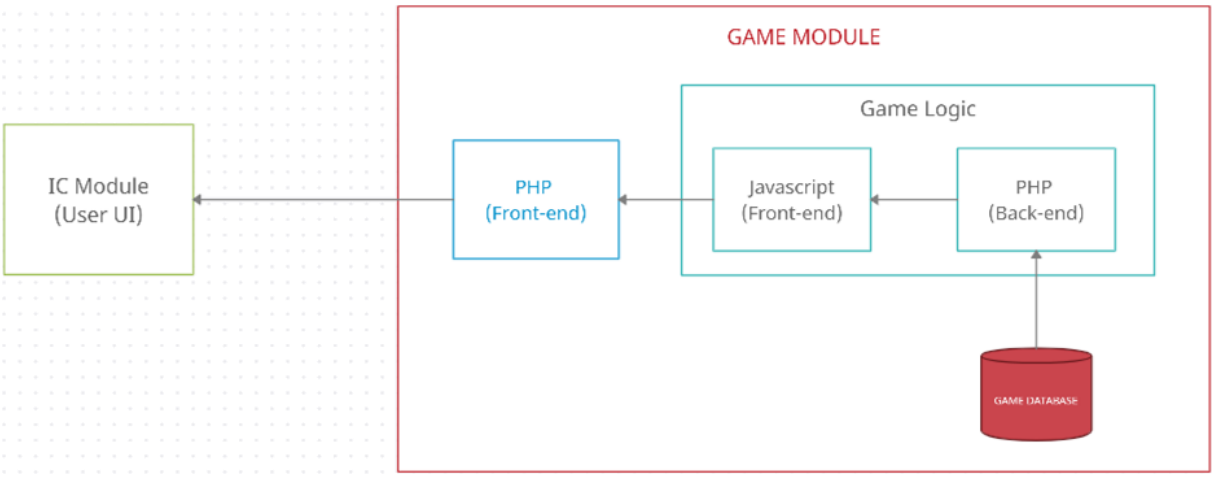

Figure 5. High-level architecture of the GAME module as a Moodle plugin

Data-centric Architecture

Data will be retrieved from modules of the FORESIGHT platform to monitor and calculate the user's progress and awards in terms of gamification. The modules to receive data from are:

- Training Evaluation module (TE): Data regarding a user's/ team's performance in a FORESIGHT content will be retrieved. Those data are essential to evaluate the awards and the user's/ team's progression in regards to the gamification content.
- Cyber Team Collaboration module (CTC): Data regarding a user's social activity will be retrieved. The user's actions in CTC module will determine the user's socialization and the awards that need to be appointed be the GAME module. More precisely the CTC should inform the GAME module when a user is creating a comment, when a user's comments receive likes and when a user is actively participating in forums. In addition, further activity logs and user's origin should be retrieved by CTC.
- Innovative Curricula module (IC): Data regarding the user's activity in regards to the content should be provided. In addition, all necessary information regarding the user, such as userID, username etc should be provided, for the GAME module to be able to monitor the user's actions inside the IC module. In addition, TEAM formation by users and their members should be retrieved. Finally, the IC should be able to provide information regarding the Content created in IC to connect it with the different gamification schemes created under GAME module.

Finally, the GAME module will send data to Cyber Security Visualization Module (CSV), regarding the user's progress and achievements for the corresponding visual graphs to be created. The various communication and data exchanges are presented in Figure 6.

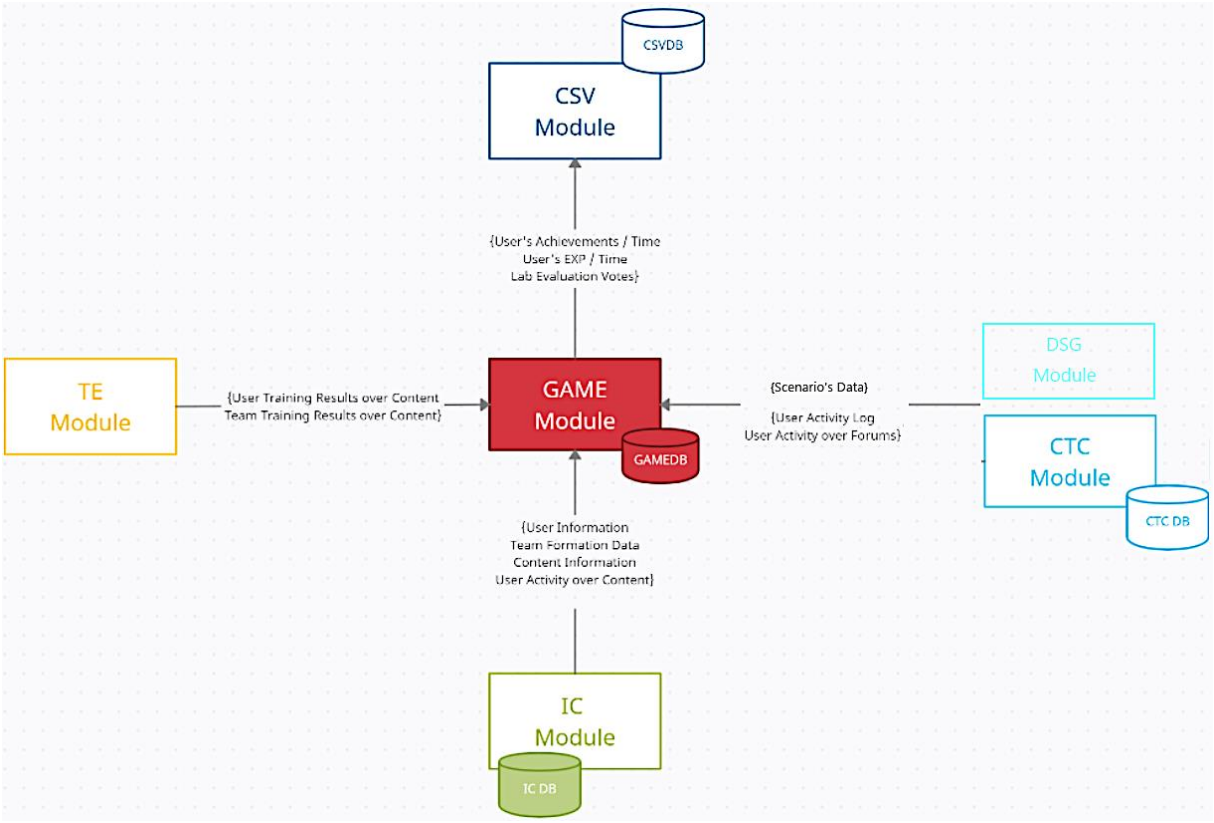

Figure 6. Data-centric architecture of the GAME module

To store all the necessary information for the GAME module to work properly and maintain the user's progress a database is being created using MySQL. This database extends the core functionality of Moodle be adding tables to describe and store the game logic. This database is presented in Figure 7.

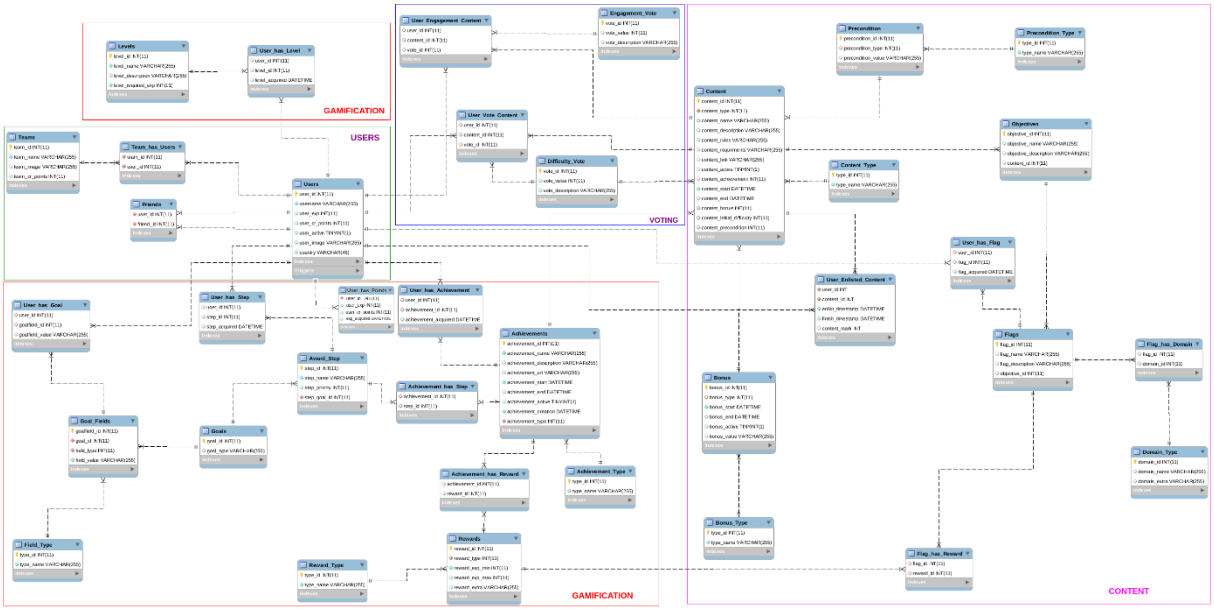

Figure 7. GAME module database tables

Figure 7 describes the most essential database tables for the GAME module to work. Each Achievement is described by each own Steps to be fulfilled, while each unique Step is described by its Goals to be performed by the User. In addition, each Achievement may offer extra rewards, such as experience points or an avatar for the user to customize his/her profile. Rewards are made to be able to be extended. Levels are stored in a separate table and the correlation tables store all information regarding the user's performance. Finally, separate tables store the information regarding the content existing in IC module and interconnect those with extra information, such as their domain and rewards upon completion if any.

Table 7. The User table of the GAME database

Attribute	Type	Example
User_id	INT	Stores the user's id originated from the IC module
username	VARCHAR	Stores the user's display name
User_exp	INT	Stores the current user's EXP
User_cr_points	INT	Stores the current user's points collected from CR sources
User_active	BOOL	Stores whether the user's account is active or no
User_image	VARCHAR	Stores the path to the user's avatar
country	VARCHAR	Stores the user's origin country

Table 8. The Levels table of the GAME database

Attribute	Type	Example
Level_id	INT	Stores the id of the level
Level_name	VARCHAR	Stores the level's display name

Level_description	VARCHAR	Stores description regarding the level
Level_required_exp	INT	Stores the total EXP required to achieve the level

Table 9. The User_has_Points table of the GAME database

Attribute	Type	Example
User_id	INT	Foreign key to the User's table
User_exp	INT	Stores the user's EXP at a certain time
User_cr_points	INT	Stores the user's points originated from CR sources at a certain time
Exp_acquired	DATETIME	Stores the timestamp when the user had the above information

Table 10. The User_Vote_Content table of the GAME database

Attribute	Type	Example
User_id	INT	Foreign key to the user table
Content_id	INT	Foreign key to the content table
Vote_id	INT	Foreign key to the Difficulty_Vote table, indicated the difficulty level that the user voted the specific content with

Table 11. The User_Engagment_Content table of the GAME database

Attribute	Type	Example
User_id	INT	Foreign key to the user table
Content_id	INT	Foreign key to the content table
Vote_id	INT	Foreign key to the Engagement_Vote table, indicated the engagment level that the user voted the specific content with

Table 12. The Content_Type table of the GAME database

Attribute	Type	Example
Type_id	INT	Stores the id of a unique content type
Type_name	VARCHAR	The name of the type, such as CHALLENGE, LESSON, HANDS_ON

Table 13. The Content table of the GAME database

Attribute	Type	Example
Content_id	INT	Stores the id for a content, originated from IC module
Content_type	INT	FOREIGN key to the Content_Type table
Content_name	VARCHAR	Stores the name of the content, originated from IC module

Content_Description	VARCHAR	Stores the description of the content, originated from IC module
Content_Rules	VARCHAR	Stores the rules of the content, originated from IC module
Content_ Requirments	VARCHAR	Stores the requirements of the content, originated from IC module
Content_link	VARCHAR	Stores the link where the module is displayed on IC
Content_active	BOOL	Stores whether the content is active or no
Content_ achievement	INT	Foreign key to the Achievements table. Could be empty
Content_start	DATETIME	Stores the timestamp on when the content started, originated from IC module
Content_end	DATETIME	Stores the timestamp for when the content to end, or have ended, originated from IC module
Content_bonus	INT	Foreign key to the Bonus Table. If the content has no bonus can be left empty
Content_initial_ difficulty	INT	Foreign key to Difficulty_Vote Table, indicating the difficulty of the content based on trainer's/creator's beliefs
Content_ Precondition	INT	Foreign key to the Precondition table, if the content has specific preconditions for the user to enter.

Table 14. The Bonus table of the GAME database

Attribute	Type	Example
Bonus_id	INT	Stores the id of the specific bonus
Bonus_type	INT	Foreign key to Bonus_Type table, indicating the type of bonus, such as EXP 20% extra
Bonus_start	DATETIME	Stores the timestamp when the bonus period started.
Bonus_end	DATETIME	Stores the timestamp when the bonus period to end
Bonus_value	VARCHAR	The mathematical description of the bonus. For instance, "+20%" to award users participating with 20% extra EXP.

Table 15. The Achievement_Typetable of the GAME database

Attribute	Type	Example
Type_id	INT	Stores the id of each unique achievement type
Type_name	VARCHAR	Stores the name of the achievement type, such as BADGES and ACHIEVEMENT

Table 16. The Achievements table of the GAME database

Attribute	Type	Example
Achievement_id	INT	Stores the id of each unique achievement

Achievement_name	VARCHAR	Stores the name of the Achievement
Achievement_ description	VARCHAR	Stores the description of this specific achievement
Achievement_uri	VARCHAR	Stores the Uniform Resource Identifier (URI) where the image of this achievement is stored
Achievement_start	DATETIME	Stores the timestamp when the achievement starts to be active. By default the value is assigned to the timestamp of creation
Achievement_end	DATETIME	Stores the timestamp for when the achievement to be deactivated. Can be left empty
Achievement_active	BOOL	Stores whether the achievement is active or no
Achievement_ creation	DATETIME	Stores the timestamp for when the achievement was created.
Achievement_type	INT	Foreign key to the Achievement_Type table indicating the type of this specific achievement

Table 17. The Award_Step table of the GAME database

Attribute	Type	Example
Step_id	INT	Stores the id of the specific Step that an achievement has.
Step_name	VARCHAR	Stores the name of the step
Step_priority	INT	Stores the priority of the step. The lower the number, the bigger the priority. By default, all steps are assigned as 1 priority. In case that an achievement needs from the user to perform the steps by a specific order, then priority should be assigned accordingly
Step_goal_id	INT	Foreign key to the Goals table, indicating the goals of this step

Table 18. The Goals table of the GAME database

Attribute	Type	Example
Goal_id	INT	The id of a goal to be stored
Goal_Type	VARCHAR	Stores the type of the goal, such as “FINISH_LESSON”.

Table 19. The Goal_Fields table of the GAME database

Attribute	Type	Example
Goalfield_id	INT	The id of the goal’s field to be fulfilled to acquire the goal.
Goal_id	INT	Foreign key to Goals table
Field_type	INT	Foreign key to Field_Type table indicating the type of this field.

Field_Value	VARCHAR	Describes the condition to be met, in order to achieve the goal's field. This field_value is in respect to the field_type.

Table 20. The Field_Type table of the GAME database

Attribute	Type	Example
Type_id	INT	The id of a type
Type_name	VARCHAR	The name of the type. For instance type_name: "LESSONS" type_name: "AMOUNT"

Table 21. The Reward_Type table of the GAME database

Attribute	Type	Example
Type_id	INT	The id of a type of reward
Type_name	VARCHAR	The name of the type, such as "EXP", "AVATAR".

Table 22. The Rewards table of the GAME database

Attribute	Type	Example
Reward_id	INT	The id of a reward
Reward_type	INT	Foreign key to the Reward_Type table
Reward_exp_min	INT	The minimum exp that this reward can award the user with. Can be zero or higher
Reward_exp_max	INT	The maximum that this reward can award the user with. Can be zero or higher. It can be assigned the same as reward_exp_min in case that no variation should be made.
Reward_extra	VARCHAR	In case that the reward_type is "AVATAR" the uri of the image is stored. In case where the exp_max and exp_min are not equal, the description on how the max and min exp are assigned should be stored, such as "TIME>20"

2.3.2 Technology stack

The GAME module is implemented under Moodle as a plugin. To provide efficiently its functionalities in terms of both end-user satisfaction and component communication it needs a variety of technologies that are being described in Table 23.

Table 23. Summary of the technologies used in GAME module

Tool	Version	Details
Moodle	3 (3.8)	Moodle is a learning platform used to augment and move existing learning environments online. Moodle is the core

		platform where the GAME module is deployed under IC module.
MySQL/ MariaDB	5.6 or higher/ 5.5.31	MySQL is a freely available open-source Relational Database Management System (RDBMS) that uses Structured Query Language (SQL). The database type used by both Moodle and the GAME module to store all necessary user information. MariaDB is developed as open-source software and as a relational database it provides an SQL interface for accessing data. Moodle is deployed under MariaDB services.
PHP	7 or higher	PHP is a recursive acronym for "PHP: Hypertext Pre-processor". PHP is a server-side scripting language that is embedded in HTML. It is used to manage dynamic content, databases, session tracking and more. The use of php is to create all pages for the interaction between the end-user and platform.
JavaScript	NodeJS: 14.15.0	JavaScript is a text-based programming language used both on the client-side and server-side that allows you to make web pages interactive. Used to create all necessary dynamic aspects of the pages presented to the user. JavaScript communicates with the corresponding PHP pages to retrieve and visualize all information regarding GAME logic.

2.4 Interfaces

The GAME module is made as part of IC module, implemented as Moodle plugin. Thus, the GAME module extends the functionality of Moodle, by offering a few extra functionalities.

Regarding the database, no Interface will be provided. For a user to visualize the game's database he/ she should directly enter the Moodle's structure and perform the actions needed. This excludes the trainer, who is possible to apply gamification schemes and created or modify awards. Thus, he/ she will be able to perform these actions through Moodle pages, created by the GAME plugin. All the aforementioned pages are presented in deliverable D10.2 "Federated User Interface".

Regarding the user's score and awards the Moodle will be extended with a number of pages to visualize all this information, regarding the Badges, Achievements, Leader boards, Gamified Lessons and content and the user's and team's gamified profile. A subscribed user is able to navigate through corresponding Moodle pages and find all information listed.

2.4.1 Application programming interfaces

The GAME module provides the game logic that is embedded and presented in the pages of Moodle. Therefore, there is no need for the GAME module to provide any REST endpoints or an API. This excludes the REST endpoints that will be used to retrieve information from other components.

The next sections present the API and REST endpoints and message formats that the GAME module expects to use in order to retrieve data from the other modules referred to in the previous sections, namely the CTC module, the DSG module, the TE module and the IC module.

The gamification module needs to interact and retrieve information from the IC and TE component regarding the content, the users and their performance through the exercises using API calls and interact with CTC and DSG module through REST API calls.

Communication with the IC module

The information needed to display all information regarding users and general information presented by the platform and the GAME module are mostly extracted from the IC module. Thus, the information and the calls needed to support such functionality is presented as follows:

- **GetUser(userID: INT)**: Based on a user's ID the function should return all information regarding the user (username, country, avatar, role, etc.)
- **GetAllUsers():** Returns a list of all user's IDs
- **GetUserTeam(userID: INT):** Based on a user's ID the teamsID he/she is a member of should be provided
- **GetTeam(teamID: INT):** Based on a team's ID the team's information should be provided (team name, members, avatar)
- **GetUserContent(UserID: INT):** Given a user's ID a list of all contentID that he/she is enlisted in should be provided
- **GetUserContentStatus(UserID: INT, ContentID: INT):** Given user's ID and a content's ID values regarding the progress of the user on the specific content should be provided (completed, grade)
- **GetAllContent():** A list of all ContentIDs should be provided
- **GetContent(ContentID: INT):** Based on contentID all information regarding the content should be provided (name, description, requirements, link, summary, objectives, domain, type)
- **GetContentResults(contentID: INT):** Based on contentID information regarding the users upon this content should be provided (enlistedUsers[completed], contentstatus,..)

The information regards both the users enlisted in the platform and the content to be provided, such as lessons, challenges, hands-on exercises, etc.

Finally, regarding the IC's content information and the user's actions upon it, messages should be retrieved. The message structure regarding a content's information is defined as follows in JSON format, for the general platform's information to be visualized:

```
/* Users' details data message */

[
```

```
   {
      "UserId": INT,
      "UserRole": VARCHAR, /* trainer | trainee | ... */
      "Username": VARCHAR,
      "UserCountry": VARCHAR,
      "UserImage": VARCHAR,
      "UserSubscription": DATETIME
   },
   ...
   /* more users */
]
```

All the fields have been included in the database of the GAME module, whereas `UserSubscription` is obtained from the IC and is the time at which a user's profile has been created.

```
/* Teams' details data message */

[
   {
      "TeamId": INT,
      "TeamName": VARCHAR,
      "TeamMembers": [
         UserId,
         ...
         /* more user IDs */
      ],
      "TeamImage": VARCHAR
   },
   ...
   /* more teams */
]
```

The fields of the message are defined in the following table.

Field	Description
`TeamId`	Team's unique ID
`TeamName`	Team's name
`TeamMembers`	A list of unique IDs that correspond to the users of the team
`TeamImage`	Team's picture (or avatar)

```
/* Content's details data message */

[
   {
```

```
      "ContentId": INT,
      "ContentName": VARCHAR,
      "ContentType": VARCHAR /* lesson | hands_on | challenge */
   },
   ...
   /* more content */
]
```

The fields of the message are defined in the following table.

Field	Description
`ContentId`	This is the unique ID given to gamification's content
`ContentName`	Name/title given to the content (depending on the type's value)
`ContentType`	This can be `lesson`, `hands_on` (for the virtual labs provided by cyber-ranges), or `challenge` (representing the content provided by training environments).

Note that the above message is also needed for getting information from the training evaluation (TE) and dynamic scenario generation (DSG) modules, as shown in section 2.4.1. Regarding the Users and Teams, information should be able to be retrieved upon request for a specific `user_id` or `team_id`. The same should be accessible for a specific content. In addition, regarding the user's evaluation performed under the IC module information should be provided regarding the user:

```
/* Evaluation results data message */

{
   "UserId": INT, /* OPTIONAL */
   "TeamId": INT, /* OPTIONAL */
   "ContentId": INT,
   "ContentType": "lesson",
   "ContentGrade": INT,
   "Completed": TIME
}
```

The fields of the message are defined in the following table.

Field	Description
`UserId`	User's unique ID; it is optional since the evaluation might concern a team as in the case of `hands_on` content type.
`TeamId`	Team's unique ID; it is not needed when the evaluation concerns a user. However, at least one of the `UserId` and `TeamId` should be present
`ContentId`	This is the unique ID given to gamification's content

ContentType	This equals lesson here, but in general it may take the value lesson, hands_on (for the virtual labs provided by cyber-ranges), or challenge (representing the content provided by training environments).
ContentGrade	The score achieved by a trainee or team to a specific lesson (also for hands_on, or challenge)
Completed	The time the lesson (or hands_on and challenge) was completed; it can be safely associated with the time the evaluation was submitted.

Note that an extended version of the above message is given in Section 2.4.1. In order to display a content and assign rewards to it, relevant information should be able to be retrieved; the data retrieved are shown below in JSON format.

```
/* Content details data message */

{
   "ContentId": INT,
   "ContentName": VARCHAR,
   "ContentType": "lesson",
   "ContentLink": VARCHAR,
   "ContentSummary": VARCHAR,
   "Prerequisites": [ /* OPTIONAL */
      {
         "PrerequisuiteId": INT,
         "PrerequisuiteDescription": VARCHAR
      },
      ...
      /* more prerequisuites */
   ],
   "DomainType": [
      "DomainId": INT,
      "DomainDescription": VARCHAR
   ],
   "Competencies": [
      {
         "CompetencyId": INT,
         "CompetencyName": VARCHAR,
         "CompetencyWeight": FLOAT
      },
      ...
      /* more competencies */
   ],
   "Difficulty": INT
}
```

The fields of the message are defined in the following table.

Field	Description
ContentId	This is the unique ID given to gamification's content
ContentName	Name/title given to the content (according to its type)
ContentType	This equals lesson here, but in general it may take the value lesson, hands_on (for the virtual labs provided by cyber-ranges), or challenge (representing the content provided by training environments).
ContentLink	Link to the content's page in Moodle if lesson
ContentSummary	Brief description of the lesson (or hands_on, challenge)
Prerequisites	A list of prerequisites, if any, that a trainee must have succeeded to (e.g. based on the learning path defined in IC) so as to access the content
.PrerequisuiteId	The prerequisite's unique ID
.PrerequisuiteDescription	Brief description of the prerequisite
DomainType	The domain the content belongs to; can be either Aviation, Power grid, Naval, or General (and possibly othersaccording to the training methodology
.DomainId	The domain's unique ID
.DomainDescription	The domain's name/title
Competencies	A list of competences to be acquired by the trainee when the content's objectives are achieved
.CompetencyId	Category's unique ID
.CompetencyName	The category's name as defined at the annex of deliverable D4.2, i.e. Fundamentals, Web security, ..., Steganography
.CompetencyWeight	The trainer's score on the competencies treated by lab (i.e. it can only be assigned by trainer)
Difficulty	The difficulty level of the content, an integer in the range 1 – 10, assigned by the trainer

Communication with the TE module

The TE module needs to provide results regarding the user's performance and outcomes through the exercises.

- **GetUserEvaluation(userID: INT, contentID: INT):** Based on a user's ID and the content's ID the function should return all information regarding the user's evaluation in this specific content (grade, completion time, performance data)
- **GetTeamEvaluation(teamID: INT, contentID: INT):** Based on a team's ID and the content's ID the function should return all information regarding the team's evaluation in this specific content (grade, completion time, performance data)

Regarding the information originated from TE component, the messages' structure for the user's results upon a content is defined in JSON format shown below. This is an extension of the respective message used for IC module to get the evaluation results about `hands_on` and `challenge` types of exercises from the TE module (these also have information about `Objectives` and `Flags`:

```
/* Evaluation results data message */

{
   "UserId": INT, /* OPTIONAL */
   "TeamId": INT, /* OPTIONAL */
   "ContentId": INT,
   "ContentType": VARCHAR, /* lesson | hands_on | challenge */
   "ContentGrade": INT,
   "Completed": TIME,
   "Objectives": [ /* OPTIONAL */
      {
         "ObjectiveId": INT,
         "ObjectiveStatus": INT, /* 0 for Fail | 1 for Pass */
         "ObjectiveSubmit": TIME
      },
      ...
      /* more objectives */
   ],
   "Flags": [ /* OPTIONAL */
      {
         "FlagId": INT,
         "FlagStatus": INT,
         "FlagSubmit": TIME,
         "TimeNeeded": FLOAT
      },
      ...
      /* more flags */
   ]
}
```

The fields of the message are defined in the following table.

Field	Description
`UserId`	User's unique ID; it is optional since the evaluation might concern a team as in the case of `hands_on` content type.
`TeamId`	Team's unique ID; it is not needed when the evaluation concerns a user. However, at least one of the `UserId` and `TeamId` should be present
`ContentId`	This is the unique ID given to gamification's content
`ContentType`	This can be `lesson`, `hands_on` (for the virtual labs provided by cyber-ranges), or `challenge` (representing the content provided by training environments).

ContentGrade	The score achieved by a trainee or team to a specific lesson, hands_on, or challenge.
Completed	The time the lesson, hands_on, or challenge. was completed; can be safely associated with the time the evaluation was submitted.
Objectives	A list of the objectives achieved by the trainee; this is optional since it might not be relevant to all ContentType (e.g. for a lesson)
.ObjectiveId	The objective's unique ID
.ObjectiveStatus	This indicates whether the objective has been achieved: 0 for Fail and 1 for Pass
.ObjectiveSubmit	The time at which the objective was achieved by the trainee
Flags	A list of flags submitted by the trainee; this is optional since it might not be relevant to all ContentType (e.g. for a lesson)
.FlagId	The flag's unique ID
.FlagStatus	This indicates whether the flag has been captured: 0 for Non-captured and 1 for Captured
.FlagSubmit	The time at which the flag was submitted by the trainee
.TimeNeeded	The time needed by the trainee to capture the flag (if)

Communication with the DSG module

The GAME module needs to retrieve information from the DSG module regarding the exercise's information, such as number objectives and flags. Since the DSG module is not part of Moodle, a number of REST API calls should be provided. The information needed is listed below:

- **GetContent():** Returns all the IDs of active exercises in the DSG module
- **GetContentInformation(ContentID: INT):** Returns all the information (objectives and flags) for the specific content ID.

The information to be received, is extending the message having already designed for retrieving the relevant information for content of type lesson by IC, and is described next in JSON format:

```
/* Content details data message */

{
   "ContentId": INT,
   "ContentName": VARCHAR,
   "ContentType": VARCHAR, /* lesson | hands_on | challenge */
   "ContentLink": VARCHAR,
   "ContentSummary": VARCHAR,
   "Prerequisites": [ /* OPTIONAL */
      {
         "PrerequisuiteId": INT,
         "PrerequisuiteDescription": VARCHAR
```

```
      },
      ...
      /* more prerequisuites */
   ],
   "Objectives": [ /* OPTIONAL */
      {
         "ObjectiveId": INT,
         "ObjectiveName": VARCHAR,
         "ObjectiveDescription": VARCHAR
      },
      ...
      /* more objectives */
   ],
   "Flags": [ /* OPTIONAL */
      {
         "FlagId": INT,
         "FlagName": VARCHAR,
         "FlagDescritpion": VARCHAR
      },
      ...
      /* more flags */
   ],
   "DomainType": [
      "DomainId": INT,
      "DomainDescription": VARCHAR
   ],
   "Competencies": [
      {
         "CompetencyId": INT,
         "CompetencyName": VARCHAR,
         "CompetencyWeight": FLOAT
      },
      ...
      /* more competencies */
   ],
   "Difficulty": INT
}
```

The fields of the message are defined in the following table.

Field	Description
`ContentId`	This is the unique ID given to gamification's content
`ContentName`	Name/title given to the content (according to its type)
`ContentType`	This can be `lesson`, `hands_on` (for the virtual labs provided by cyber-ranges), or `challenge` (representing the content provided by training environments).
`ContentLink`	Link to the content's page in Moodle if `lesson` or to an external system (cyber-range or training environment) if `hands_on` or `challenge`

ContentSummary	Brief description of what the lesson, hands_on or challenge is about
Prerequisites	A list of prerequisites, if any, that a trainee must have succeeded to (e.g. based on the learning path defined in IC) so as to access the content; it is not be relevant for content of hands_on and challenge type.
.PrerequisuiteId	The prerequisite's unique ID
.PrerequisuiteDescription	Brief description of the prerequisite
Objectives	A list of the objectives achieved by the trainee; this is optional since it might not be relevant to all ContentType (e.g. for a lesson)
.ObjectiveId	The objective's unique ID
.ObjectiveName	Objective's name/title
.ObjectiveDescription	Brief description of the objective
Flags	A list of flags submitted by the trainee; this is optional since it might not be relevant to all ContentType (e.g. for a lesson)
.FlagId	The flag's unique ID
.FlagName	Flag's name/title
.FlagDescritpion	Brief description of the flag
DomainType	The domain the content belongs to; can be either Aviation, Power grid, Naval, or General (and possibly othersaccording to the training methodology
.DomainId	The domain's unique ID
.DomainDescription	The domain's name/title
Competencies	A list of competences to be acquired by the trainee when the content's objectives are achieved
.CompetencyId	Category's unique ID
.CompetencyName	The category's name as defined at the annex of deliverable D4.2, i.e. Fundamentals, Web security, ..., Steganography
.CompetencyWeight	The trainer's score on the competencies treated by lab (i.e. it can only be assigned by trainer)
Difficulty	The difficulty level of the content, an integer in the range 1 – 10, assigned by the trainer

Communication with the CTC module

The GAME module needs to retrieve information regarding the user's action in the CTC module, such as commenting in forums, posting comments, etc. The information needed is listed below:

- **GetUserAction(UserID: INT):** Based on user's ID the user's actions should be received in terms of his/ her forum activity.

- **GetUserActivity(UserID: INT, login: DATETIME, logout: DATETIME):** Based on a user's ID the user's login activity should be received. Activity refers to the TIME when user logs in or out.

Regarding the CTC component, information about the user's social activity should be received. The message structure for the user's social activity is defined as follows in JSON format:

```
/* Users social message format */

[
   {
      "UserId": INT,
      "SocialType": VARCHAR, /* forum | chat | email | file | etc. */
      "Action": {
         "ActionId": INT,
         "ActionInfo": VARCHAR, /* OPTIONAL */
         "ActionTime": TIME
      }
   },
   ...
   /* other users' entries */
]
```

The fields of the message are defined in the following table.

Field	Description
UserID	The user's unique ID
SocialType	This identifies the type of social media used by the user; e.g. this could be forum, chat, email, file, etc.
Action	This contains additional data about a user's action
.ActionId	The identifier of the user's action in the CTC tools
.ActionInfo	Additional (optional) information about the post/comment a user has made in a forum, the message exchanged via the chat, etc.
.ActionTime	The time at which the user has performed the action

Similarly, the message for the user's activity is given below. Note that this message is also used by the CSV module to visualize aggregated information about users' activity.

```
/* Users activity message format */

[
   {
      "UserID": INT,
      "Login":  DATETIME,
      "Logout": DATETIME
   },
```

```
   ...
   /* other users' entries */
]
```

The fields of the message are defined in the following table.

Field	Description
`UserID`	The user's unique ID
`Login`	The (UNIX) time at which a user has logged into FORESIGHT platform
`Logout`	The (UNIX) time at which a user has logged out from FORESIGHT platform

Other API calls

The GAME module upon request, e.g. from the CSV module, can provide a list of information regarding the user's progress in respect to the gamification aspect. This information is presented in Table 24.

Table 24. The API calls supported by the GAME module

#	Function	Description	REQ_ID and Use Cases
1	GetUserScore(userID)	Based on a user's ID information regarding the user's score will be provided (score, cr_score)	REQ-131, REQ-141
2	GetUserScorePeriod(userID)	Based on a user's ID information regarding the user's score will be provided through a list of items (score, cr_score, timestamp) indicating the user's progression	REQ-131, REQ-138, REQ-141
3	GetUserAchievements(userID)	Based on a user's ID the earned achievements and badges will be provided	REQ-134, REQ-135
4	GetContentVotes(contentId)	Based on a content's ID a list of all votes will be provided.	REQ-105

Other data messages

The gamification module needs to communicate and receive information from IC regarding the users and the content, from TE regarding the user's evaluation upon a specific task and CTC component regarding the user's social activity.

No asynchronous messages are foreseen to be exchanged by the GAME module with other modules or external systems in the context of FORESIGHT. To send information regarding a user's progress in gamification, the general structure of a JSON format message is as follows.

```
/* User's progress message format */
```

```
{
   "UserId": INT,
   "UserExp": INT,
   "UserCrPoints": INT,
   "Badges": [
      {
         "BadgeId": INT,
         "BadgeDescription": VARCHAR,
      },
      ...
      /* other badges */
   ],
   "Achievements":[
      {
         "AchievementId": INT,
         "AchievementDescription": VARCHAR
      },
      ...
      /* other achievements */
   ]
}
```

The fields of the message are defined in the following table.

Field	Description
`UserId`	The user's unique ID
`UserExp`	The current amount of experience points (`Exp`) collected by the user, originated from all possible sources
`UserCrPoints`	The current amount of experience points (`Exp`) collected by the user, originated from CR sources only (i.e. from `hands_on`)
`Badges`	A list of all acquired badges by the user
`.BadgeId`	The badge's unique ID
`.BadgeDescription`	The badge's description
`Achievements`	A list of all acquired achievements by the user
`.AchievementId`	The achievement's unique ID
`.AchievementDescription`	The achievement's description

For the user's score progression, information regarding the score and timestamps will be provided; it is next displayed for the particular case of `hands_on` exercises in JSON format:

```
/* User's scores message format */
```

```
{
   "UserID": INT,
   "UserScores": [
      {
         "UserExp": INT,
         "UserCrPoints": INT,
         "Time": TIME
      },
      ...
      /* other scores */
   ]
}
```

The fields of the message are defined in the following table.

Field	Description
`UserID`	The user's unique ID
`UserScores`	A list containing information of the user's `Exp` in various timestamps
`.UserExp`	The total amount of `Exp` the user hold at a specific time, originating from all possible sources
`.UserCrPoints`	The total amount of `Exp` the user hold at a specific time, originating from CR sources only (i.e. from `hands_on`)
`.Time`	The time that the user held the aforementioned `Exp` amount

Another message is about a user's vote on the difficulty level of a challenge:

```
/* User's votes message format */

{
   "UserId": INT,
   "UserVotes": [
      {
         "VoteId": INT,
         "ContentId": INT,
         "VoteValue": INT
      },
      ...
      /* other votes */
   ]
}
```

The fields of the message are defined in the following table.

Field	Description
`UserId`	The user's unique ID
`UserVotes`	This is a list with the particular user's votes on various material
`.VoteId`	This is the vote's type unique ID
`.ContentId`	This is the unique ID of the content (in this case a `challenge`)
`.VoteValue`	The perceived difficulty level of a `challenge` or other content; it is an integer in the range 1 – 5, as voted by the trainee

2.4.2 User Interfaces

The GAME module will provide several user interfaces to achieve its goals. The interfaces are categorized in two major categories:

- Interfaces accessible by all users: Those interfaces will provide all information in regards to the gamified content of FORESIGHT platform, such as User Profile, Leader-boards, Gamified Modules, Badges, Achievements, etc.
- Interfaces accessible only by trainers: Those interfaces will assist the creation/ modification of existing gamification schemes and Rewards; in addition to their connection with content existing in the IC module.

The GAME UI will constitute an integral part of Moodle and therefore will be accessible to FORESIGHT platform users through the Moodle general interface. The GAME specific interfaces will extend Moodle functionality. A user will be able to access his/ her personal awards and monitor his/ her performance through the supported interfaces. Other game elements will be presented for each content and exercise for the user to observe the awards that he/ she can receive upon completing the particular content. Finally, the users will be able to create friends and through leader-boards to monitor their progression in comparison to other users of the FORESIGHT platform.

3 Cyber security visualisation module

The present chapter provides the details about the current state of development of the CSV module. The chapter's layout is similar to the previous one devoted on the GAME module, starting from a high-level overview and design/state-of-the-art aspects of the module to a more detailed description of its architecture and available interfaces.

3.1 High-level overview

3.1.1 Objectives

The CSV module provides visualisation facilities for the FORESIGHT platform. It aims to enrich the user experience of the IC and the GAME modules with graphs showing:

- Trainee progress over time, regarding achievements, points, etc.
- Team progress over time, regarding achievements, points, etc.
- Network graphs of the scenarios/labs offered by the platform, accompanied with relevant statistics regarding the trainees' outcomes and evaluations.
- Live network graphs of the scenarios/labs as they are being played, so that the evolution of the exercises is evident.
- Platform usage statistics regarding user and team participation and activity.

All these graphs should be highly interactive, allowing the user to customise the time period shown and the information displayed where appropriate. Furthermore, the graphs should accompany the corresponding information provided by the GAME module in a coherent and user-friendly environment.

In order to fulfil these objectives, the CSV module was implemented jointly with the GAME module as a plugin to the Moodle instance hosting the IC module. In this way, there is a central point where the user can find the training material, the gamification elements, and the visualisations of all this data.

3.1.2 Functionality coverage

Related requirements

Table 25 lists the requirements related to the CSV module and the provisions made to support the fulfilment of these requirements.

Table 25. Requirements of CSV module and use-case references

REF_ID	Description of implementation	Use Cases
REQ-083	**Requirement:** Active users and teams over time plot	UC-A-03

	Implementation: A line or bar chart presenting the number of users over time will be provided within a relevant dashboard. The user will be able to select the time period to be displayed.	
REQ-084	**Requirement:** Users per weekday plot **Implementation:** A bar chart presenting the number of users per week day will be provided within a relevant dashboard. The user will be able to select the time period to be displayed.	UC-A-03
REQ-085	**Requirement:** Users per hour of day plot **Implementation:** A bar chart presenting the number of users per hour of day will be provided within a relevant dashboard. The user will be able to select the time period to be displayed.	UC-A-03
REQ-086	**Requirement:** Users per month plot **Implementation:** A bar chart presenting the number of users per month will be provided within a relevant dashboard. The user will be able to select the time period to be displayed.	UC-A-03
REQ-088	**Requirement:** Users per challenge category plot **Implementation:** A bar chart presenting the number of users per challenge category will be provided within a relevant dashboard.	UC-A-03
REQ-094	**Requirement:** Scenario network graph **Implementation:** For each scenario / lab, a static graph showing participating computers and other equipment will be provided within a relevant dashboard, with appropriate icons for each of them and basic data (name, IP address – to be decided) on the graph for easy overview. In order to avoid cluttering, more details will be provided on hovering, see REQ-095. Furthermore, a dynamic / live graph will show the progress of the lab, whenever it is being played.	UC-A-04
REQ-095	**Requirement:** VM details pop-up on network graph **Implementation:** On the graph of REQ-094, the user will be able to hover over the different elements of the graph to see more detailed data (for example, Operating System, MAC Address, etc.)	UC-A-04
REQ-096	**Requirement:** Scenario user ratings plot **Implementation:** Users will be able to rate the difficulty of each scenario through the GAME module. A histogram showing the distribution of their ratings will be provided.	UC-A-04
REQ-097	**Requirement:** Times scenario was played plot **Implementation:** A line graph showing the number of times a scenario was played over time will be provided and displayed with the dashboard containing info on the scenario. The user will be able to select the time period to be displayed.	UC-A-04
REQ-098	**Requirement:** Team achievements over time plot **Implementation:** On the team profile page, a timeline showing the team achievements over time will be provided. The user will be able to select the time period to be displayed. Also, a line graph showing the actual number of achievements over time will be provided.	UC-A-02

REQ-099	**Requirement:** Team achievements per category over time plot **Implementation:** On the graphs detailed int REQ-098, team achievements will be visually differentiated (by colour, icon, or other) per category.	UC-A-02
REQ-100	**Requirement:** Team ranking over time plot **Implementation:** On the team profile page, a line graph showing the ranking of the team over time will be provided. The user will be able to select the time period displayed.	UC-A-02
REQ-101	**Requirement:** Trainee achievements over time plot **Implementation:** On the user profile page, a timeline showing the achievements of the trainee over time will be provided. Also, a line graph showing the actual number of achievements over time will be provided. The user will be able to select the time period to be displayed.	UC-A-01
REQ-102	**Requirement:** Trainee achievements per category over time plot **Implementation:** On the graphs detailed int REQ-101, trainee achievements will be visually differentiated (by colour, icon, or other) per category.	UC-A-01
REQ-104	**Requirement:** Trainee ranking over time plot **Implementation:** On the user profile page, a line graph showing the ranking of the trainee over time will be provided. The user will be able to select the time period displayed.	UC-A-01

Related use cases

Table 26 lists the use cases related to CSV module and the provisions made to support their fulfilment.

Table 26. Use-cases related to the CSV module

REF_ID	Description of implementation
UC-A-01	**Use case:** Profile visualisations **Implementation:** On the user profile page, several graphs will present visually the progress of the trainee: number of achievements (per category) over time, timeline of achievements (per category), ranking over time. Data will come from IC and GAME modules.
UC-A-02	**Use case:** Team visualisations **Implementation:** On the team profile page, several graphs will present visually the progress of the trainee: number of achievements (per category) over time, timeline of achievements (per category), ranking over time. Data will come from IC and GAME modules.
UC-A-03	**Use case:** System visualisations **Implementation:** Several visualisations depicting system-wide data will be provided in a coherent dashboard: total users over time, users per hour of day / day of week / month, users per challenge category. Data will come from the CTC and IC modules.

UC-A-04	**Use case:** Challenge – Labs – Exercise visualisations **Implementation:** On the dashboard showing lab information, a wealth of graphs will show important and relevant data: a radar plot showing the various security domains that this lab focuses on and respective percentages (relative importance) as specified by the creator of the lab; the number of times the lab was played over time; a histogram with the difficulty of the lab as rated by the users; and a network graph showing the nodes of the lab and their properties. Furthermore, a live version of the network graph will show the progress of the lab while it is being played. Data for these graphs comes from the GAME and the DSG modules; data for the lab progress will be provided by the Cyber-Ranges (CRs) via the platform message bus.

3.2 Design details

The CSV module is designed and implemented as a Moodle plugin, tightly integrated with the GAME module. In fact, all the CSV graphs are embedded in pages shared with the GAME module. In this way, trainees have a coherent view of their achievements and their progress through the gamified aspects of the platform; similarly, trainers have a unified overview of the scenarios/labs (both their network topology and relevant statistics) as well as access to live graphs depicting the progress of the exercises in near real time; finally, all users are able to see interesting statistics on user and team activity on the platform.

Earlier iterations of the CSV design assumed that the module would be essentially isolated of the other modules and provide its own window on the Federated User Interface, through which the users would be able to see all graphs relevant to their progress, monitor labs, etc. After careful examination of the FORESIGHT platform design and architecture, it was decided that such an approach would severely degrade the user experience, and thus it was abandoned in favour of the unified design outlined in the previous paragraph. This design has additional benefits with respect to the CSV module architecture and performance, as it will be explained in the next section.

Since the graphs provided by the CSV module are embedded in standard web pages, they can be implemented using standard web technologies: HTML, PHP and JavaScript. The most prominent JavaScript graphing library is **d3.js**[10] and it was selected as the main tool for CSV due to its extensive capabilities and great flexibility. Furthermore, numerous graphing libraries are built on top of d3.js and they can be leveraged for some specific types of graphs such as timelines and network graphs.

It should be noted that Moodle itself provides some simple types of graphs, such as line and bar charts, through its Charts PHP API. It was decided not to use them for two reasons: first, they do not cover all the graph types that CSV must implement, and even the ones that are available do not have all the capabilities envisaged for these types of graphs; second, implementing some graphs using one library and some using another would lead to inconsistencies (concerning the appearance and the interactivities of the graphs) which would adversely affect the user experience of the platform.

Finally, as explained in more detailed in Section 3.4.2, great care will be taken while developing the graphs in order to adhere to established visualisation and human-computer interaction principles while providing extensive data exploration capabilities to the users.

[10] https://d3js.org

3.3 Architectural aspects

3.3.1 Application Architecture

High-level Architecture

The CSV module is implemented as a Moodle plugin. Moodle also hosts the IC and GAME modules. Apart from the User Experience advantages of this approach outlined in the previous section, additional benefits pertain to the architecture and performance of the module:

- The CSV module does not need to provide its own user interface. Instead, it takes advantage of the facilities of Moodle, which is very extensible through various types of plugins: visualisations simply need to be embedded in standard web pages. Overall, a much simpler architecture and implementation is possible.
- The majority of the graphs provided by the CSV module visualise data of the GAME module. Since both modules co-exist within Moodle, direct access to GAME data is possible for CSV and, therefore, the needs for data interchange using expensive API calls are minimised. Of course, some graphs are constructed using external data from the CTC module, the DSG module and the CRs, as presented in more detail in Section 3.3.1.

The CSV module is comprised of the following components:

- **The visualisation engine.** This is implemented as a Moodle "local plugin" jointly with the GAME module.
- **A database.** CSV needs to store some data which are not available in the Moodle or the GAME databases. Even though the same database server as Moodle is used, a separate database is created specifically for CSV (just as another one is created for GAME) in order to avoid conflicts between these three databases.
- **A polling service.** It is envisaged that the CTC module will be polled periodically (for example, every hour) to retrieve user login/logout data in order to create system usage graphs. This service will call a CTC REST endpoint, receive the data in response, aggregate it and store it in the CSV database.
- **A RabbitMQ client.** This is necessary in order to retrieve data from the platform message bus regarding live scenario/lab execution; the data will be published by the CRs. This component will process the data appropriately and send it to the CSV module so that live scenario network graphs can be created.

The architecture of the CSV module is depicted in Figure 8.

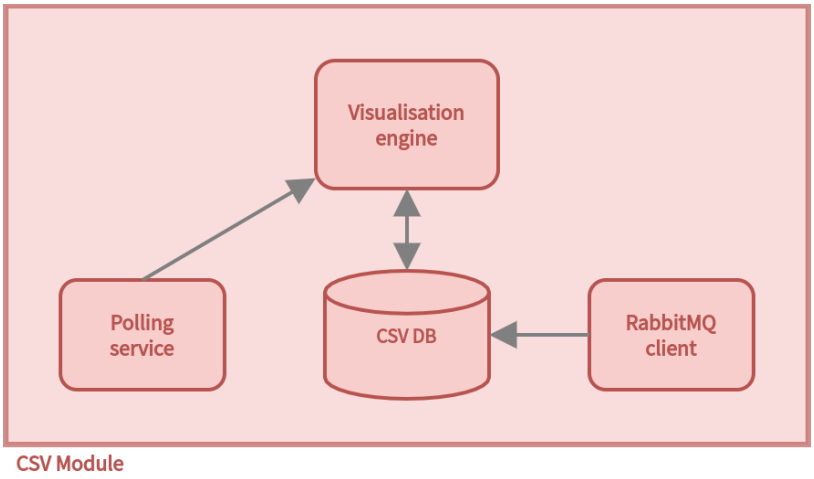

Figure 8. High-level architecture of the CSV module

Data-centric Architecture

Data is retrieved from various modules of the FORESIGHT platform to create and update the user and system graphs. The modules to receive data from are:

- Gamification (GAME): Data regarding the user's progress and achievements, held at the GAME module, will be used by CSV to create the corresponding visual graphs. Since both GAME and CSV will be implemented as Moodle plugins and reside on the same machine, CSV can access directly both the Moodle database and the GAME database (see Section 3.2) to query this data. Data on the Moodle database may also be accessed in a more structured way through Moodle PHP API calls.
- Cyber Team Collaboration module (CTC): CSV gets data regarding user logins and logouts and aggregates them in order to provide system usage graphs (users per hour of day/day of week/day of month). Data is expected to be retrieved at regular intervals and aggregated user counts will be stored in the CSV module own database; see below for the tables that will be needed and Section 3.4.1 for the messages that will be exchanged.
- Dynamic Scenario Generation (DSG): DSG holds the specifications of the scenarios/labs provided by the FORESIGHT platform. CSV will query DSG upon request to visualise a scenario/lab, in order to create the corresponding network graph. The data required for this task is a subset of the full scenario specification held at DSG and is detailed below in Section 3.4.1.
- Cyber Ranges (CRs): CSV needs from CRs data regarding live network graphs as they are played by users in order to create the relevant visualisations. This data will be put to the FORESIGHT message bus (RabbitMQ) by the CRs and CSV will subscribe to the relevant queue in order to retrieve it. The structure of the data message for this task is given below in Section 3.4.1.

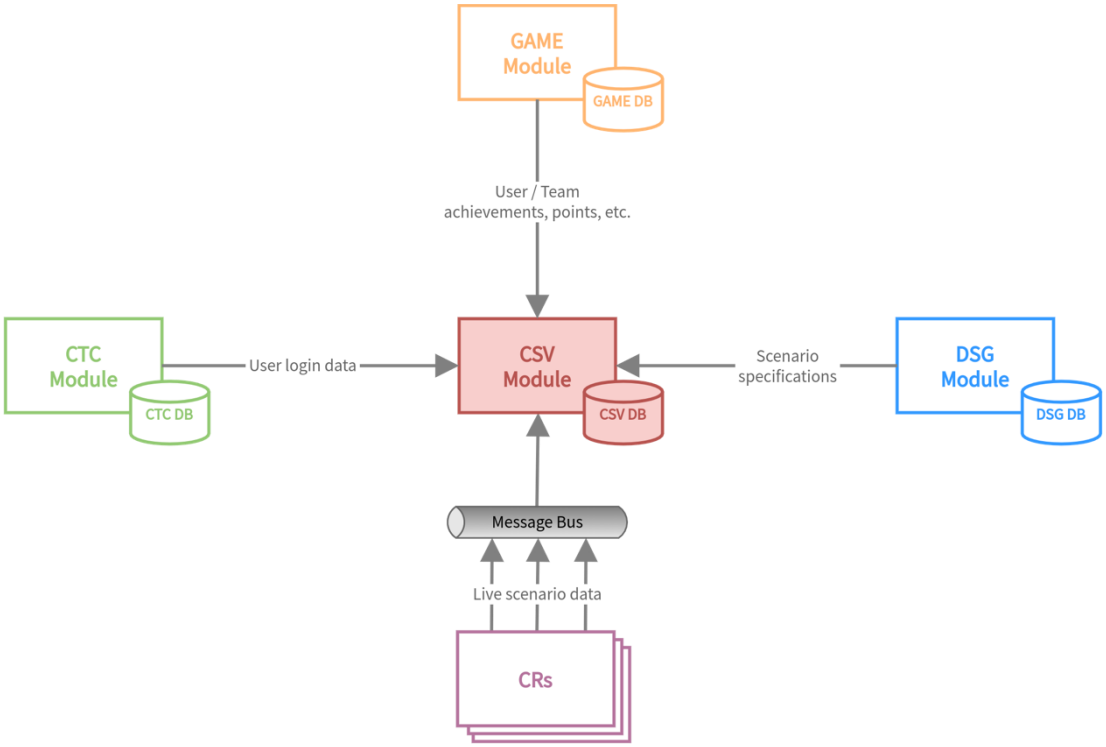

Figure 9. Data-centric architecture of the CSV module

The data required to produce system usage graphs will be stored in two database tables, named users_per_hour and users_per_month. These tables store aggregates of number of users per hour and users per month, respectively.

Table 27. The users_per_hour table of the CSV database

Attribute	Type	Example
ID	INT (auto increment)	0, 1, 2, 3, etc.
year	INT	2020, 2021, etc.
month	INT	1 for January, 2 for February, etc.
day_of_month	INT	1, 2, …, 31
day_of_week	INT	0 for Sunday, 1 for Monday, etc.
hour_of_day	INT	0, 1, …, 23
count	INT	256 (total count of users)

Table 28. The users_per_month table of the CSV database

Attribute	Type	Example
ID	INT (auto increment)	0, 1, 2, 3, etc.
year	INT	2017

month	INT	1 for January, etc.
total_users_ per_month	INT	2045

These two tables are used for answering the next queries in order to create the relevant graphs:

- Users per Hour of Day
- Users per Week day
- Users per Month

All other data is read by the Moodle and GAME module databases.

3.3.2 Technology Stack

The CSV module is implemented within Moodle as a plugin. To provide efficiently its functionalities in terms of both end-user satisfaction and component communication, it needs a variety of technologies that are being described in Table 29.

Table 29. Summary of the technologies used in CSV module

Tool	Version	Details
Moodle	3 (3.8)	Moodle is a learning platform used to augment and move existing learning environments online. Moodle is the core platform where the GAME module is deployed under IC module.
MySQL/MariaDB	5.6 or higher /5.5.31	MySQL is a freely available open-source RDBMS that uses SQL. The database type used by both Moodle and the GAME module to store all necessary user information. MariaDB is developed as open-source software and as a relational database it provides an SQL interface for accessing data. Moodle is deployed under MariaDB services
PHP	7 or higher	PHP is a recursive acronym for "PHP: Hypertext Pre-processor". PHP is a server-side scripting language that is embedded in HTML. It is used to manage dynamic content, databases, session tracking and more. The use of PHP is to create all pages for the interaction between the end-user and platform.
D3.js	V6	D3 is a JavaScript library for visualizing data with HTML, SVG, and CSS. All graphs of the CSV module will be created using D3, possibly with the help of some external plugin libraries for specific graphs (timelines, network graphs)
JavaScript	NodeJS: 14.15.0	JavaScript is a text-based programming language used both on the client-side and server-side that allows you to make web pages interactive. Used to create all necessary dynamic aspects of the pages presented to the user. JavaScript

		communicates with the corresponding php pages to retrieve and visualize all information regarding GAME logic.
Docker	20.10.5	Docker is a tool designed to create, deploy, and run applications in the form of containers. If the IC Moodle instance is provided as a Docker container, the joint GAME-CSV plugin can be integrated to the image. Furthermore, CSV will ship the polling service and the RabbitMQ client in dockerized form.

3.4 Interfaces

3.4.1 Application programming interfaces

The CSV module produces various graphs which are embedded in the pages of Moodle and the GAME module. Therefore, there is no need for the CSV module to provide any REST endpoints or an API. The next sections present the REST endpoints and message information that the CSV module expects to use in order to retrieve data from the other modules referred to in the previous sections, namely the GAME module, the CTC module, the DSG module and the CRs. The first one is omitted since the data required are obtained by directly querying the module's database.

Communication with the CTC module

The CSV module will query the CTC module for user activity (login and logout times). Therefore, it expects for the following REST endpoint to be provided by the CTC module (shown here as the function of CSV performing the retrieval):

GetUserActivity(login: DATETIME, logout: DATETIME): This call will return a list of user logins and logouts that were recorder in the time interval bounded by login and logout times. The response is expected to be provided in the following format:

The details of the CTC endpoint are elaborated during the integration phase, but the expected message format is shown below.

```
/* Users' Activity message format */

[
   {
      "UserID": INT,
      "Login":  DATETIME,
      "Logout": DATETIME
   },
   ...
   /* other users' entries */
]
```

The message can also be called as **GetUserActivity(UserID: INT, login: DATETIME, logout: DATETIME)** to get the entries for a specific user. The fields of the message are defined in the following table.

Field	Description
`UserID`	The user's unique ID
`Login`	The (UNIX) time at which a user has logged into FORESIGHT platform
`Logout`	The (UNIX) time at which a user has logged out from FORESIGHT platform

Communication with the DSG module

The CSV module will query the DSG module in order to retrieve the data to be visualised for a specific scenario/lab. This data is a subset of the scenario specification held at the DSG, as shown below. Therefore, the CSV module expects the following REST endpoint to be provided by the DSG module:

GetScenario(ScenarioID: INT): This call will return the following elements about the scenario with the given id. It is intended to complement the information retrieved by the GAME module in the respective call towards the DSG module by focusing on the network topology visualisation aspects.

```
/* Scenario details data message */

{
   "ScenarioID": INT,
   "ScenarioDescription": VARCHAR,
   "ScenarioStatus": VARCHAR,
   "PlatformID": INT,
   "Systems": [
      {
         "SystemID": INT,
         "SystemName": VARCHAR,
         "SystemOS": VARCHAR,
         "SystemOSversion": VARCHAR,
         "SystemType": VARCHAR,
         "SystemIP": IP Address,
         "SubnetID": VARCHAR,
         "Interfaces": [ /* OPTIONAL */
            {
               "InterfaceIP": IP Address,
               "SubnetID": VARCHAR
            },
            ...
            /* other interfaces */
         ],
         "Services": [ /* OPTIONAL */
            {
               "name": VARCHAR,
               "version": VARCHAR,
               "protocol": TCP | UDP |...,
```

```
                "port": INT
            },
            ...
            /* other services */
         ],
         "Vulnerabilities": [ /* OPTIONAL */
            {
               "VulnID": INT,
               "VulnName": VARCHAR,
               "cveID": VARCHAR,
               "Description": VARCHAR
            },
            ...
            /* other exploitations */
         ]
      },
      ...
      /* other systems */
   ],
   "Networks": [
      {
         "SubnetID": VARCHAR,
         "SubnetName": VARCHAR,
         "SubnetIP": IP Address,
         "Gateway": IP Address,
         "Systems": [
            INT, /* SystemID */
            ...
         ]
      },
      ...
      /* other networks */
   ],
}
```

The fields of the message are defined in the following table.

Field	Description
`ScenarioID`	This is the unique ID given to the scenario by the DSG module
`ScenarioDescription`	High level description of the particular scenario
`ScenarioStatus`	Current status of the scenario
`PlatformID`	This is the unique ID of the cyber-range hosting the scenario
`Systems`	A list with the details about the systems included in a scenario. **This information may not be shown to trainees but only to trainers.**
`.SystemID`	System's unique ID
`.SystemName`	System's primary host name

.SystemOS	System's operating system name
.SystemOSversion	System's operating system version
.SystemType	System's type; it is used to determine the system's visualisation (i.e. icon) and can be one of the following. ▪ *Devices:* Laptop, Personal Computer, Printer, Smart Phone, Tablet, Workstation ▪ *Networking:* Access Point, Bridge, Gateway, Hub, Router, Switch ▪ *Servers:* Active Directory, Application, Certificate, Database, DNS, Email, File, FTP, Generic, KMS, Proxy, Web ▪ *Security:* Attacker, CnC, Firewall, IDS ▪ *Other:* CCTV, Cloud, IoT Generic, IoT Sensor, IP Camera, PLC, RTU, Smart LED, Smart Meter As an example, "Servers - Email" is the value for an email server.
.SystemIP	System's *primary* IP address (may be generic/template)
.SubnetID	System's *primary* subnet ID
.Interfaces	A list of system's additional interfaces; this is optional since a system may have only one interface whose (primary) IP address was given above
.InterfaceIP	Interface's IP address (may be generic/template)
.SubnetID	Interface's subnet ID
.Services	A list of system's services that are made available. **This information may not be shown to trainees but only to trainers.**
.name	This is the name of the service, e.g. "openssh ssh"
.version	This is the version of the service
.protocol	The communications protocol of the associated service
.port	The network port number of the associated service
.Vulnerabilities	A list of system's designed vulnerabilities; this is optional since not all systems need to necessarily have vulnerabilities. **This information should not to be shown to trainees but only to trainers.**
.VulnID	The identifier of the vulnerability in the system
.VulnName	The name given to the vulnerability
.cveID	The CVE identifier of a vulnerability
.Description	Brief description of the vulnerability
Networks	A list of networks (subnets) included in the scenario. **This information may not be shown to trainees but only to trainers.**
.SubnetID	The subnet's unique ID
.SubnetName	The name given to the subnet
.SubnetIP	The subnet's IP address (may be generic/template)
.Gateway	The gateway's IP address (may be generic/template)

.Systems	A list of system IDs that belong into the particular network

Communication with the CRs

The CRs are expected to put data related to the progress of scenarios/labs being played to the FORESIGHT message bus (RabbitMQ), on a commonly agreed queue. The CSV module will subscribe to this queue and retrieve the respective messages, in order to create and update live network graphs.

The general structure of the messages will be the following:

```
/* Asynchronous data messages format */

{
   "header": {
      "source": "source-id",
      "msg_topic": "category-of-topic",
      "msg_id": "unique-message-identifier",
      "cor_id": "correlation-identifier",
      "timestamp": 1488179381517,
   },

   "payload": {
      /* dependent on the topic */
   }
}
```

The fields within the generic structure are as follows:

Field	Description
header	The header of the message containing useful metadata
.source	The identifier of the module that has produced the message
.msg_topic	An identifier of the topic of the message (controlled vocabulary). Can be e.g. training.eval, cr.event, alert, etc.
.msg_id	This is a universal identification number (UUID) that uniquely identifies each message generated by the platform's components.
.cor_id	The correlation identifier is an OPTIONAL field that is used to group individual messages together (in the form of a conversation). An initial message will have an empty cor_id; otherwise, if the message is a reply to an older message, then cor_id contains the msg_id of that message.
.timestamp	The UNIX time when the message was generated
payload	The actual payload of the message. It will contain varying fields, according to type, accommodated in the payload.

For the case of the *live scenario data* required by the CSV module, the payload must be in the following format:

```
/* LabLive data message */

   "payload": {
      "ScenarioID": INT,
      "PlatformID": INT,
      "SystemLive": [ /* OPTIONAL */
         {
            "SystemID": INT,
            "SystemIP": IP Address,
            "Status": UP | DOWN,
            "ResourceLoad": { /* OPTIONAL */
               "CPU": FLOAT,
               "Memory": FLOAT,
               "Storage": FLOAT,
               "Network": FLOAT
            },
            "ServiceActive": [ /* OPTIONAL */
               {
                  "name": VARCHAR,
                  "version": VARCHAR,
                  "protocol": TCP | UDP |...,
                  "port": INT
               },
               ...
               /* other services */
            ]
         },
         ...
         /* other systems */
      ],
      "NetLive": [ /* OPTIONAL */
         {
            "EndPoints": [
               IP Address,
               IP Address
            ],
            "BitRate": FLOAT,
            "PacketRate": FLOAT,
            "Loading": FLOAT,
            "Errors": FLOAT,
            "Latency": FLOAT
         },
         ...
         /* other links */
      ],
      "AttackLive": [ /* OPTIONAL */
         {
            "SystemID": INT,
```

```
            "VulnExploited": [
               {
                  "VulnID": INT,
                  "cveID": VARCHAR,
                  "AttackSources": [
                     IP Address,
                     ...
                  ]
               },
               ...
               /* other exploitations */
            ]
         },
         ...
         /* other systems */
      ]
   }
```

As shown above, the message has three parts:

- `SystemLive`: gives details on a system participating in a scenario: load data as well as services that are active
- `NetLive`: gives details on network traffic between the two endpoints.
- `AttackLive`: gives details on the vulnerabilities that have been exploited on this system.

All the parts are optional (but at least one should be present), and therefore the CRs broadcasting such a message may include only the parts that are relevant to the event that is being reported, and not re-transmit the whole information over the network.

Field	Description
`ScenarioID`	The lab's unique ID as given by the DSG (the term is used for the hands–on training performed in the CRs)
`PlatformID`	This is the unique ID of the cyber-range hosting the scenario
`SystemLive`	A list with details about a system's live data
`.SystemID`	System's unique ID
`.SystemIP`	System's IP address (during deployment)
`.Status`	A system's operation status (either on or off); this can change due to an attack
`.ResourceLoad`	The system's current usage of resources
`.CPU`	The system's CPU current load *
`.Memory`	The system's memory current load *
`.Storage`	The system's storage current load *
`.Network`	The system's network current load *

.ServiceActive	A list of system's services that are active
.name	This is the name of the service, e.g. "openssh ssh"
.version	This is the version of the service
.protocol	The communications protocol of the associated service
.port	The network port number of the associated service
NetLive	A list with the details about the network links' current status
.EndPoints	It is a list used to define the endpoints of a network link; i.e. it is a pair of IP addresses uniquely identifying a link
.BitRate	A network link's current bit rate *
.PacketRate	A network link's current packet rate *
.Loading	A network link's current bandwidth consumption *
.Errors	A network link's current errors *
.Latency	A network link's current latency *
AttackLive	A list with information related to attacks carried out towards systems of a virtual infrastructure
.SystemID	System's unique ID
.VulnExploited	A list of system's vulnerabilities that have been exploited
.VulnID	The identifier of the vulnerability in the system
.cveID	The CVE identifier of a vulnerability
.AttackSources	A list of system's targeting a given system

** These metrics are obtained so as to define historical data*

3.4.2 User Interfaces

Graphs produced by the CSV module will be embedded in pages created in co-operation with the GAME module within Moodle, thus providing a unified user experience with the IC module. The graphs constructed will conform to current visualisation and Human-Computer Interaction (HCI) principles and practices:

- Standard colour palettes will be employed, suitable for people with vision deficiencies.
- Intuitive interactive controls will be provided so that users can alter parameters of the graphs, such as the time period visualised, where appropriate.
- Tooltips will augment the information shown in the graphs.
- Axes will be clearly labelled and legends will be drawn where needed.

All interfaces are provided as mock-ups in deliverable D10.2.

4 Unit testing approach

Unit testing refers to the process of verifying that the individual artefacts comprising software modules operate as expected. These artefacts can be individual source code units, sets of one or more computer programs together with associated control data, as well as usage and operating procedures. The scope of verification in unit testing should involve both the externally observable behaviour of the method and any side effects that the unit has, such as updating repositories. The artefacts comprising a module are classified in a number of core layers as shown below.

- *Module's API:* The API exposed by a FORESIGHT module to the external world.

  The code in this layer is responsible for intercepting incoming API requests, extracting input parameters from the protocol-specific message, passing the request to the appropriate business logic module (typically to the service layer), retrieving the results, packing results back into protocol-specific messages and returning the result to the requesting client.

- *Module's service layer:* Defines the module's boundary to the outside world by encapsulating the core business logic.

  Since the functionality of modules is exposed through the associated API, it is expected that there is a one-to-one mapping between operations exposed by the module's API and the elements exposed by the service layer. Exceptions to this might be interactions of the modules with the innovative curricula (IC) tool, since both GAME and CSV are implemented as plugins to the learning management system (Moodle) of FORESIGHT.

- *Module's domain:* Contains the objects realising the business logic of the module (e.g. the players' rewards in the case of the GAME module).

- *Module's persistence:* Serves persistent domain objects to the backend of the system.

  The persistence layer manages the domain objects, which however are not necessarily all the domain objects. This layer may perform data mapping to deal with the representational differences between the repositories layer and the external data repositories (e.g. databases) where the domain objects actually persist.

- *Module's asynchronous communication layer:* Defines the push notifications sent to other modules, as well push notifications from other modules that are received and processed

  The asynchronous communication layer manages the creation/consumption of asynchronous messages, exchanged through the FORESIGHT's message bus; although this is not reflected in deliverable D2.4 "FORESIGHT Architecture Report", it has been an addition to the high-level architecture to allow efficient communication of cyber-ranges with the federated platform. A module's core business logic dictates that such messages should be created when important information about an event or condition must be made available to other modules; conversely, when such information is needed from other modules, relevant asynchronous messages are intercepted by the communication layer and passed to the module's core business logic for processing. This part mainly concerns the CSV module.

Unit testing of each module (GAME and CSV) included all the above layers, where the main approach taken is briefly documented in the following sections.

4.1 Unit tests per layer

4.1.1 Unit tests for the REST API layer

The REST API layer in both modules (GAME and CSV) comes from either open source software tools, which required extensions and customisation to cover FORESIGHT needs, or has been built from scratch; therefore this layer requires thorough testing. In both modules, code implementing parameter validation, as well as the overall implementation of the business logic through the REST API were rigorously tested to ensure the correct implementation of the documented functionality.

4.1.2 Unit tests for the service layer

The functionality exposed by modules' service layer was targeted by unit tests. To promote efficiency and isolation in unit testing at this level, it was recommended that any dependencies to other external services and data repositories be mocked, using stubs and pre-determined data. The unit tests used for the service layer investigated whether the correct operation was delivered when valid data were provided as input, whilst they the response of FORESIGHT modules to invalid inputs and business logic errors were also covered.

4.1.3 Unit tests for the domain layer

Classes and methods within the domain layer were targeted by unit tests. Typically, the classes packed within a single module have high cohesion and the operations of one class depend on other classes within the module. The approach taken during the unit testing at this layer was that such dependencies are not mocked; however, dependencies to other FORESIGHT modules were mocked to promote test autonomy and modularity. Similarly, to the case of the service layer, the unit tests used for the domain layer aimed not only to ensure the correct operation when valid inputs are supplied, but additionally the cases where invalid inputs are provided, as well as business logic errors.

4.1.4 Unit tests for the persistence layer

Classes and methods in the persistence layer were targeted by unit tests. Each operation in this layer typically requires no other information than the objects to be managed (and possibly elementary-type parameters). Therefore, each operation in the persistence layer was tested in isolation from the other parts of a module.

4.1.5 Unit tests for the asynchronous communication layer

Classes and methods in the asynchronous communication layer (message bus) of the CSV module, which requires communication with the cyber-range systems, were targeted by unit tests. At this stage, the asynchronous communication layer was examined in isolation, with the role played by the peer communication parties being mocked (i.e. fake senders and receivers were created). Tests related to

asynchronous communications and jointly involving the external FORESIGHT systems (i.e. the CRs) will be conducted at the integration phase.

4.2 Test cases and requirements

In this section, the unit test cases for the developed modules are presented. For those executed, their results are being presented. They also include references to the system functional and non-functional requirements as presented in sections 2.1.2 and 3.1.2 of this document. The priorities defined in the test cases are *low* (L), *normal* (N), *high* (H).

4.2.1 Gamification module

This first subsection introduces the test cases related to the gamification module.

Test Case ID	GAME-01	**Module**	Gamification
Description	The trainee should feel engaged while interacting with FORESIGHT platform. After interacting with the platform (Lesson participation and performing hands-on exercises) they should earn rewards, which can be viewed from their profile.		
Req ID(s)	Req-105, Req-129, Req-131, Req-134, Req-135	**Priority**	N
Prepared by	UOP	**Tested by**	UOP
Pre-condition(s)	User is already registered in FORESIGHT platform. User is already enrolled in a content and hands-on exercises.		
Test steps			
1	The user logs in to his/her Moodle account and finishes a Content associated with a reward		
2	The user accesses his/her profile under the “Gamification Navigation” Menu		
3	The user views his/her updated profile, which includes the newly gained rewards under the “Acquired Badges” and “Acquired Achievements” section.		
Input data	▪ Action performed by User (i.e. Content Finished) ▪ Above event captured by Moodle plugin		
Result	List of acquired rewards		
Test Result			

Test Case ID	GAME-02A	**Module**	Gamification
Description	The user should be rewarded based on his social activity such as forming teams and participating in forums.		
Req ID(s)	Req-141, Req-185	**Priority**	N

Prepared by	UOP	**Tested by**	UOP
Pre-condition(s)	User is already registered in FORESIGHT platform The user is already enrolled in a Content The Content's forum is already created		
Test steps			
1	User posts a comment on a forum		
2	When 10 users, different from the posting user, flag the posted comment as "liked", then the user is rewarded with 5 EXP		
3	User visits their profile to view their exp being increased		
Input data	▪ User comment, which is posted on the forum ▪ Action performed by users (Like on post)		
Result	User's exp score being increased		
Test Result			

Test Case ID	GAME-02B	**Module**	Gamification
Description	The user should be able to add other users of FORESIGHT platform as Friends or curate their friends list.		
Req ID(s)	Req-141, Req-185	**Priority**	N
Prepared by	UOP	**Tested by**	UOP
Pre-condition(s)	User is already registered in FORESIGHT platform		
Test steps			
1	User (u1) visits another user's (u2) profile with whom they are not friends		
2	User (u1) selects "Add Friend"		
3	User (u2) observes the "Friend request" listed in a relevant section of his/her profile		
4	User (u2) approves the invitation		
5	Users (u1/u2) visit their own profile		
6	Users can see their updated friend list		
Input data	▪ Friend request ▪ Friend request Acceptance/Rejection		
Result	▪ Friend Request is Generated ▪ Friend list is Updated		
Test Result			

Test Case ID	GAME-03A	**Module**	Gamification

Description	The platform should have the ability to maintain the awards presented to users		
Req ID(s)	Req-130, Req-132, Req-133, Req-136, Req-137, Req-138, Req-139, Req-140	**Priority**	N
Prepared by	UOP	**Tested by**	UOP
Pre-condition(s)	User is already enrolled in a Content The Content is already connected to the specified gamification scheme The gamification scheme has been altered (if needed) based on needs User has already received rewards		
Test steps			
1	User finishes the Content		
2	The system rewards the user with 5 EXP		
3	User visits their profile to view their EXP being increased		
4	The acquired EXP will not decrease (expire) in the future.		
Input data	▪ Action performed by User (i.e. Content Finished) ▪ Above event captured by Moodle plugin		
Result	▪ User's progress stored in GAME Database ▪ EXP does not decrease with time.		
Test Result			

Test Case ID	GAME-03B	**Module**	Gamification
Description	The platform should have the ability to maintain the awards presented to users, in addition to the ability to introduce multiple rewarding schemes that are able to be extended based on needs. Based on the situation the trainer should be able to design/ assign the most suitable awarding scheme.		
Req ID(s)	Req-130, Req-132, Req-133, Req-136, Req-137, Req-138, Req-139, Req-140	**Priority**	N
Prepared by	UOP	**Tested by**	UOP
Pre-condition(s)	User has a trainer role User has created the specific content in Moodle (i.e. Lesson)		
Test steps			
1	User access "Manage Content" page under the gamification menu		
2	User creates a new Content item and connects it to Moodle (using an appropriate URI)		
3	User selects the corresponding gamification scheme for that Content (i.e. Lesson)		

4	The user selects the difficulty of the content (e.g. EASY)
5	The appropriate gamification structures for the Content, as specified by the gamification scheme, are automatically generated
6	The user can add more objectives (goals) to the scheme
Input data	▪ Create Content ▪ Moodle Content URI ▪ Difficulty selection ▪ Gamification Scheme selection
Result	▪ The appropriate gamification structures are automatically generated ▪ Rewards automatically generated
Test Result	

Test Case ID	GAME-04A	**Module**	Gamification
Description	Time limited events should be provided		
Req ID(s)	Req-142, Req-143	**Priority**	N
Prepared by	UOP	**Tested by**	UOP
Pre-condition(s)	User has a trainer role User has created the specific content in Moodle (i.e. Lesson)		
Test steps			
1	The user accesses the “Manage Content” page under gamification navigation menu		
2	User creates a new Content and connects it to Moodle (using an appropriate URI). A form appears with information for the trainer to fill-in, such as Content Name, Description, etc		
3	User fills all necessary information and assigns to the content a time limited event.		
4	User specifies the period during which the event will be active		
5	User specifies the event type (i.e. Bonus EXP) and the rewards (i.e. +20%)		
6	After that period the event is finished		
Input data	▪ Create Content ▪ Moodle Content URI ▪ Difficulty selection ▪ Gamification Scheme selection ▪ Event Type ▪ Event Rewards		
Result	▪ Content is created ▪ Bonus Period is visible in corresponding page ▪ Event ends at corresponding time		
Test Result			

Test Case ID	GAME-04B	Module	Gamification
Description	Leader-boards should be provided with top N trainees, based on prespecified schemes		
Req ID(s)	Req-142, Req-143	Priority	N
Prepared by	UOP	Tested by	UOP
Pre-condition(s)	User is enrolled in FORESIGHT platform		
Test steps			
1	User access “Hall of Fame” under the gamification navigation menu		
2	User can view TOP N users of FORESIGHT platform. The Leaderboard is presented ordered by User acquired Points.		
3	User selects to order Leaderboard by “CR_Points” using the appropriate UI controls		
4	Leaderboard is updated based on CR_Points		
Input data	▪ Ordering Type (EXP, CR_Points)		
Result	▪ Leaderboard is created accordingly		
Test Result			

4.2.2 Cyber security visualisation module

The test cases are based on the requirements as described in document D2.4.

Test Case ID	CSV-01	Module	Cyber security visualisation
Description	The trainee sees statistics about their general progress over time.		
Req ID(s)	Req-101, Req-102, Req-104	Priority	N
Prepared by	UOP	Tested by	UOP
Pre-condition(s)	The FORESIGHT platform is up and running.		
Test steps			
1	The trainee accesses “Profile” from FORESIGHT game-csv navigation menu		
2	The trainee observes the corresponding graphs and plots		
3	The trainee can interact by narrowing down the results to a given time period		
Input data	There is no need for input data. However, the user can define the time period for some plots.		

Result	Visualisations related to the profile of the current user: ▪ Achievements over time ▪ Achievements over time per category ▪ Ranking over time
Test Result	

Test Case ID	CSV-02	**Module**	Cyber security visualisation
Description	The trainee sees statistics about the team progress over time.		
Req ID(s)	Req-098, Req-099, Req-100	**Priority**	N
Prepared by	UOP	**Tested by**	UOP
Pre-condition(s)	The FORESIGHT platform is up and running. The user should belong to a team for this to work.		
Test steps			
1	The trainee accesses “Team Profile” from FORESIGHT game-csv navigation menu		
2	The trainee observes the corresponding graphs and plots		
3	The trainee can interact by narrowing down the results to a given time period		
Input data	There is no need for input data. However, the user can define the time period for some plots.		
Result	Visualisations related to the team profile of the current user: ▪ Team Achievements over time ▪ Team Achievements over time per category ▪ Team Ranking over time		
Test Result			

Test Case ID	CSV-03	**Module**	Cyber security visualisation
Description	FR admins and CR admins see system statistics over time.		
Req ID(s)	Req-083, Req-084, Req-085, Req-086, Req-088	**Priority**	N
Prepared by	UOP	**Tested by**	UOP
Pre-condition(s)	The FORESIGHT platform is up and running.		
Test steps			
1	An admin accesses “System Info” from FORESIGHT game-csv navigation menu.		
2	The admin observes the corresponding graphs.		
3	The admin can interact by narrowing down the results to a given time period.		

Input data	There is no need for input data. However, the user can define the time period for some plots.
Result	Visualisations related to the entire FORESIGHT system: ▪ Active users and teams over time ▪ Number of users per day of the week ▪ Number of users per hour of day ▪ Number of users per month ▪ Number of users per challenge category
Test Result	

Test Case ID	CSV-04	**Module**	Cyber security visualisation
Description	Displays detailed information about an available scenario.		
Req ID(s)	Req-094, Req-095, Req-096, Req-097	**Priority**	N
Prepared by	UOP	**Tested by**	UOP
Pre-condition(s)	The FORESIGHT platform is up and running.		
Test steps			
1	A user accesses “Labs” menu item from FORESIGHT game-csv navigation menu.		
2	The user observes the difficulty of each lab.		
3	The user selects a specific scenario.		
4	The user observes the graph “Times played and outcomes over time”.		
5	The user observes the network topology of the scenario.		
6	The user clicks on the network topology graph in order to get the Live Topology plot.		
7	The user interacts with the live topology filters to get extra information about the nodes.		
Input data	The user selects a specific scenario and gets relevant information and plots back.		
Result	Visualisations and information related to the chosen scenario. ▪ Difficulty histogram for each scenario ▪ Topology graph for each scenario ▪ Live Topology graph for each scenario ▪ Information for each machine in a scenario ▪ Times played and outcomes over time plot		
Test Result			

5 Conclusions

The deliverable provided a high-level overview of the GAME and CSV modules and how they have proceeded to satisfy the requirements and expected functionalities. State-of-the-art aspects were also included in the report along with the design details for each module (regarding the technologies used, the DB schema, etc.). The deliverable also gave a detailed description of the modules' architecture as part/plugins of the IC tool, i.e. Moodle; this approach is expected to provide a considerably improved integration of the GAME and CSV modules with the tools that the trainees and trainers are most used to access. This approach also defined the APIs being developed to provide the envisaged functionalities and interact with other FORESIGHT modules.

As part of the currently ongoing effort for a first integration of FORESIGHT modules, the functionalities and APIs documented herein will evolve to support this and meet the work-package's and project's objectives. This improvement process will continue during the second development and integration cycles that will result into the final version of the GAME and CSV modules.

6 References

[1] PwC, “Game of Threats™ – Cyber threat simulation”. [Online] Available at https://www.pwc.ru/en/publications/game-of-threats.html

[2] R. Luh, M. Temper, S. Tjoa, S. Schrittwieser, and H. Janicke, "PenQuest: a gamified attacker/defender meta model for cyber security assessment and education,” J. Comput. Virol. Hack. Techn., vol. 16, pp: 19–61, Mar. 2020, DOI: 10.1007/s11416-019-00342-x.

[3] M. Adams and M. Makramalla, “Cybersecurity skills training: An attacker-centric gamified approach,” Technol. In nov. Manage. Rev., vol. 5, no. 1, pp: 5-14, Jan. 2015. DOI: 10.22215/timreview/861

[4] D5-IQ, “Project ARES: The next generation AI-powered platform for cyber readiness” Circadence Corp., 2018 [Online] Available at https://gamifiedcyberlearning.com/wp-content/uploads/Project-ARES-D5-IQ.pdf

[5] M. Gondree, Z. Peterson and T. Denning, “Security through play,” IEEE Security and Privacy Magazine, pp. 64-67, 2013. DOI: 10.1109/MSP.2013.69.

[6] S. Scholefield and L. A. Shepherd, “Gamification techniques for raising cyber security awareness,” in HCI for Cybersecurity, Privacy and Trust, A, Moallem, Ed., Cham, Switzerland: Springer, 2019.

[7] S. Duggan and C. Thorpe, “An analysis of how information security e-learning can be improved through gamification of real software issues,” in Proc. 16th Eur. Conf. Cyber Warfare and Secur., M. Scanlon and L.-K. Neihn-An, Eds., Dublin, Ireland, Jun. 29-30, 2017, pp. 666-672

[8] SC Media, “Gamification: A winning strategy for cybersecurity training”, Sep. 2019. [Online] Available at https://www.scmagazine.com/home/opinion/executive-insight/gamification-a-winning-strategy-for-cybersecurity-training

[9] P. Čeleda, J. Cegan, J. Vykopal, and D. Tovarnák, “KYPO – a Platform for cyber defence exercises,” 2015.

[10] ARES project, “Development of reconfiguration mechanisms”, 2012. [Online] Available at: https://crises-deim.urv.cat/ares/