**Table of contents**

**Glossary of acronyms** (1 page)

**Acknowledgments** (1 page)

**Abstract** (1 page)

**List of figures**

**List of tables**

**General introduction** (1 page)

1. **Project Context** (8-9 pages)
   1. Introduction
   2. Host company presentation
      1. Sofrecom Tunisia presentation
      2. Sofrecom activity
   3. Project presentation
      1. Problem Statement
      2. Project objectives
   4. Critical analysis of the State of the Art
      1. State of the art
         1. Samknows One
         2. SMAQ
      2. Critical of the state of the art
      3. Proposed solution
   5. Modeling language
   6. Software development methodology
   7. Conclusion
2. **Requirements analysis and specification** (12-14 pages)
   1. Introduction
   2. Identification of the Actors
   3. Functional requirements
   4. Nonfunctional requirements
   5. Requirements analysis
   6. Conclusion
3. **Design of the physical and logical architectures** (18-20 pages)
   1. Introduction
   2. Physical architecture
   3. Logical architecture
      1. Conceptual model
      2. Modular decomposition
         1. Class diagram of entities
         2. Package diagram
         3. Devices management model
         4. Metrics and test management model
         5. Network monitoring model
         6. Statistics and charts management model
         7. Authentication and users management model
      3. Broadband supervision and monitoring theory
   4. Conclusion
4. **Project Achievements** (18-20 page)
   1. Introduction
   2. Developing environment
      1. Hardware environment
      2. Software environment
      3. Frameworks and technologies
   3. Achieved work
      1. Authentication and users management
      2. Devices management
      3. Metrics and tests management
      4. Statistics and dashboard management
      5. Alerts management
   4. Conclusion

**General conclusion and future work** (1 page)

**Bibliography**

**Netography**

**Glossary of acronyms**

**Acknowledgements**

In conducting this report, I have received meaningful assistance from many quarters which I like to put on record here with deep gratitude and great pleasure.

First and foremost, I express my sincere gratitude to my supervisors, Mr. Sabri Mtibaa, Ms. Marwa Drissi, and Mr. Aref Meddeb who extended their complete support and helped to make me deliver my best. I would also like to thank all of my teachers at the National Engineering School of Sousse for their continuous help and treasurable training during our study years.

Finally, special thanks to the jury members who honored us by examining and evaluating this modest contribution.

**Abstract**

This report describes the design and the development of our graduation project internship, which is carried out at Sofrecom Tunisia, the project consists in creating a remote and communicating devices platform to achieve broadband supervision and monitoring and assess the quality of service “QoS” of fixed network access.

With our solution, the user could manage connected probes and create broadband tests configuration before running them in the probes, also he could manage statistics configuration with attractive charts for testing results. Users could create customized alerts configuration to simplify broadband supervision. The project also helps network supervisors to get accurate statistics according to a time period and zone area.

**Keywords:**

Probe – Spring boot – Angular – Raspberry Pi – Java – Kafka – Broadband – Quality of experience

**Résumé**

Ce rapport décrit la conception et le développement du projet de notre stage de fin d'études, qui est effectué chez Sofrecom Tunisie. Il consiste à créer une plateforme d’équipements distants et communicants permettant de superviser et de contrôler le haut débit et d'évaluer la qualité de service « QoS ».

Avec notre solution, l'utilisateur peut gérer les sondes connectées et exécuter des tests pour apprécier la qualité de service du réseau perçue par les clients. Il peut également gérer la configuration statistique des résultats de tests avec des graphiques attrayants. Les utilisateurs peuvent configurer des alertes personnalisées afin de simplifier la supervision du haut débit. Le projet aide également les superviseurs de réseau à obtenir des statistiques précises en fonction des zones de localisation et des périodes temporelle.

**Mots clés:**

Sonde – Haut débit – Qualité de service – Qualité d’expérience – Indicateurs clé de performance – Analyse des donnés

**ملخّص**

يتضمن هذا التقرير تصميم و تطوير مشروع التخرج، و الذي تم تنفيذه في مؤسسة "سوفريكوم" تونس. يتمثل هذا المشروع في إنشاء منصة للأجهزة المتحكم فيها عن بعد، و ذلك لتحقيق الإشراف على شبكات النطاق العريض و مراقبة و تقييم جودة الخدمة.

من خلال هذه المنصة ، يمكن للمستخدم إدارة إعدادات الأجهزة المتصلة و إنشاء اختبارات لتقييم شبكة النطاق العريض ، كما يمكنه إدارة الإحصائيات عبر إنشاء مخططات حسب الطلب لنتائج الاختبار. يمكن للمستخدمين إنشاء تنبيهات مخصصة لتبسيط الإشراف على شبكة النطاق العريض. يساعد المشروع أيضًا مشرفي الشبكة على الحصول على إحصائيات دقيقة وفقًا لفترات زمنية ومناطق محددة.

**كلمات مفاتيح:**

مسبار – نطاق عريض – جودة الخدمة – جودة التجربة – مؤشر الأداء – تحليل البيانات

**General introduction**

With the global spread of the internet nowadays, many mobile and internet services operators appear. In the coming days, connected devices will spread massively worldwide, especially with the appearance of new technologies such as the internet of things, big data, personal area networks (PAN), artificial intelligence (AI). So mobile and internet services operators desire to achieve a better quality of service for their customers. To do so, operators design and develop solutions for broadband monitoring and supervision mainly to massively assess and monitor the quality of service of networks access perceived by customers. Sofrecom Tunisia, subsidiary of Orange Group, attempts to serve this purpose by designing and developing a platform for broadband supervision and monitoring and specifically to massively assess the quality of service (QoS) of fixed network access. In this context comes our mission during the graduation project internship.

This report contains four chapters as follows:

The first chapter titled “Project context” will be devoted to the company presentation and setting the project in its general context.

The second chapter “Requirements analysis and specification” will be about the global system analysis to design and develop.

The third chapter “Design of the physical and logical architectures” will be dedicated to present physical and the logical architectures of our solution, also, we will explain the theory behind the broadband monitoring and supervision.

At last but not least, the final chapter “Project achievements” will be about the application achievements. First, we will present the developing environments and technologies and tools, secondly, we will illustrate our application with several interfaces.

Finally, we will close the report with a general conclusion, future work and perspectives will be mentioned at last to illustrate some ideas to improve our solution.

**Chapter 1: Project context**

* 1. **Introduction:**

In the first place, this chapter will be about the host company Sofrecom Tunisia presentation, a consulting and engineering firm specializing in telecommunications. Then, we will talk about the work and the project environments, by putting the light on the project goals and the analysis of the state of the art. Finally, we will present our software development methodology and the modeling language that we are going to use.

* 1. **Host company presentation:**

Sofrecom, a subsidiary of Orange Group, has built up 50 years’ worth of unique know-how in the telecoms operator line of business, making it a world leader in telecom consultancy and engineering. Sofrecom Tunisia is the youngest subsidiary of Sofrecom. Launched in November 2011, Sofrecom Tunisia expands Sofrecom’s presence in North Africa and the Middle East region to meet the growing demand for dedicated solutions and to offer its customers adapted and competitive offers.

* 1. **Project presentation:**
     1. **Problem statement:**

Operators of mobile and internet services have probes, Key Performance Indicators (KPIs), installed in several network elements, but they desire to appreciate the quality perceived by their customers (Quality of Experience, QoE) in order to improve their network performance and reliability.

In order to get accurate information about the quality of experience we should get the measurements according to zone area and specific periods of time. Also, it is difficult to manage the configuration and QoS/QoE tests running in the widespread probes, which indicates a lack of flexibility. The internet usage is improving fast; therefore, we should improve the quality of service measuring as well.

* + 1. **Project objectives:**

This work will be considered as a graduation project to obtain applied computer science diploma from the national engineering school of Sousse (ENISO).

The main project goal is to design and develop a platform for remote and communicating devices, to serve broadband monitoring and supervision. The application should satisfy these needs:

* The creation and interpretation of messages exchanged with terminals and embedded equipment.
* Information exchanges with client applications (northbound interface).
* Management of messages from terminals (southbound interface).
* Sending requests and messages to the terminals.
* On-the-fly supervision of exchanges.
* Management of supervised elements.
* Resolution of message referral rules.
  1. **Critical analysis of the state of the art:**

This step is essential to propose our solution in the relation to those offered by other companies. To do so, we have studied the characteristics and the features of some available apps while we focus on their weak points.

* + 1. **State of the art:**
       1. **Samknows One:**

The SamKnows One is a cloud-based analytics platform that includes a full range of measurement agents for fixed and cellular internet connection with a global test infrastructure. The SamKnows solution stores and visualizes performance data in real-time. This solution is implemented by a UK company “Sam” founded in 2008 by Sam Crawford. SamKnows One has developed a suite of tests as follows:

* Speed tests: includes download and upload over TCP and UDP speed tests.
* Latency, loss and jitter: latency, jitter and packet loss over UDP, latency over HTML5.
* DNS resolution: DNS resolution time and failure rate (UDP).
* Web browsing: web browsing test over TCP.
* CDN performance: content delivery network (CDN) measurements over TCP.
* Video streaming: video streaming measurements that stream real content from major video streaming providers.
* Gaming: measures performance for a number of major games.
* Online storage: tests upload and download from popular online storage services.
* Voice over IP: measures the quality of a voice call between client and test server.
* Traceroute: tests the path that traffic takes around the internet, it is useful in diagnosing routing issues.
* Data usage: measures of data used on the broadband connection.

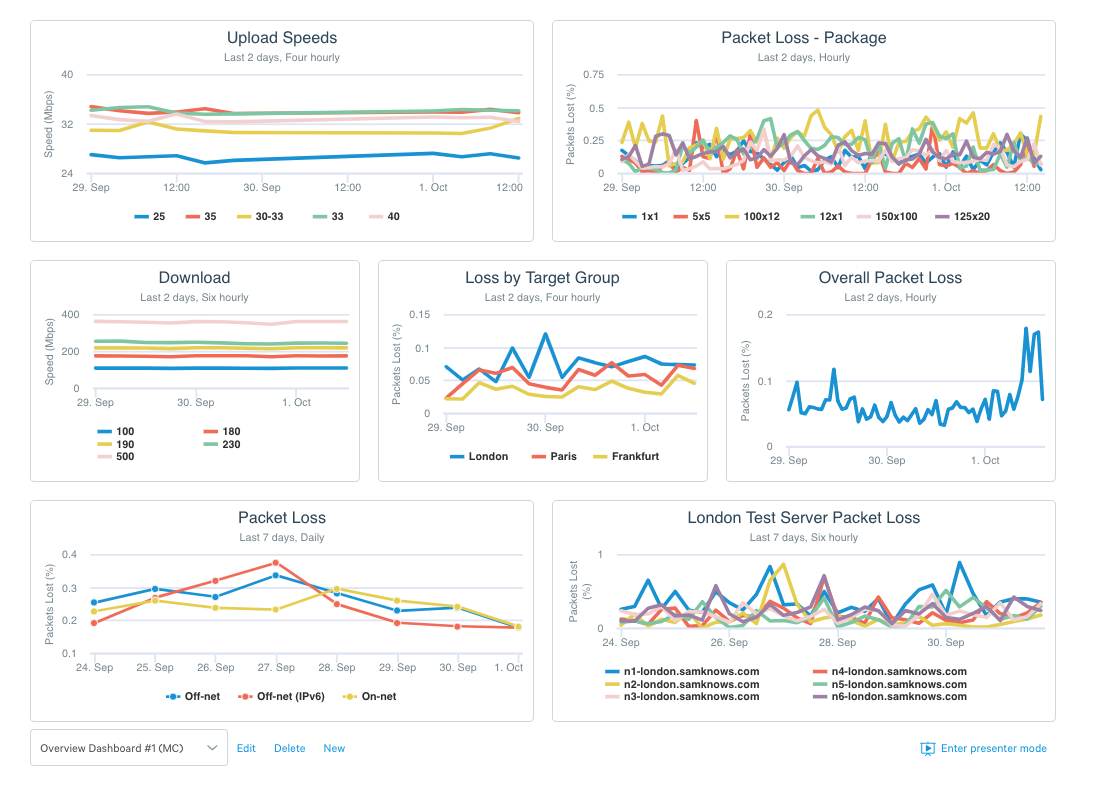


Figure 1.1: Screenshot of SamKnows One dashboard

* + - 1. **SMAQ:**

SMAQ is a solution implemented by Sofrecom to generate reports and analysis based upon different broadband tests, this solution is used by the Orient Middle East and Africa Orange affiliates.

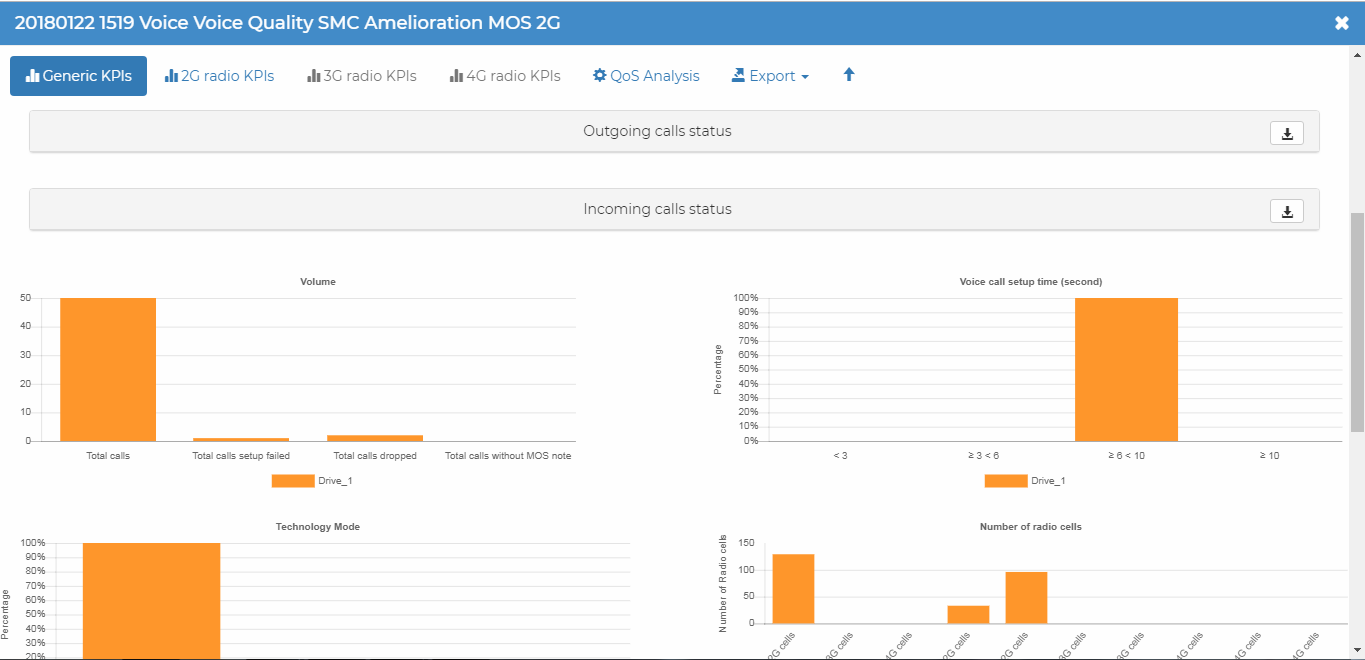


Figure 1.2: Sample of SMAQ online dashboard

* + 1. **Critical of the state of the art:**

The following table shows in detail the difference between the two mentioned solutions:

|  |  |  |
| --- | --- | --- |
| Solution | SamKnows One | SMAQ |
| Paying solution | YES | NO |
| Analytics | YES | YES |
| Custom dashboard | YES | NO |
| Mapping data | YES | YES |
| Generate reports | YES | YES |
| Network monitoring (alerting) | YES | NO |
| User management | YES | NO |
| Devices management | NO | NO |
| Tests management | NO | NO |

Table 1.1: Comparison of state of the art

Sofrecom Tunisia is looking for an open source solution for the issue in question so SamKnows One solution is not convenient for them, in the other hand SMAQ provides just the basic functionalities. The weak point in the mentioned solutions, they don’t give users the access to devices configuration. Network supervisors can’t customize tests to satisfy their specific needs.

* + 1. **Proposed solution:**

The solution proposed by Sofrecom Tunisia is to design and develop a platform for broadband monitoring and supervision, “SMAQ Probes”, the solution should respond the following needs:

* Online device management and task scheduling.
* Online test management.
* Online statistics and customized charts configuration.
* Online alerts configuration and customization.
* Online user management
  1. **Modeling language:**

During the work on our solution we used UML “Unified Modeling Language” for describing and modeling the specifications of our project. UML is a flexible and versatile modeling language, also it is the most popular and widely used by the community. We are going to present some diagrams from UML that we find it useful during our work:

* Use case diagram: it helps to structure the needs of users and the corresponding objectives of our system by identifying its users and their interactions.
* Sequence diagram: it is a time focus representation of objects and their interactions.
* Package diagram: it gives an overview of the application packages. It is a high abstraction that presents the application modularity.
* Class diagram: it gives a presentation of classes and interfaces of our system and relations between them.
* Activity diagram: it gives an overview of the dynamic aspects of the system.
  1. **Software development methodology:**

Before starting the project design and development, we should choose appropriate software development methodology to work with. The software development methodology helps to describe the different phases and the sequences of application development process.

During our project, we used agile kanban because it is a most convenient method to us. I am the only intern working on the project. Changes in the project can happen any time. We are continuously improving the flow of work. We are trying to limit work in progress and to maximize efficiency. Also, we focus on reducing the time it takes to take a project from start to finish.

* 1. **Conclusion:**

In this chapter, we presented the general context of the project by presenting the host company Sofrecom Tunisia, the problem statement and the state of the art.

In the next chapter, we will model the requirements of our solution through use case diagrams.

**Chapter 2: Requirements analysis and specification**

**2.1 Introduction:**

The requirements analysis and specification phase is an essential step for the development of a new application. It allows presenting the application’s features in detail.

In this chapter, the first part will be devoted to identify the different actors of the application who are interacting with the system and to give the functional and nonfunctional requirements definitions. Subsequently, we will present the general system analysis using use case diagrams.

**2.2 Identification of the actors:**

An actor is an abstraction of a role of actual user who is in a perpetual interaction with the application. Following on, our system’s actor along with his role and granted permissions.

* **Internal actors:**
* Application administrator: the administrator is responsible of managing users and their permission, he has the permission to create, read, update, and delete users. Also, he has the permission to check all the other configurations like devices configuration and test configuration.
* Network supervisor: the network supervisor has the permission to read the different configurations without editing them. He has the right to supervise the quality of experience “QOS”.
* Network and broadband administrator: he has all the rights of the network supervisor, in addition, he has the permission to create, read, update, and delete broadband monitoring configurations.
* **External actors:**
* Probe: the probe is the entity able to get devices and tests configuration, and send the metrics to the server after running tests.

**2.3 Functional requirements:**

Functional requirements refer to primary functions that each component of our solution must exhibit. It is a set of services which are:

* **For the web application:**
* The application should give the administrator the hand to manage users’ accounts.
* The application should give permitted users the possibility to customize their dashboards.
* The application should give permitted users the access to manage the network monitoring (alerting).
* The application should provide permitted users with the access to tests and devices configurations.
* The application should be able to process received metrics data in real-time.
* **For the hardware:**
* The boards should be able to receive and implement their configurations in real-time.
* The boards should be able to run tests according to the time scheduler and send results to the server.
* The boards should be able to send their current configuration (location, IP address, device identifier, jobs configuration).
* The boards should be able to keep the tests results if the server is not available.

**2.4 Non-functional requirements:**

Non-functional requirements refer to several key features that are beyond the purpose of the solution, they specify criteria that judge the operation of a system, rather than specific behaviors in order to ensure the client’s satisfaction.

* Extensibility: the system must be open to some extension like for example adding new features if needed without radical modification in the code.
* Performance: The web application should be as efficient as possible with especially a good response time. Users should be able to receive the quality of experience (QOE) from the cloud server within a reasonable amount of time.
* Re-usability: The system shouldn’t be exclusive for our case and must be adaptive to other use cases.
* Robustness: The system must cope with errors during execution and should be able to reboot within a short time in case of failure.
* Security: The user’s personal information must be kept safe from others and only system administrator has permissions to access it. The broadband monitoring and supervision access must be permitted to the supervisors of the network in question.

**2.5 Requirements analysis:**

On one hand, this section offers a better understanding of the mentioned requirements by declaring them in a semi-formal way. On the other hand, it emphasizes the interactions between the actor and our application. In contemplation of breaking down the complexity of these goals, we use the use case diagrams.

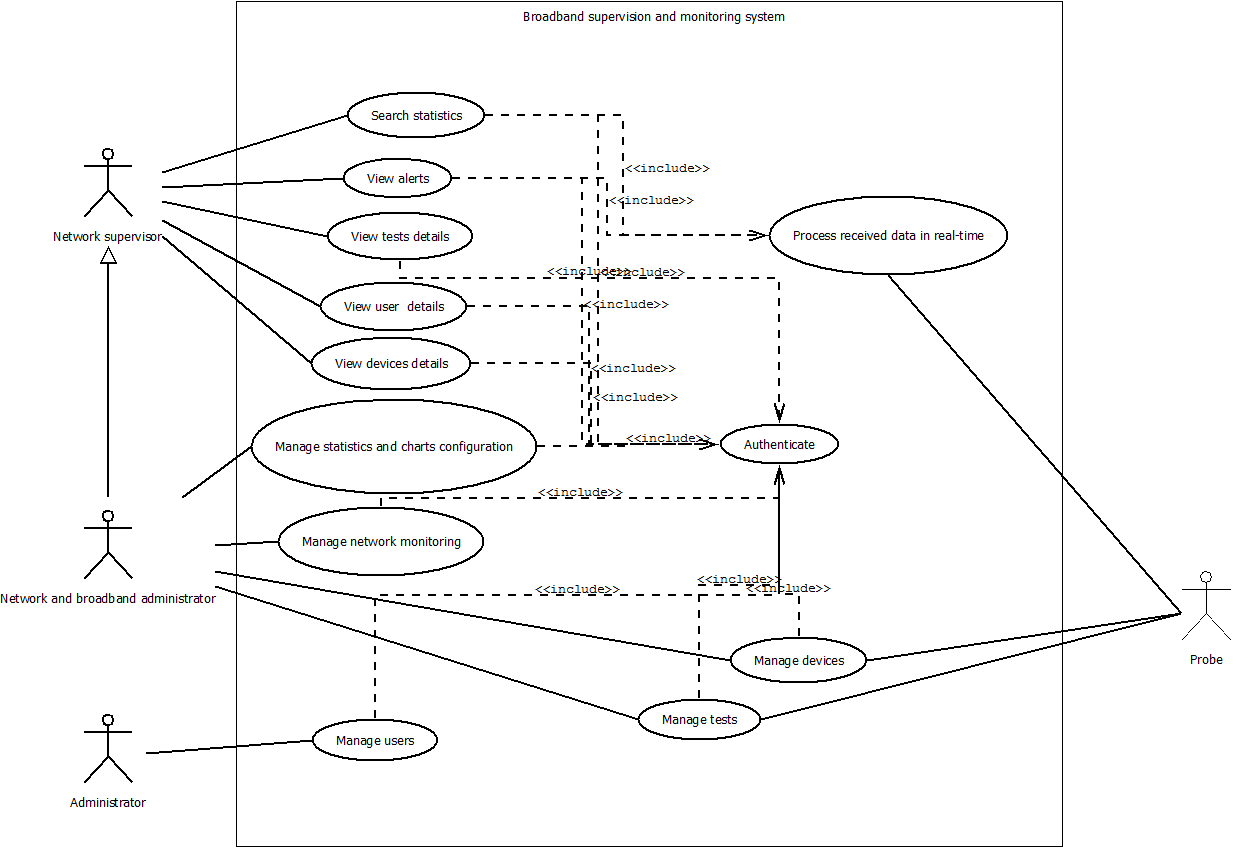
****

Figure 2.1: general use case diagram

As shown in the general use case diagram, only the administrator can register all types of users. All the features of the application must go through authentication. The network and broadband supervisor is responsible for managing network monitoring, devices configuration, tests, and statistics configuration. Any configuration that concerns probes configuration will be sent to probes. Probes send their information and runs tests according to a job scheduler, then probes send tests results to the server, thus, our system process the received data in real-time. Finally our system is prepared to generate statistics and quality of service for the network supervisor.

**2.5.1 Manage devices:**

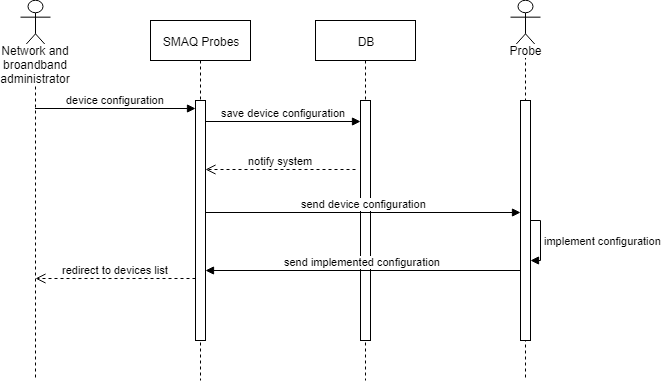
****

Figure 2.2: Devices management sequence diagram

|  |  |
| --- | --- |
| Title | Device Management |
| Author | Ismail MEKNI |
| Version | 1.0 |
| Objectives | Allow users to manage connected devices configuration |
| Actors | Network and broadband administrator – SMAQ Probes – Probe |
| Pre-conditions | The user should authenticate as network and broadband administrator.  The device should be connected. |
| Post-conditions | New device configuration is persisted in the database.  New device configuration is implemented in the probe. |
| Story | 1. The user enters device new configuration (status, IP address, client name, job scheduling). 2. The user submits the changes. |
| Alternative story |  |
| Exceptional story | The device in question is not connected; the configuration’s message will be suspended waiting the device to reconnect. |

Table 2.1: Device management description

Device management functionality is permitted to network and broadband administrators. User can access to the devices list. User can select a device to edit its configuration (IP address, location, registered client name, job scheduling), if user submits the new configuration, the configuration will be sent to the backend system to persist it to database. Also the configuration will be sent to the device in question, the probe, device, implements the changes. Finally, the probe sends back the implemented configurations to the system as a confirmation.

**2.5.2 Manage tests:**

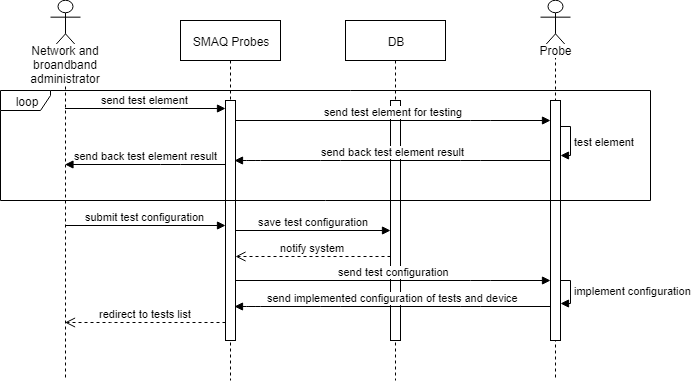


Figure 2.3: Tests management sequence diagram

|  |  |
| --- | --- |
| Title | Tests Management |
| Author | Ismail MEKNI |
| Version | 1.0 |
| Objectives | Allow users to manage tests configuration |
| Actors | Network and broadband administrator – SMAQ Probes – Probe |
| Pre-conditions | The user should authenticate as network and broadband administrator.  At least one device should be connected. |
| Post-conditions | New tests configuration is persisted in the database.  New tests are running on the probes. |
| Story | 1. The user tests each element directly on the device. 2. The user submits the test configuration with all elements. |
| Alternative story |  |
| Exceptional story | There is no connected device, so user can’t test elements, the operation with be suspended until at least one device reconnect. |

Table 2.2: Test management description

The access to the test configuration is granted to users with network and broadband administrator role. To edit test configuration, user should enter test elements, each element must be tested directly on the probe, device, and then test’s element result will be sent back to the user. After creating and testing all the elements, a new test configuration will be sent to the backend system. The configuration is persisted to database. The new test configuration is sent to the probes, all the probes implement the new test configuration. Finally a signal messages is sent from all the probes holding the current configurations.

**2.5.3 Manage network monitoring:**

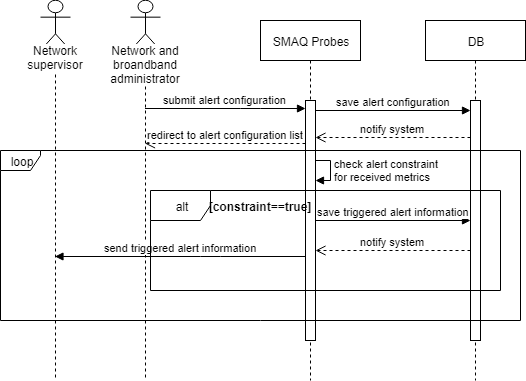


Figure 2.4: Network monitoring management sequence diagram

|  |  |
| --- | --- |
| Title | Network monitoring management |
| Author | Ismail MEKNI |
| Version | 1.0 |
| Objectives | Allow users to manage network monitoring configuration, alerting system |
| Actors | Network and broadband administrator – Network supervisor – SMAQ Probes |
| Pre-conditions | The user should authenticate as network and broadband administrator to access the network monitoring management.  To view triggered alerts, the user should authenticate as a network supervisor. |
| Post-conditions | New alert configuration is persisted in the database.  Alert listener is running on the received metrics. |
| Story | 1. The user send alert configuration containing the constraint 2. The network supervisor can view alerts if triggered. |
| Alternative story |  |
| Exceptional story |  |

Table 2.3: Network monitoring management description

The user authenticates with network and broadband administrator. This feature aims to configure customized alerts. User creates alerts with a specific constraint. This configuration will be persisted to database. An alert checker will be run for every received metrics data, if the constraint is satisfied an alert with full description will be triggered. The triggered alerts are persisted to database so network supervisors can check them.

**2.5.4 Manage statistics and charts configuration:**

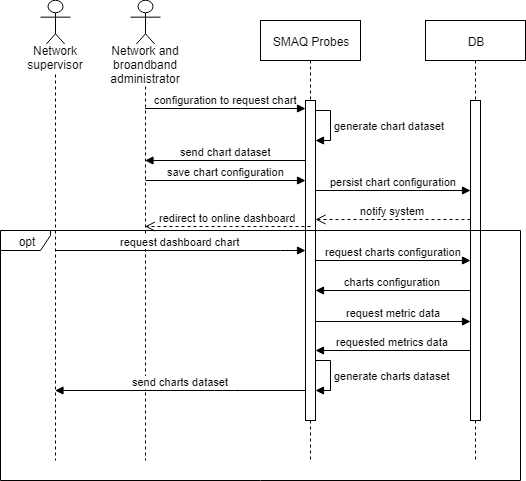


Figure 2.5: Charts management sequence diagram

|  |  |
| --- | --- |
| Title | Charts and statistics management |
| Author | Ismail MEKNI |
| Version | 1.0 |
| Objectives | Allow users to manage charts configuration |
| Actors | Network and broadband administrator – Network supervisor – SMAQ Probes |
| Pre-conditions | The user should authenticate as network and broadband administrator to access charts and statistics management.  To view dashboard, the user should authenticate as a network supervisor. |
| Post-conditions | New chart configuration is persisted in the database.  New chart is added to dashboard. |
| Story | 1. The user send chart configuration, system generates chart dataset. 2. Chart displayed to user. 3. User submit chart configuration. |
| Alternative story |  |
| Exceptional story |  |

Table 2.4: Charts management description

To access to charts and statistics configuration, users should have network and broadband administrator privileges. User enters the chart parameters; our system generates the chart in question. User submits this configuration to be persisted and added to dashboard. Thus, users with network supervisor permission can see the configured charts on the dashboard. This feature aims to allow users to create customized charts and views.

**2.5.5 Manage users:**

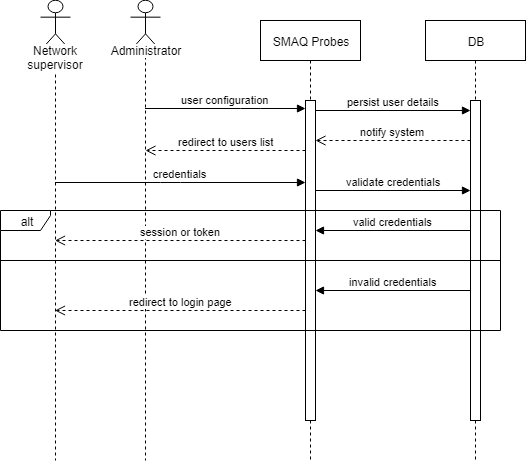


Figure 2.6: Users’ management sequence diagram

|  |  |
| --- | --- |
| Title | Users management |
| Author | Ismail MEKNI |
| Version | 1.0 |
| Objectives | Allow administrator to manage users’ configuration and details. |
| Actors | Administrator – Network supervisor – SMAQ Probes |
| Pre-conditions | The user should authenticate as application administrator to access the users’ management. |
| Post-conditions | User credential is persisted to database.  User can authenticate. |
| Story | 1. The administrator submits user configuration. 2. The user submits his credentials. |
| Alternative story |  |
| Exceptional story | If the user enters invalid credentials, he will be prompted to try to login again. |

Table 2.5: Users management description

The users’ management feature is only allowed to the application administrator. The administrator enters the credentials of each user. Thus, user is now registered to the application and he can access to the application features according to his privileges. To sign in to the application, the user enters his credentials, generally a username and a password, if the credentials are valid, he will be redirected to the dashboard, and else he will be prompted to login again.

**2.6 Conclusion:**

Throughout this chapter, we specified and analyzed the requirements that solution should deliver to users, and we presented the main scenarios and the use cases that it should offer.

The next chapter aims to go a step further in the process of developing the application via presenting the design of the different components of our system.

**Chapter 3: Design of the physical and logical architectures**

**3.1 Introduction:**

In order to reach the appropriate result as described in the specifications, we need to clarify the project’s main architecture as well as the architecture of its components. This chapter will focuses on designing a suitable structure for the smart parking system. This step is considered as the most crucial of the process because it prepares the ground for the implementation phase.

**3.2 Physical architecture:**

The architecture presented in figure 3.1 is the physical architecture of our system.

It represents the physical layout of our system and its components in a global diagram and it refers to some representations of the structure or organization of the physical elements that build the system.

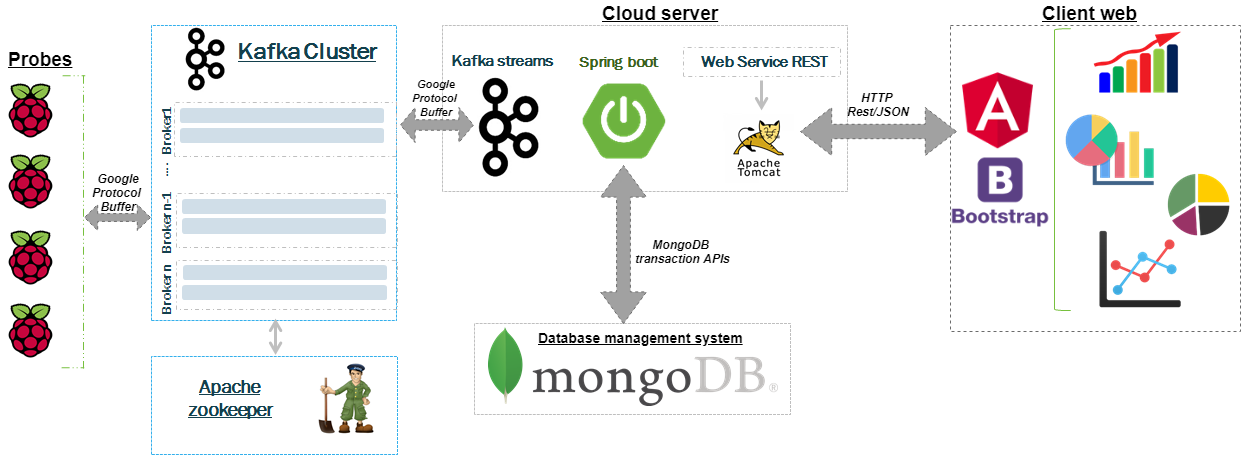


Figure 3.1: Physical Architecture of SMAQ Probes solution

This architecture describes the main components of the system and how they interact in order to achieve the objectives mentioned in the previous chapter.

The system is composed mainly from the following parts: the probes (Raspberry Pi boards), the user interface (web browser) and the cloud server including Kafka cluster, the database management system and our web application.

The probes represent the entity that executes the scheduled tests and sends the metrics to Kafka cluster through the Google protocol buffer.

The client web part represents the part with which the final users interacts and they are essentially: the Web platform accessible by the application users, application administrator, network supervisors and network and broadband administrators.

The third part, the cloud server, is where the application will be hosted, this part is responsible for receiving and processing data coming from Kafka cluster, also it is responsible for data analysis and configurations persistent to our database.

**3.3 Logical architecture:**

**3.3.1 Conceptual model:**

The figure 3.2 shows the logical architecture of the system.

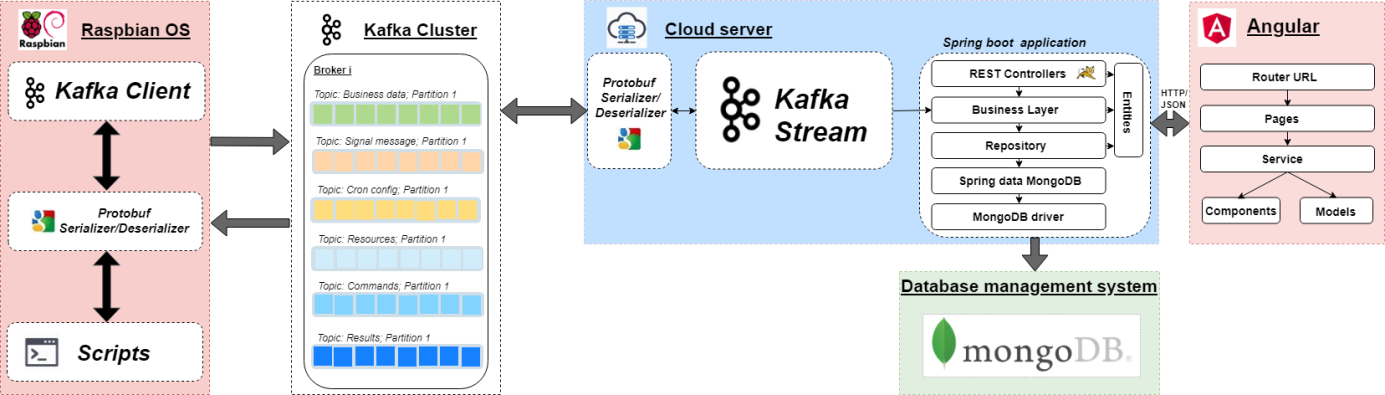


Figure 3.2: Logical architecture

According to the figure of the logical architecture, the system is composed from three major parts:

* **Probes:** this parts represents widespread devices, these devices are the entities that handles tests. There are several scripts responsible for running the tests with efficient schedules. All the messages are serialized with Google Protocol Buffer. There is a Kafka client responsible for publishing and receiving messages.
* **Cloud server:** this server is the entity that holds the application logic. It holds within him a Google Protocol Buffer converter. Kafka stream is the layer that process the received data in real-time with high performance. Spring Boot application is a three tiers web application, it is responsible for managing the features of our system including metrics analytics.
* **Angular:** this entity is the frontend application accessible to users across the Web.

The connection between the web application and the user interface is guaranteed through HTTP protocol and REST web services.

Kafka cluster is playing the role of a middleware between the probes and the backend server.

Finally we have our database, we have chosen MongDB as a database management system for performance reasons, and it is accessible only through the three tiers web application.

**3.3.2 Modular decomposition:**

**3.3.2.1 Class diagram of entities:**

The figure 3.3 shows the class diagram of our system, this diagram summarizes relationships between our entities in the database.

This class diagram is efficient in the case of MongoDB database management system. MongoDB is an oriented document database management system.

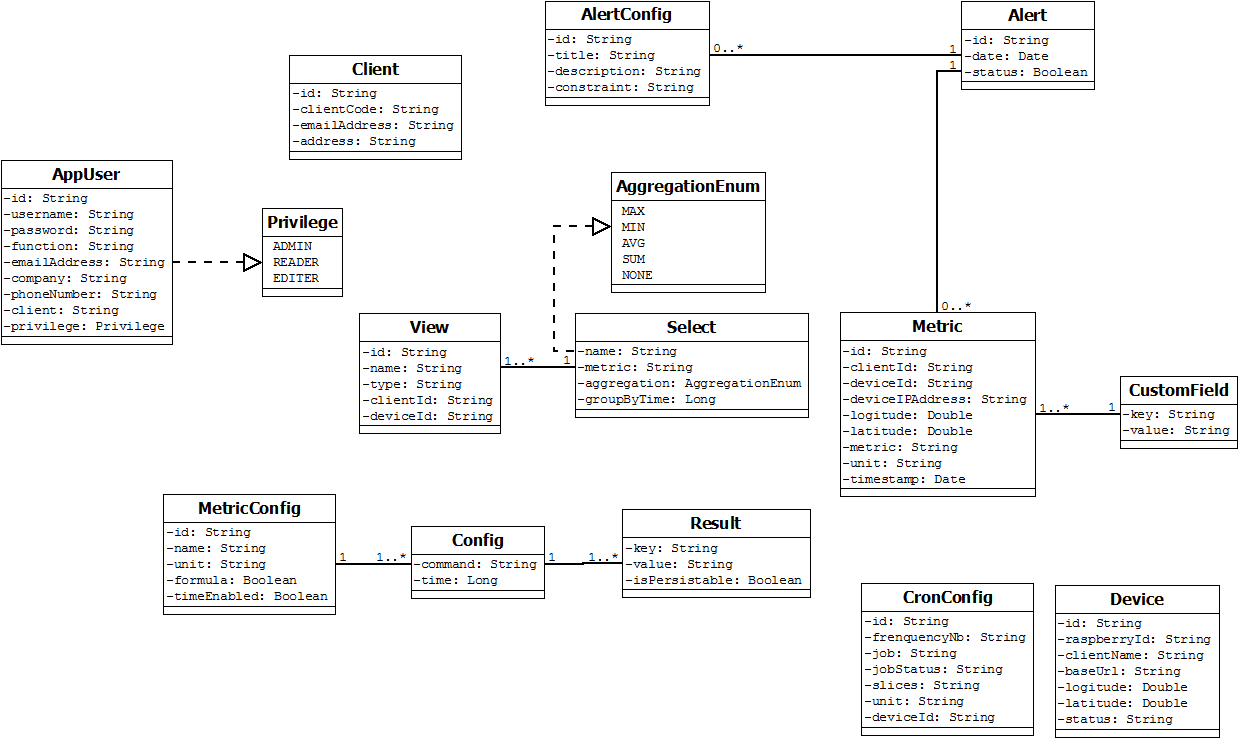


Figure 3.3: Database entities class diagram

**3.3.2.2 Package diagram:**

The package diagram is a static view that serves to globally describe the different components of the application. The figure 3.4 presents the package diagram of our solution to have an overview its different elements.

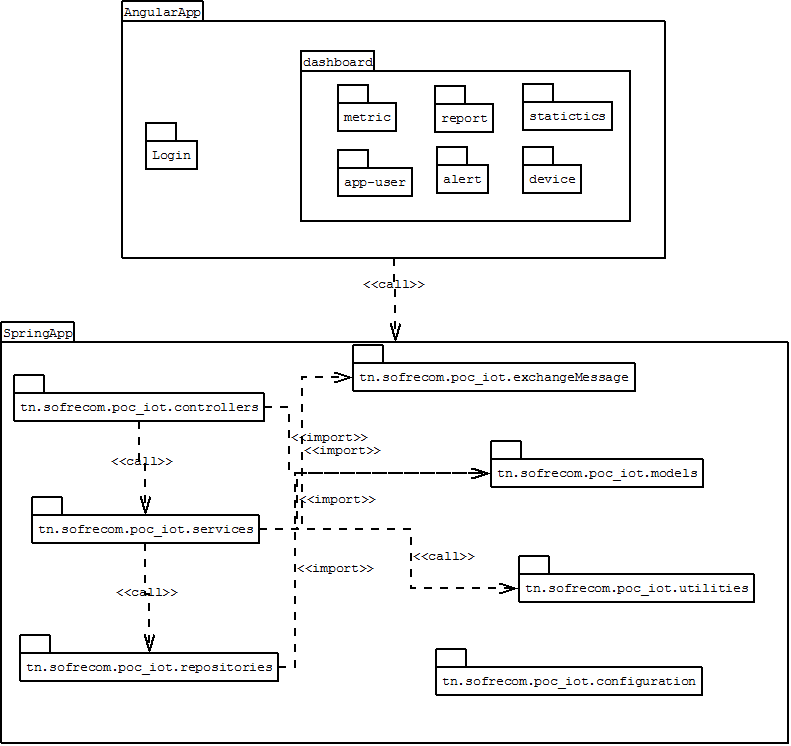


Figure 3.4: Package diagram of the backend application

**3.3.2.3 Devices management model:**

This module describes devices management process. This module is about editing, deleting, and checking available devices. There is no meaning of creating device because the device information will be sent by the device if it is connected.

The following class diagram figure 44 presents the involved classes and components that build the module.

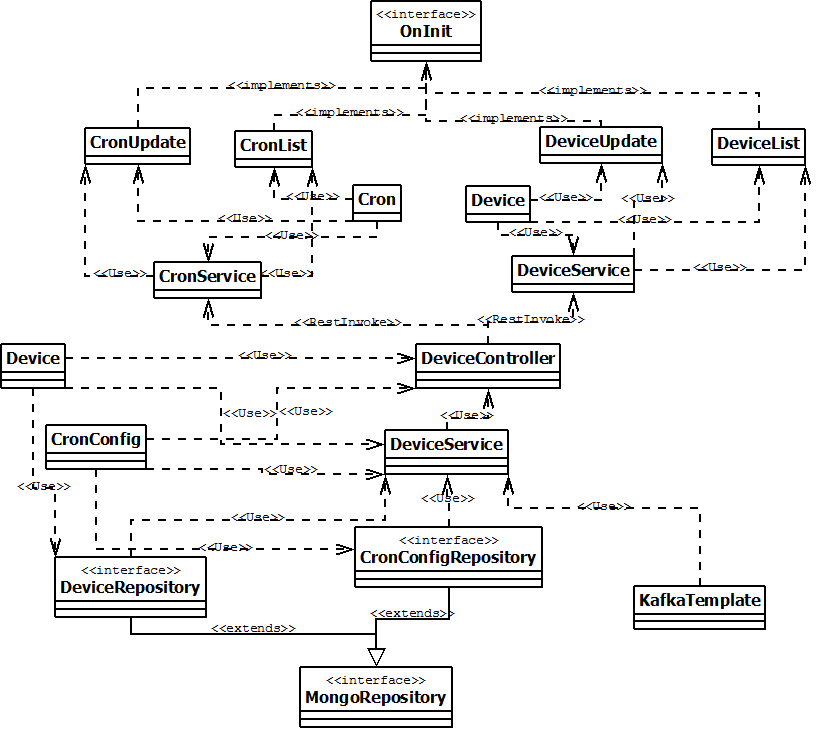


Figure 44: Class diagram of the devices management module

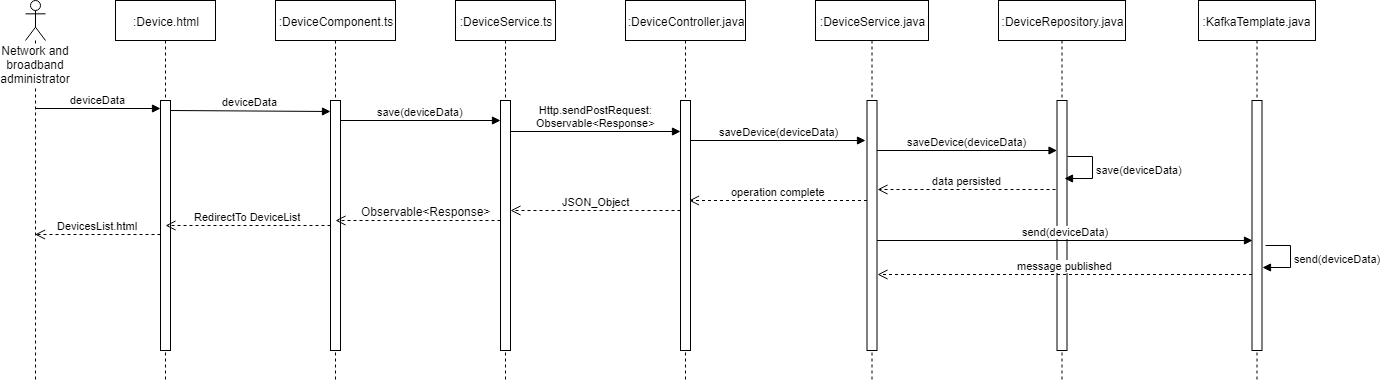
Then, we have the sequence diagram that describes the interaction between the module components; this diagram presents the update device operation:  


Figure 33: Sequence diagram of device updating operation

The diagram below is the sequence diagram for the update Cron operation. Cron is a job already running on the probe according to a specific schedule.

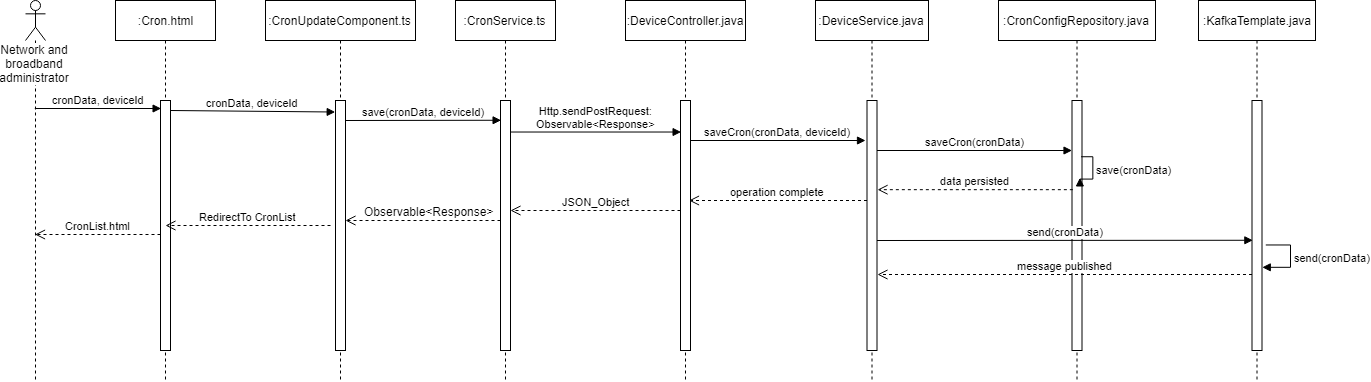


Figure 33: Sequence diagram of Cron updating operation

The device configuration consists of the device general information (IP address, device ID, location…) and the jobs running on the device in question. First, the user fills the form of device information, after, he submits this information. Using REST web services we deliver the new configuration from frontend application to backend side. Finally, the backend save the new configuration to our database and send it to the device in question in order to consider the new changes.

For Cron configuration, we follow the same process; the user submits the new configuration, after, this configuration will be sent to backend in JSON objects forma. Finally, our backend application implements the changes in the database and the device.

**3.3.2.4 Metrics and tests management module:**

This module describes tests management process. This module is about creating, editing, deleting, and checking available devices.

The following class diagram figure 44 presents the involved classes and components that build our module.

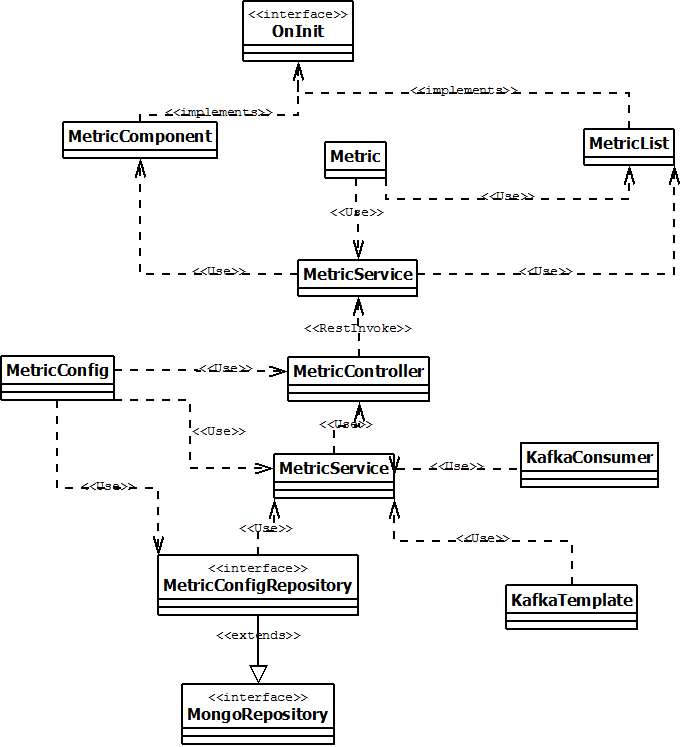


Figure 44: Class diagram of the tests and metrics management module

Then, we have the sequence diagram that describes the interaction between the module components; this diagram presents the creation and updating tests operations:

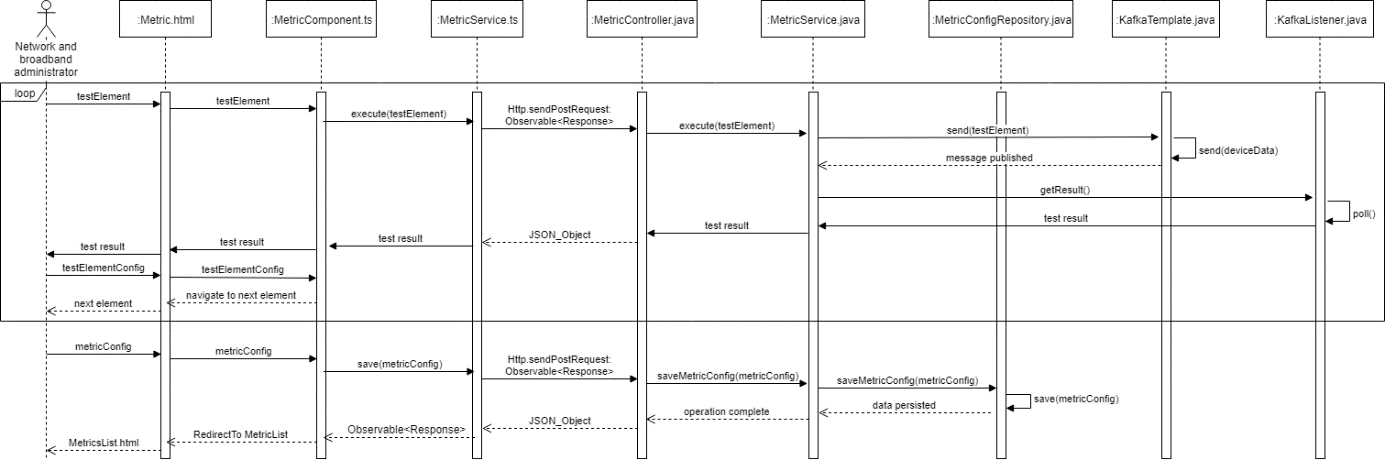


Figure 33: Sequence diagram of metrics and tests creation and updating operation

The metrics and tests configuration consists of the describing the general test information (test name, test unit, schedules) and the test elements configuration. A test element is a command line or even a script. To achieve better performance, each test element will be tested directly on the Raspberry board going through the frontend application to the backend application to reach our Kafka cluster, the element will be tested and the results will be sent back to Kafka, finally the results data will takes back the same path to reach the user interface. The user configures the element with its results. To finish, the user submits the entire test’s configuration to the web application.

We must note that the updating and creation operations are the same because we are using MongoDB as a database, if the object is already in the database, MongoDB will edit it, and else the object will be created.

**3.3.2.5 Network monitoring management module:**

This module describes network monitoring management process. This module is about creating, editing, deleting, and checking available alerts information and constraints.

The following class diagram figure 44 presents the involved classes and components that build network monitoring module.

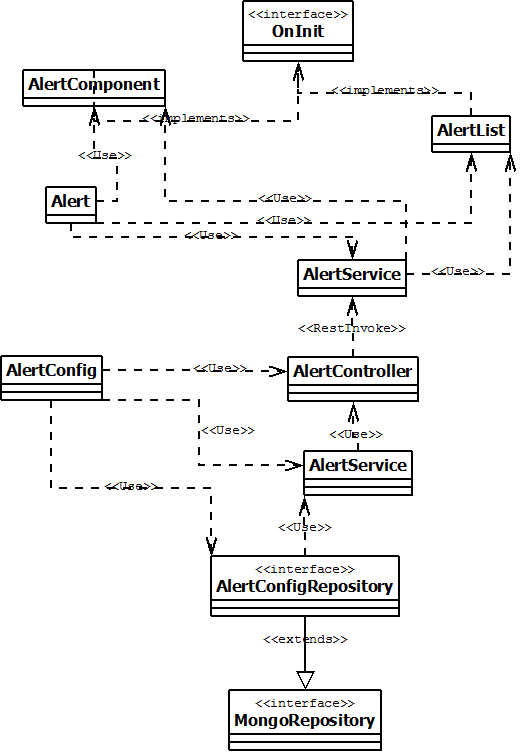


Figure 44: Class diagram of the network monitoring management module

Then, we have the sequence diagram that describes the interaction between the module components; this diagram presents the creation and updating alerts operations:

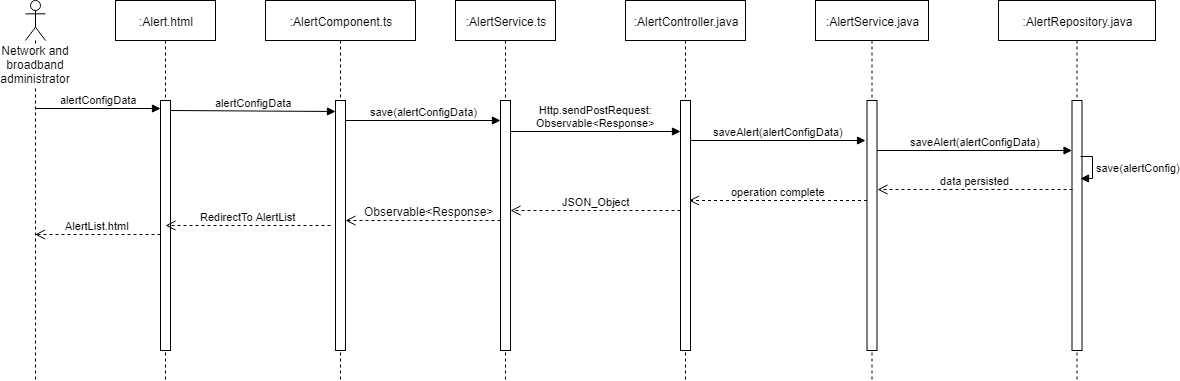


Figure 33: Sequence diagram of alerts creation and updating operation

The network monitoring configuration consists of creating customized alerts to help network supervisors. The network and broadband administrator fill the form that contains alert information (alert title, alert description, involved test, alert constraint) through the graphic user interface. The alert constraint is a logic expression that defines when the alert will be triggered. The configuration is sent from the frontend to backend with REST web services. The web application saves the alert configuration. The other details about the network monitoring concerning alert triggering and supervision will be explained and clarified in a separate part.

**3.3.2.6 Statistics and charts management module:**

This module describes statistics and charts management process. This module is about editing, deleting, and checking available charts and views.

The following class diagram figure 44 presents the involved classes and components that build the module.

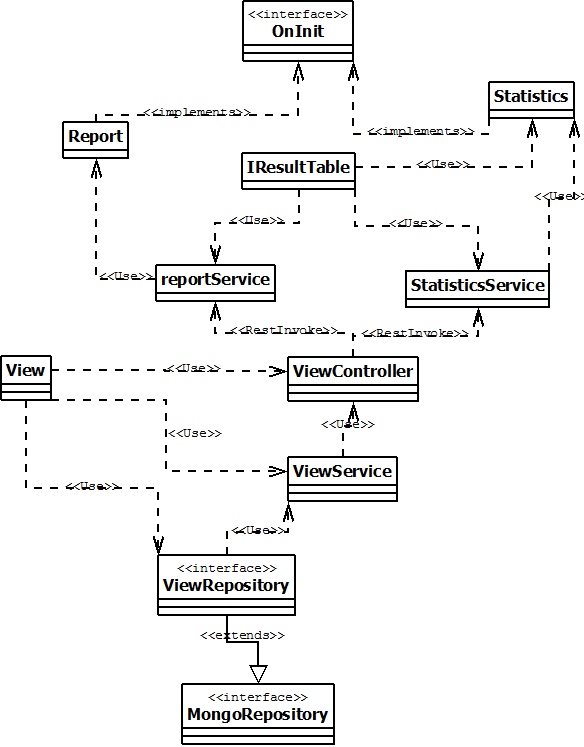


Figure 44: Class diagram of the statistics and charts management module

Then, we have the sequence diagram that describes the interaction between the module components; this diagram presents the creation views operation:

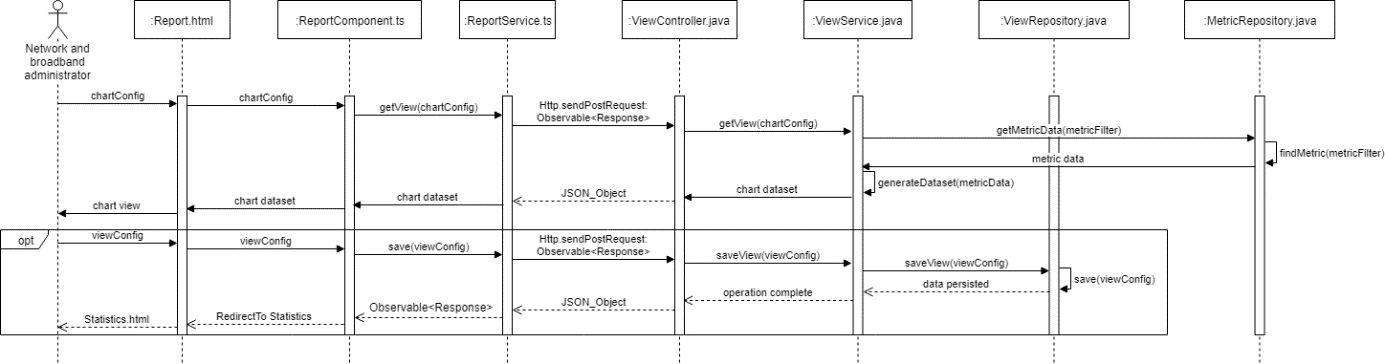


Figure 33: Sequence diagram of views creation operation

The diagram below is the sequence diagram for dashboard generation operation.

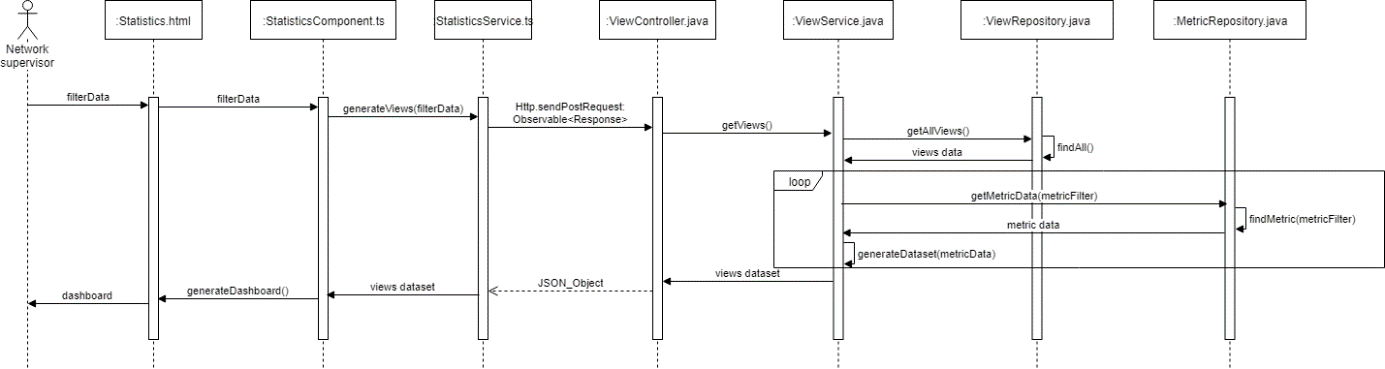


Figure 33: Sequence diagram of dashboard operation

The statistics and charts configuration consists of creating a model carrying a view configuration that we can use it to generate the same view each time. The statistics and charts management module contains two parts, the first one is the views configuration and the second one is the dashboard generation.

To achieve creating customized views, from the presentation layer, Angular application, the user put the configuration he needs to generate the chart. This configuration goes through REST web service to the backend controller, then to the service layer. The service layer will take the view configuration as parameter to fetch the metrics data, stored in MongoDB, and process the received data in order create view dataset. The chart dataset goes through REST web service again to reach the frontend application arriving to the user interface. At this point, the user has the option to add this view to the dashboard, so the data.

The second part is the dashboard generation, when the user navigate to dashboard, there are three filters, time filter, location filter, and test filter. A request will be sent from the user interface and the Angular component and service to the backend through REST web service carrying the filters inputs. Views are fetched according to the test filter, and then the metrics data is fetched according location and time filter. The metrics data is processed to generate dataset for each view in the dashboard. Finally, the charts datasets goes back to the user interface.

**3.3.2.7 Authentication and users management module:**

This module describes authentication and users accounts management process. This module is about creating, editing, deleting, and checking available users’ information.

The following class diagram figure 44 presents the involved classes and components that build our module.

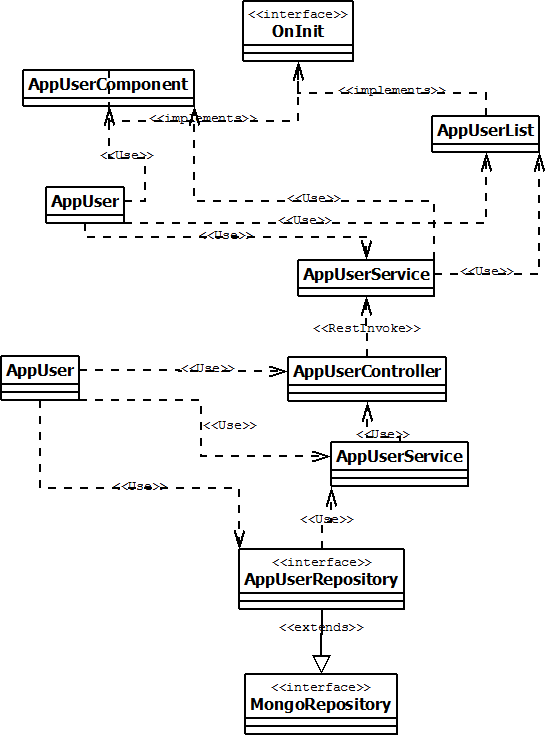


Figure 44: Class diagram of the authentication and users management module

Then, we have the sequence diagram that describes the interaction between the module components; this diagram presents the creation and updating users’ information operations:

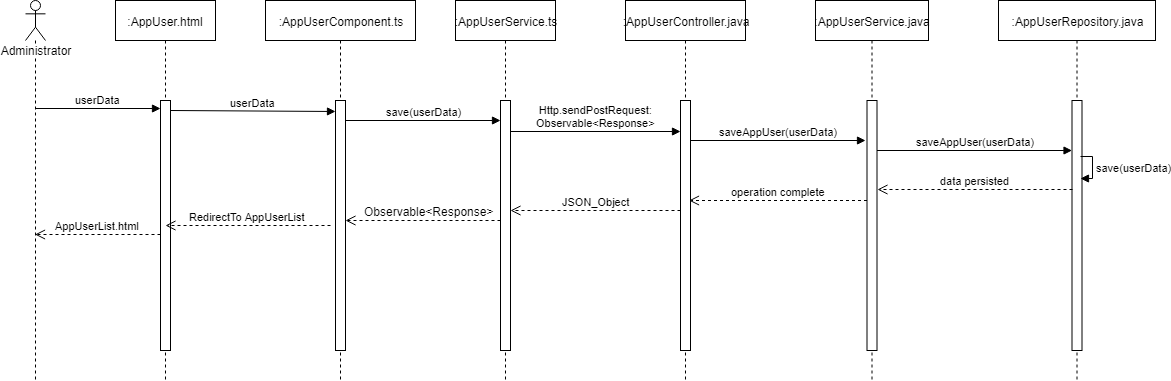


Figure 33: Sequence diagram of users’ creation and updating operations

The user configuration consists of creating a personalized user account that holds a username, a password, and a role or privilege.

The administrator just needs to fill the form with the user information. After finishing, a request is sent to the backend application in order to persist the user information. At this point, the user is able to authenticate.

**3.3 Broadband supervision and monitoring theory:**

In the previous parts of the report we talked about the features offered by our solution concerning analytics and components management, although we did not emphasize the real-time network monitoring and metrics data processing. Also we need to clarify how the probes are handling tests and received configurations.

**Tests and jobs handling between probes and Kafka:**

We are using Kafka as mediator between the server and the probes. For every reboot, devices send their initial configuration to Kafka, in the counterpart, the devices receives their configurations, these configurations holds within it jobs and tests. The embedded application schedules the jobs using Crontab. There a listener implemented on the boards in order to wait for configuration changes. The following figure 55 illustrates the interactions between a probe and Kafka.

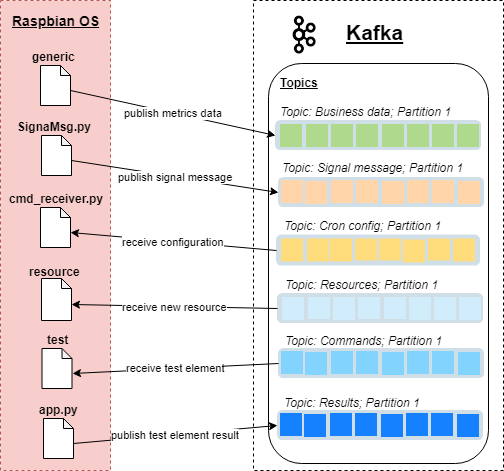


Figure 55: Device and Kafka interactions

**Real-time processing and network monitoring:**

As we mentioned earlier, after setting up the tests and alerts configurations, we have real-time data processing, this point is the crucial point of our solution. This part can decide the level of performance of the system. To achieve this task, we chose Kafka Stream.

Kafka Stream is a Kafka client that can handles real-time processing on data coming from a Kafka cluster or broker. Kafka stream subscribes to Business data topic, for each received test result, Kafka Stream brings the alerts configuration concerning the test in question, after, it tests the alerts constraints on the received data, if there is a satisfied constraint, it triggers a specific alert.

Another mission for Kafka Stream is to parse the received data in order to extract the metrics values to store it to MongoDB. However, between data parsing and storage, there is a specific processing for specific tests. In order to explain that, we should know that defining the quality of experience needs to have measurements from different timestamps with specific periods of time. To conclude, a complex calculation is required before storing data to have a better idea about the quality of experience.

Another crucial point in our solution, we need to have a high performance data analytics, MongoDB is an oriented document non-relational database that supports well indexing technique. MongoDB provides advanced querying operations that we used to perform our analytics with customized filters.

**3.4 Conclusion:**

Through this chapter, we described each part of the solution, its functionalities both separately and when coordinating with other parts of the system. We also explained subsequently the choice of our logical and physical architecture. Concerning the detailed design, we exhibited the class and sequence diagram. In the next chapter, we present and expose the technologies employed during the process of the creation of our product.

We studies new technologies

Manage user configuration and details.

**Chapter 4: Project achievements**

**4.1 Introduction:**

In this chapter, we will discuss the process of implementing the different parts of the system. We start by presenting the different tools both software and hardware used in every task in order to complete the implementation process.

**4.2 Developing environment:**

**4.2.1 Hardware environment:**

To achieve our project, we have used a DELL computer with a Windows 7 operating system. The characteristics of the used computer are provided below:

* CPU: intel i5 2.3 Ghz
* RAM: 8 GB
* Hard Disk Drive: 500 GB

For the probes, we have used Raspberry Pi 3 B+ model with an embedded Raspbian operation system. The characteristics of the board are provided below:

* CPU: 64-bit quad-core ARM v8
* RAM: 1 GB
* Memory card: 16 GB

**4.2.2 Software environment:**

In this part, we list the software programs and applications we used throughout the development of our system.

* Netbeans IDE 8.2: Netbeans is an integrated development environment developed by Oracle. Netbeans can be used with many programming languages; in our case we used it with Java programming language. Netbeans put in our hand many features to help developers achieving better coding performance.
* Visual studio code: Visual studio code is MIT open source licensed software, developed and maintained by Microsoft. It supports many programming languages with many different technologies just by integrating plugins within it. We used Visual studio code for Angular development. This software supports different platforms, Linux operation system, Windows, MacOS.
* Code Blocks IDE: Code Blocks is an integrated development environment for Fortan, C and C++ programming languages. We used it for C programming.
* MongoDB: MongoDB is a non-relational database management system based in documents. It provides many advanced features like indexing and advanced operations algorithm to query on the documents. To generate analytics with high performance, we need a database that supports well indexes; also we need a database that provides different and flexible querying techniques like map and reduce algorithm. So the better choice for us was MongoDB.
* Apache Tomcat web server: Tomcat is a web server that supports Java web logic. In our case used an embedded Tomcat web server.
* Postman: Postman allows users to execute and build personalized HTTP requests; to achieve this Postman provides many optional features.
* Thonny: Thonny is an integrated development environment for Python programming language running on Raspbian operating system. We used it to develop our Python scripts.
* Geny: Geny is an integrated development environment that supports many programming languages. This software is running on Raspbian operating system. We used it to develop C programs on RaspBerry Pi board.

**4.2.3 Frameworks and technologies:**

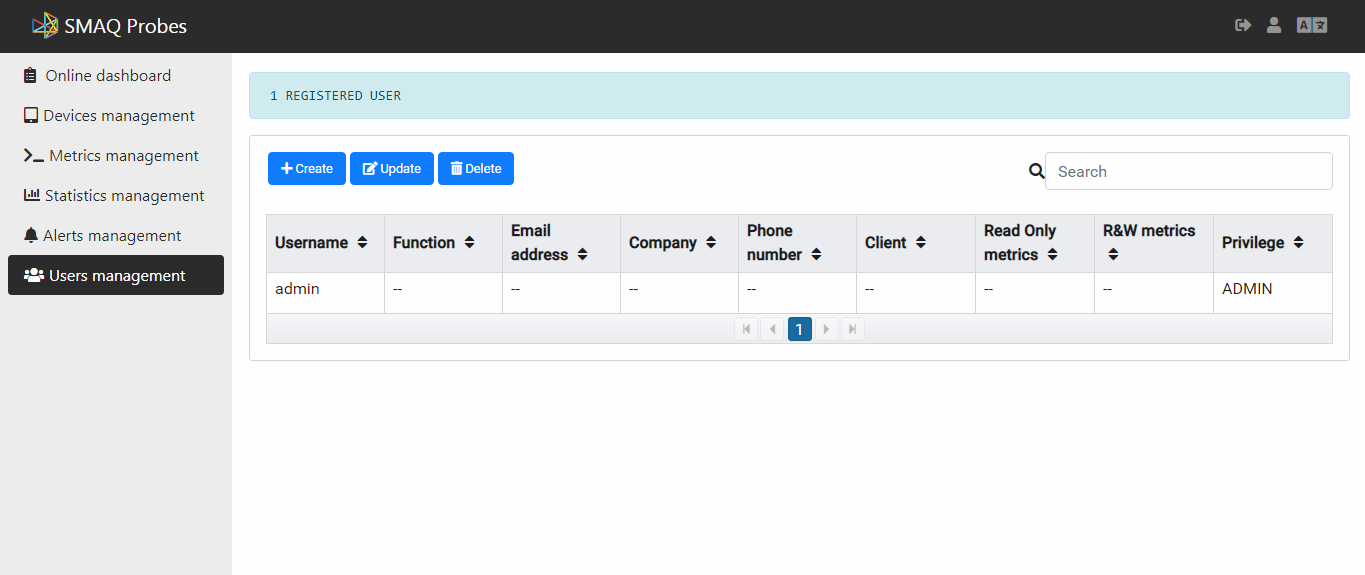
In this section, we discuss the technical choices we made to achieve our final product. We start by presenting the programming languages used in the development of the project. Afterwards, we defend our choice for the frameworks we used.

* **Programming languages:**
* Java: Java is a general purpose oriented object programming language. We used it to develop the backend of our solution.
* Typescript: Typescript is a programming language developed and maintained by Microsoft. Typescript is an object oriented programming language. We used it to develop our frontend application with Angular.
* Python: Python is a general purpose, high-level programming language. We used it in an embedded environment with Raspberry Pi boards.
* C: C is a procedural and a general purpose programming language. We used it in an embedded environment, with Raspberry Pi boards.
* Shell script: Shell script is a command line interpreter. We used it to install our embedded application.
* **Frameworks and technologies:**
* Spring framework: Spring is a Java application framework. Spring allows users to create enterprise services with POJO (Plain Old Java Objects). Spring uses dependency injection technique; also it provides many application configuration features.
* Spring data MongoDB: Spring data MongoDB is a part of Spring data project which aims to provide a Spring-based APIs (Application Programming Interfaces) for new datastores such as MongoDB. It gives the possibility to execute complex queries on MongoDB, especially with Mongo Template.
* Angular 7: Angular is an open source framework developed by Google. Angular is used for frontend application development, it gives the possibility to handle dynamically user interface. Angular is written entirely in Javascript, although it used Typescript as programming language.
* Apache Kafka: Kafka is distributed, fault-tolerant, horizontally scalable, wicked fast streaming platform. It is used for building real-time data pipelines and streaming applications with publish and subscribe technique. We used it as a middleware between widespread probes and the backend application.
* Kafka Stream: Kafka stream is a client library for building applications where the input and output data messages are stored in Kafka cluster or broker. It allows run real-time processing on the data. We used it for real-time processing.
* Google protocol buffer: Protocol buffer is a protocol for structured data serializing, Protocol buffer, also known as Protobuf, supports several programming language such as Java, Objective-C, Python and C++.We used it to on the messages coming from the probes. We used this protocol because it is faster than the ordinary data format like JSON.
* OAuth 2.0: OAuth is a protocol for authorization flows for web application. It is simple for developers to use. We used it for security issues in our web application.
* Spring Security: Spring Security is a framework to build applications with powerful and highly customizable authentication and access control.
* Linux Crontab: Crontab is a tool for job and task scheduling on Linux operating system, in our case Raspbian. We used it to schedule test running.
* MXparser: MXparser is a library for mathematical expressions parsing and evaluating. We used it with application features that need formula parsing such the case of alerting constraints.
* Bootstrap: Bootstrap is an open source frontend library created by Twitter for developing with HTML, CSS and Javascript in order to build responsive, mobile-first projects.

**4.3 Achieved work:**

**4.3.1 Authentication and user management:**

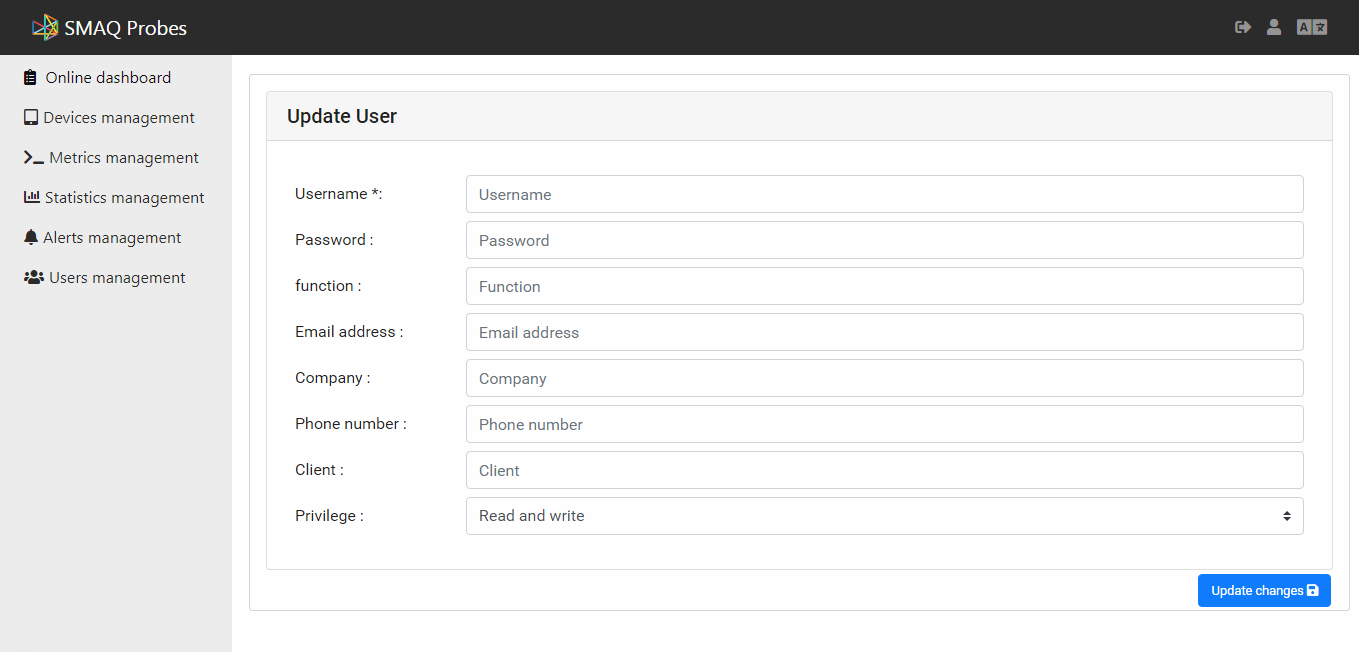
The figure 55 shows the users list with their information; this screen is only accessible from an administrator account:



When installing the application for the first time there is a default administrator account. With this account, the application administrator can create the accounts for the network supervisors and the network and broadband administrators.

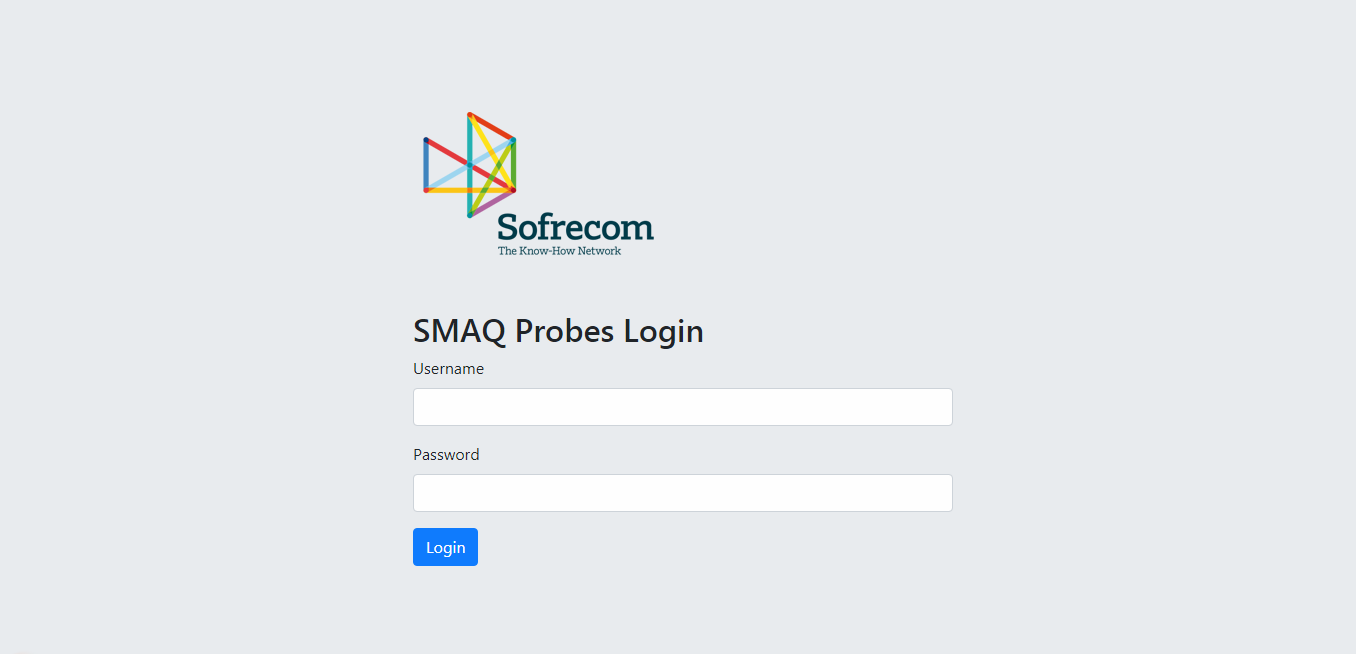
With the above screen figure 55, we can access to the user creation form.

This screen figure 66 is used for creating and updating users:



This form holds the user information, the username, the password and the privilege are essential for user creation.

If a user is registered successfully, an authentication is required before starting using the application features, the figure below figure 44 presents the authentication screen:

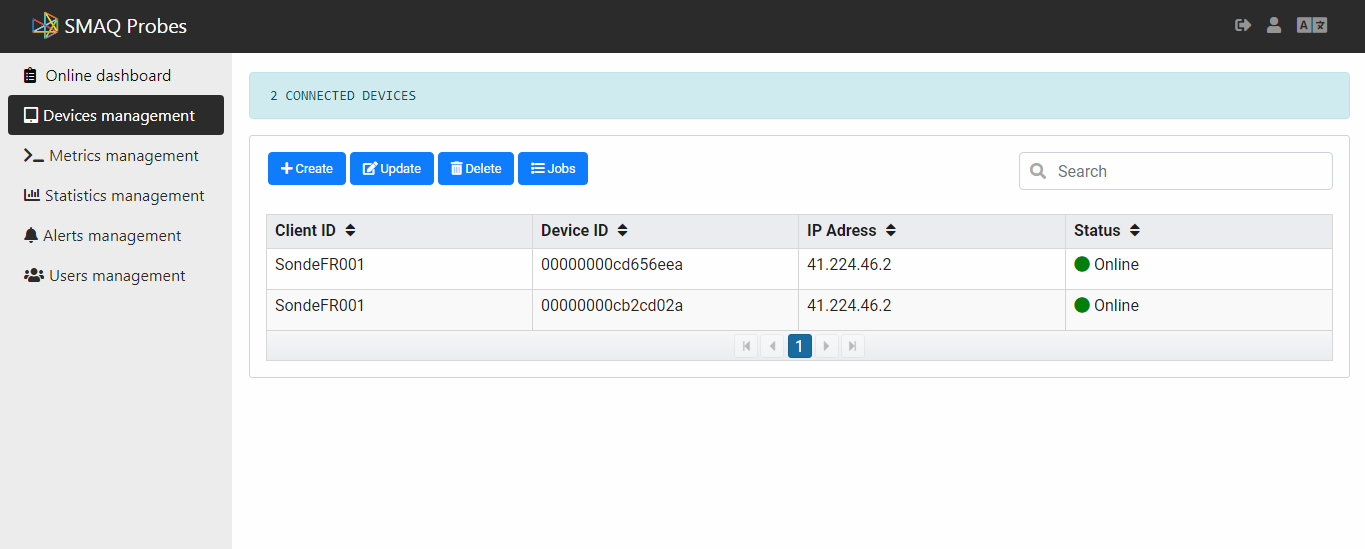


If the user enters the right credentials, the application redirects him to the dashboard screen, else an error message is shown in the login screen. If the access is granted to the user, he can use the application features depending on his role.

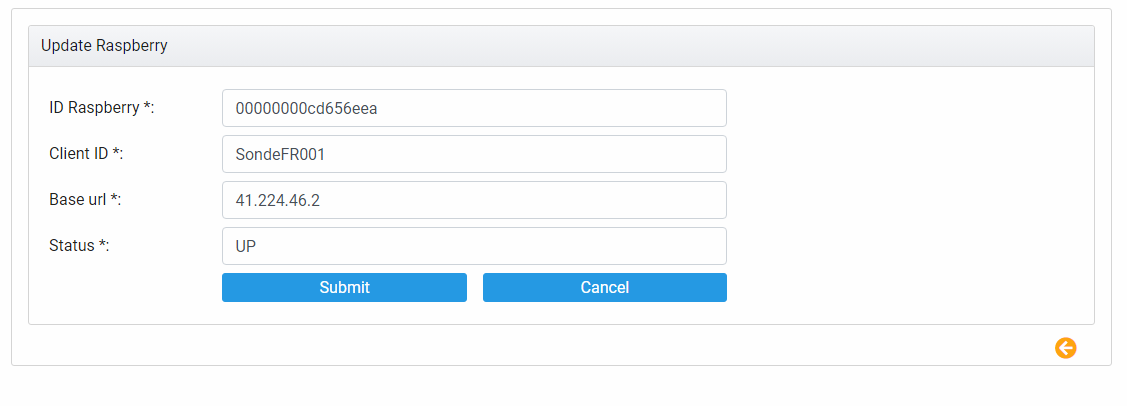
**4.3.1 Devices management:**

After authentication, to start using all the application features, the probes, in our case Raspberry Pi boards, must be connected in addition to the installation of the embedded application.

When the probe is connected, it sends its information to the application, the information contains the IP address, the device identifier, the device position, the client name and the running tasks on it. If the application receives the device information, the tasks and the configuration that concerns this device is sent to it for implementation. The device information is persisted to MongoDB. Now, the devices configuration can be displayed to the application users, shown in figure figure 55 below:

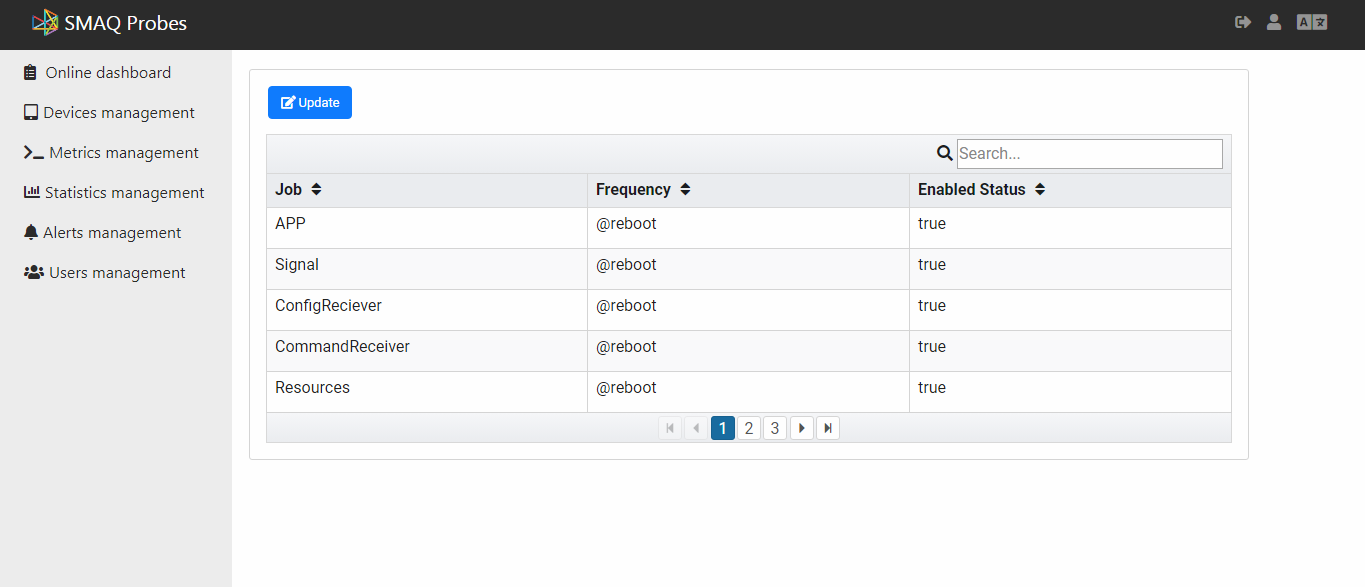


With the previous screen, we can navigate to the updating screen of a specified device, the figure below figure 69 presents the device updating form, this form contains the general information of the probe.

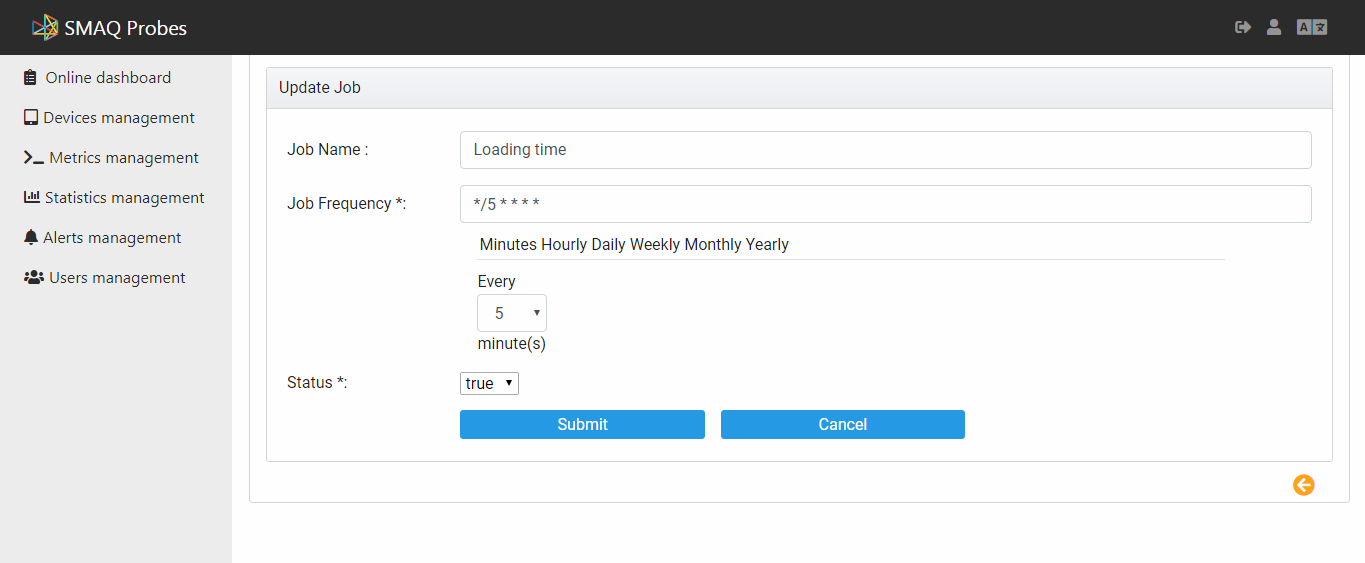


After updating the device information, the new configuration is persisted to MongoDB.

With the device list screen, we navigate to the running jobs of a specified probe, as shown below in figure 96:



With the previous figure figure 96, we can update the configuration of the device jobs. A job configuration is essentially the job status and the job frequency. For the job frequency, we are using the task scheduler Crontab. The figure below figure 96 presents the job updating screen:

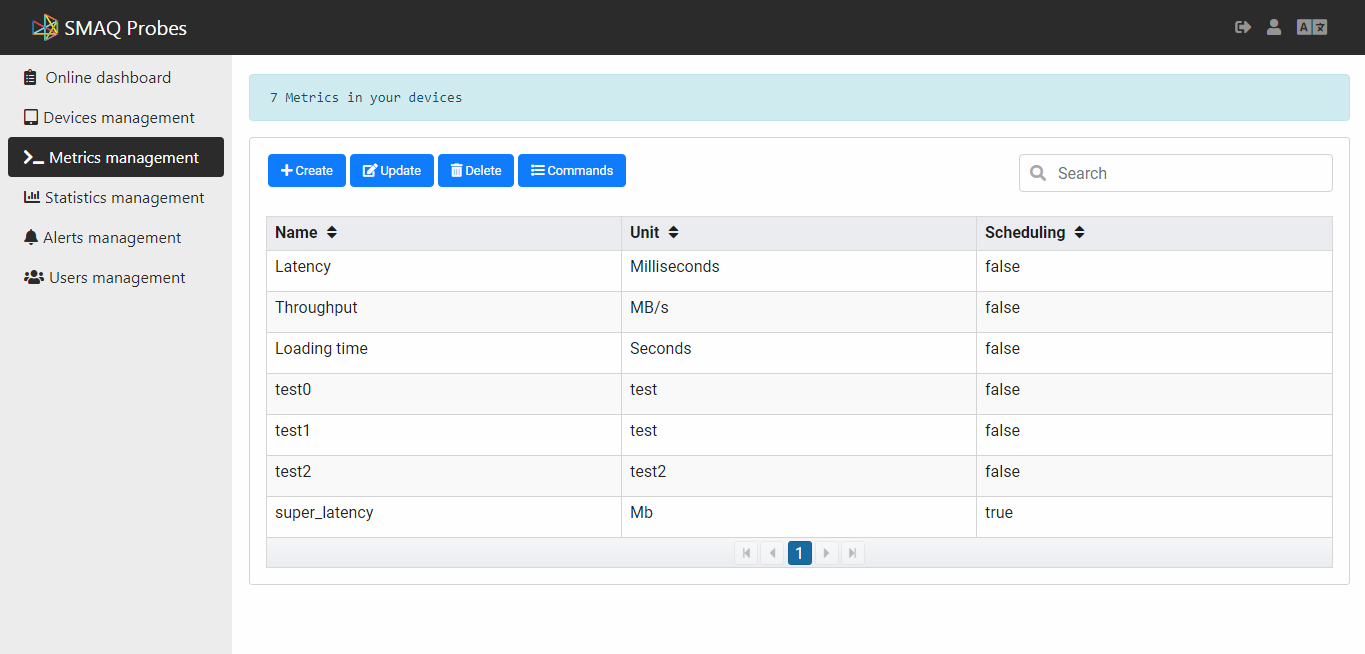


After submitting this form, the new jobs configuration is sent to the device in question in order to implements it.

The device jobs include the running tests, we next section is about how to manipulate these tests.

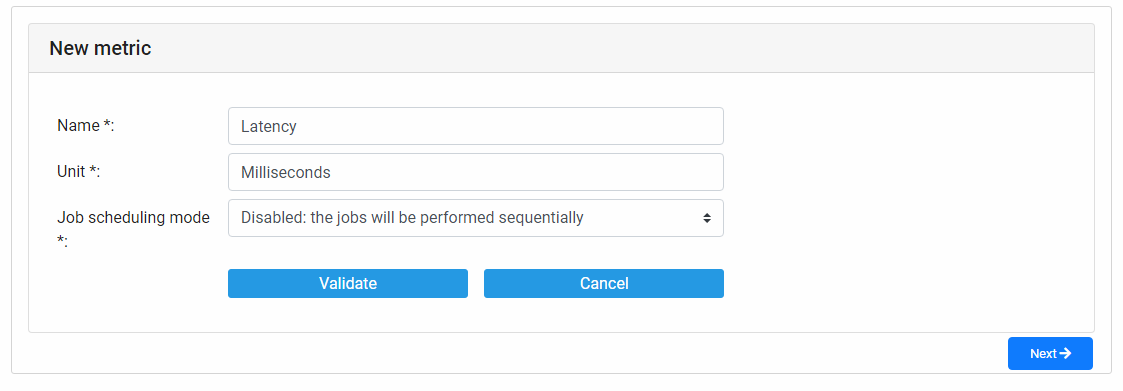
**4.3.1 Metrics and tests management:**

The metric and test management are the feature that gives the possibility to the network and broadband administrator to manipulate the tests running on the probes. The figure below figure 96 presents the list of tests already created and running on the devices:

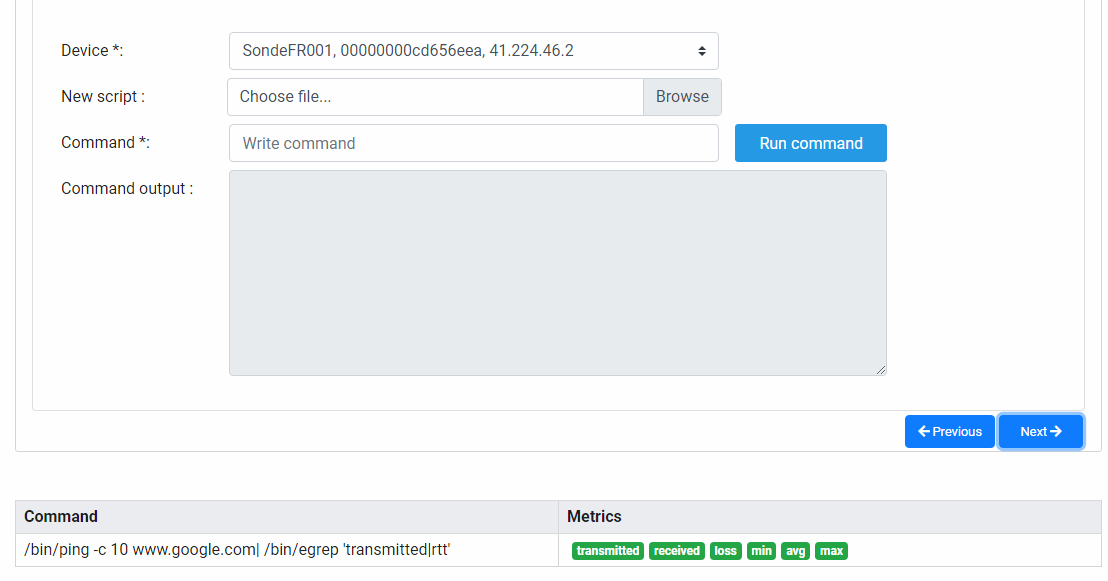


Using the previous figure, we can access to the creation screen of the tests. The process of test creation goes through several steps, these steps are illustrated in a wizard as described next.

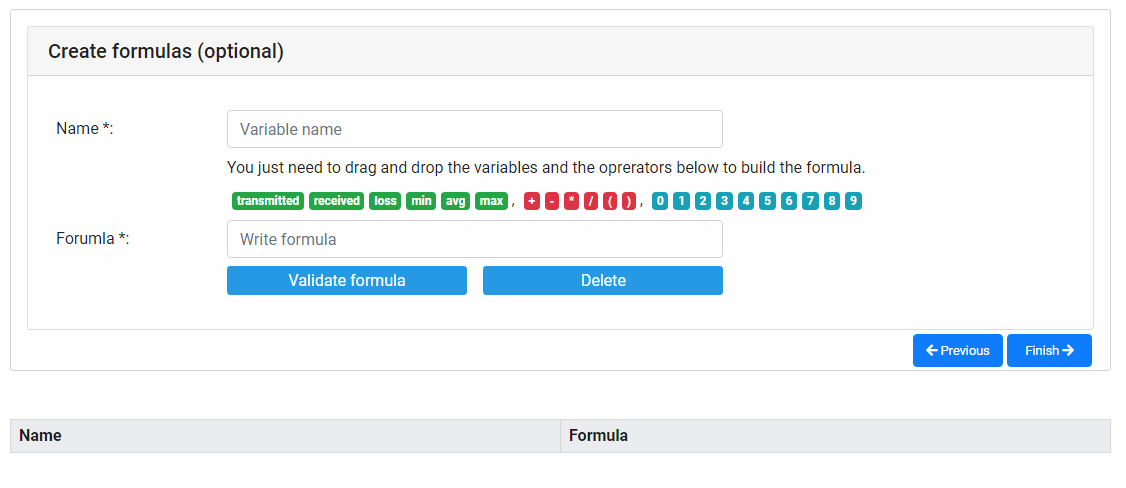
First, we have the form that contains the general information of the test, its name, the unit that is used for the metrics of the test, and the job scheduling mode. If the job scheduling mode is enabled, each element of the test runs in a specified order. The figure below figure 96 presents the general test information form:



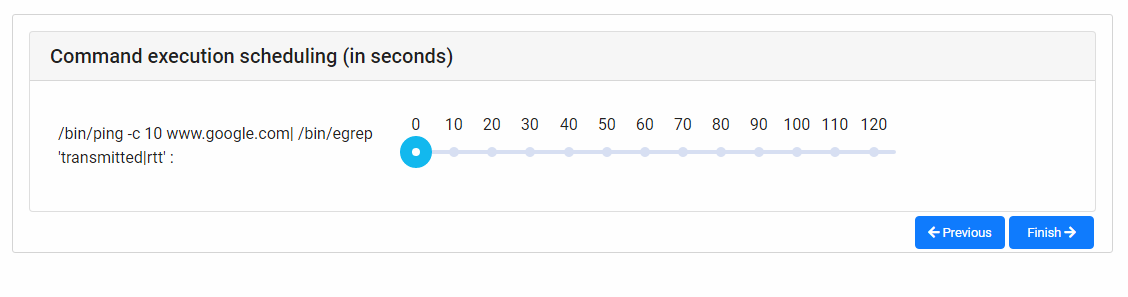
Now the user should enter the elements of the test. An element can be a Linux command line that runs an available script or an existing program. To complete this step, we must have a least one connected device, this device is used to test the elements on it. After entering the command line and the script, if we need it, we run the command on the available device. The test result is shown in the command output field. The network and broadband supervisor highlight the needed metric value. Here the system creates a parsing key that is used to extract the desired results in the process. We repeat this process for all the test elements. The figure below figure 963 shows the form that contains the element information.



To improve the use of our application, the network and broadband administrator has the possibility to configure some formulas that use the already configured test metrics. These formulas help us get more relevant results about the QoS. The figure figure 852 presents the form that is used to enter formula configuration. We have used a drag and drop component to give the network and broadband administrator a better experience with the application.



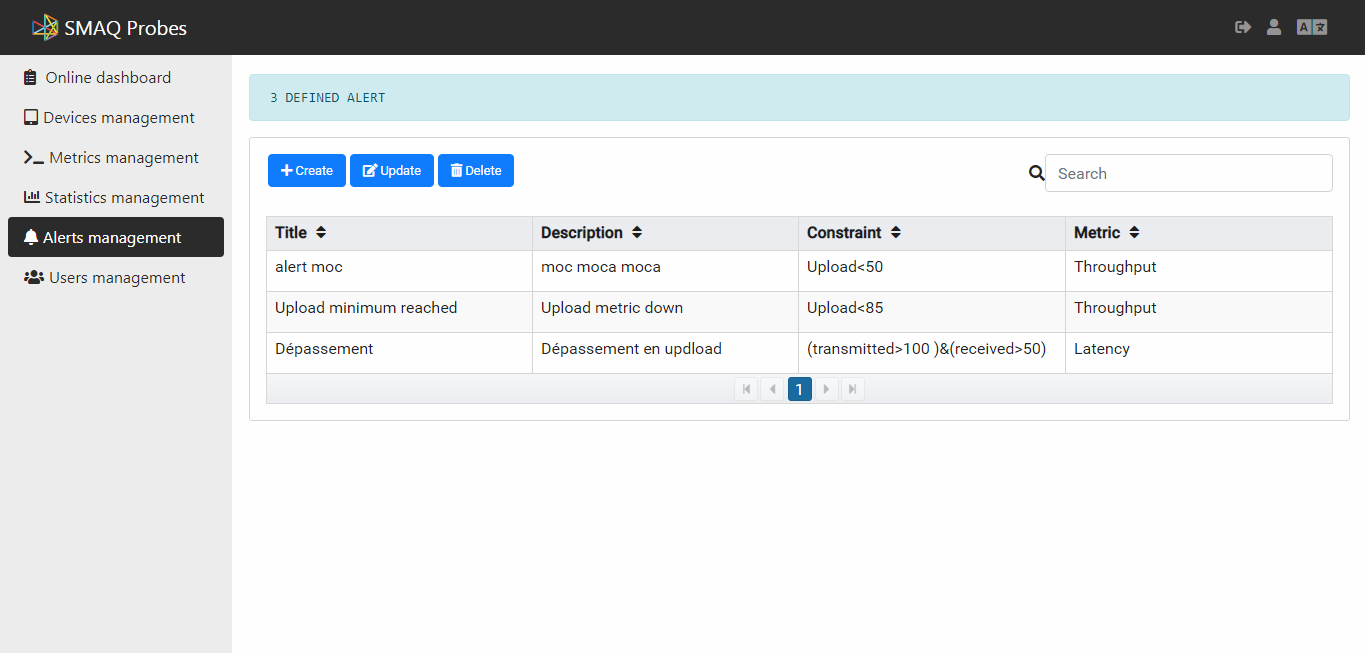
The last step is the test elements scheduling, this step is only activated if the job scheduling mode is enabled. In this step, we have all the test element information displayed in the screen. The network and broadband administrator enters the schedule of each element in seconds. The test elements run on the probe according to the specified order. The figure below figure 741 presents the form in question:



Finally, the network and broadband administrator confirms the test configuration. This new configuration is persisted to MongoDB and it is sent to the involved devices in order to be considered. This test is added to the jobs of devices.

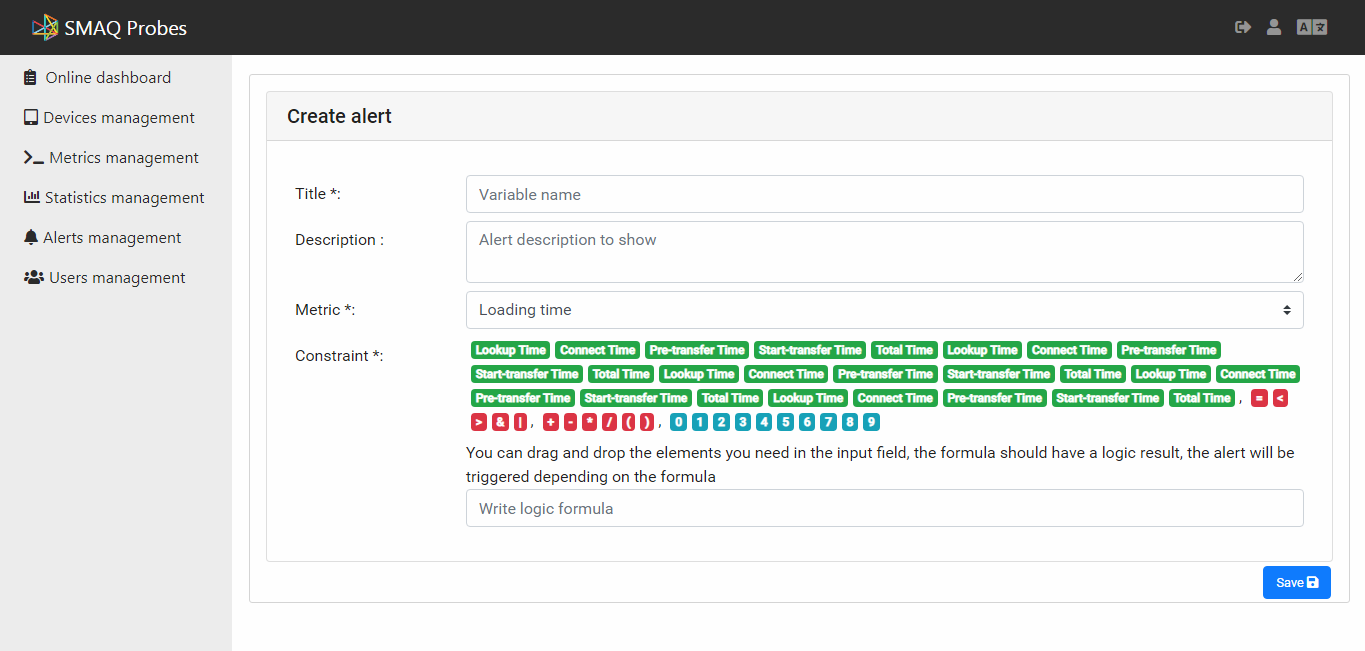
**4.3.1 Alerts management:**

Until now we have the connected devices list, and we have the running tests configurations. In this part, we talk about the network monitoring and the alert management. First, we have the figure figure 852 that shows the list of the configured alerts.



As we can see, with the above screen we have the access to create and update the alerts configurations.

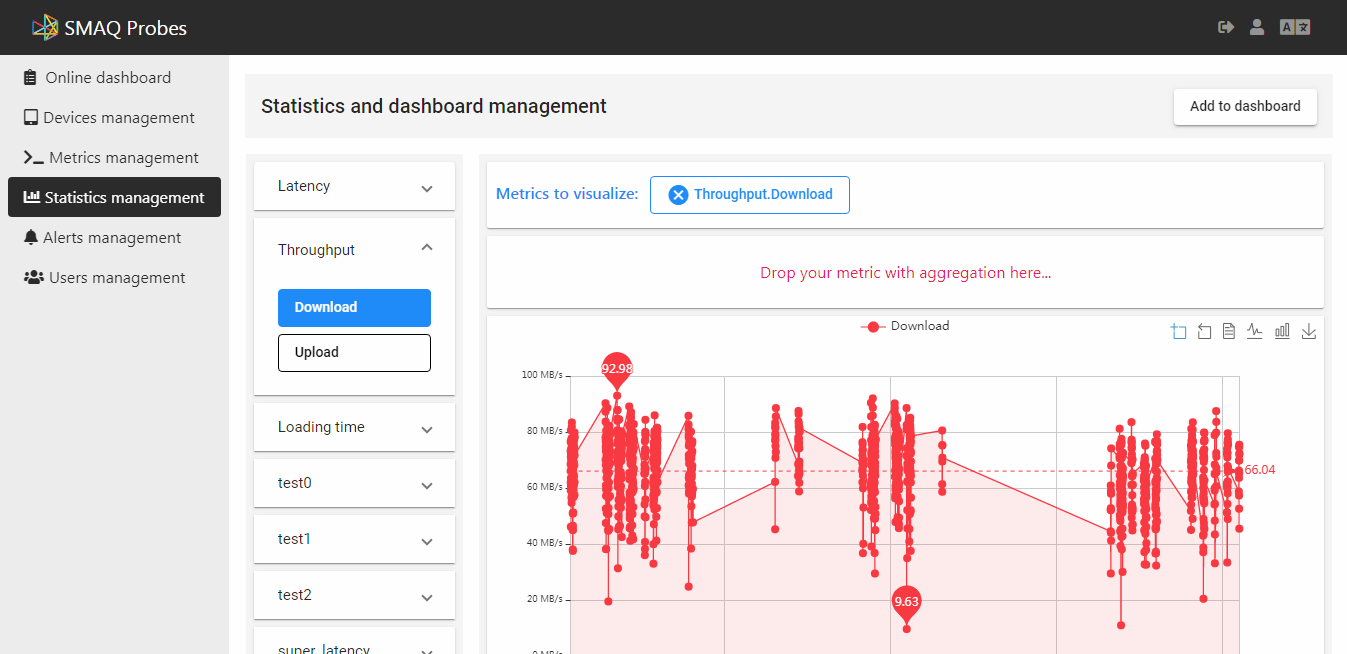
The figure below figure 852 shows the alert form configuration:



To create an alert, we have the general information of the alert, the title, the description that is shown when the alert is triggered. An alert is created according to a specified test and metric configuration, so the network and broadband administrator should choose a specified test for the alert. The test metrics names are shown. Finally, the network and broadband administrator must specify the constraint of alert triggering. This constraint is a logic formula composed using the configured metrics of the specified test. The network and broadband administrator submits the configuration details, this configuration is persisted to MongoDB. The alert configuration is fetched in the real-time processing with Kafka Stream, the constraint is tested according to the involved metrics, if the constraint is satisfied, the alert is triggered with the custom description.

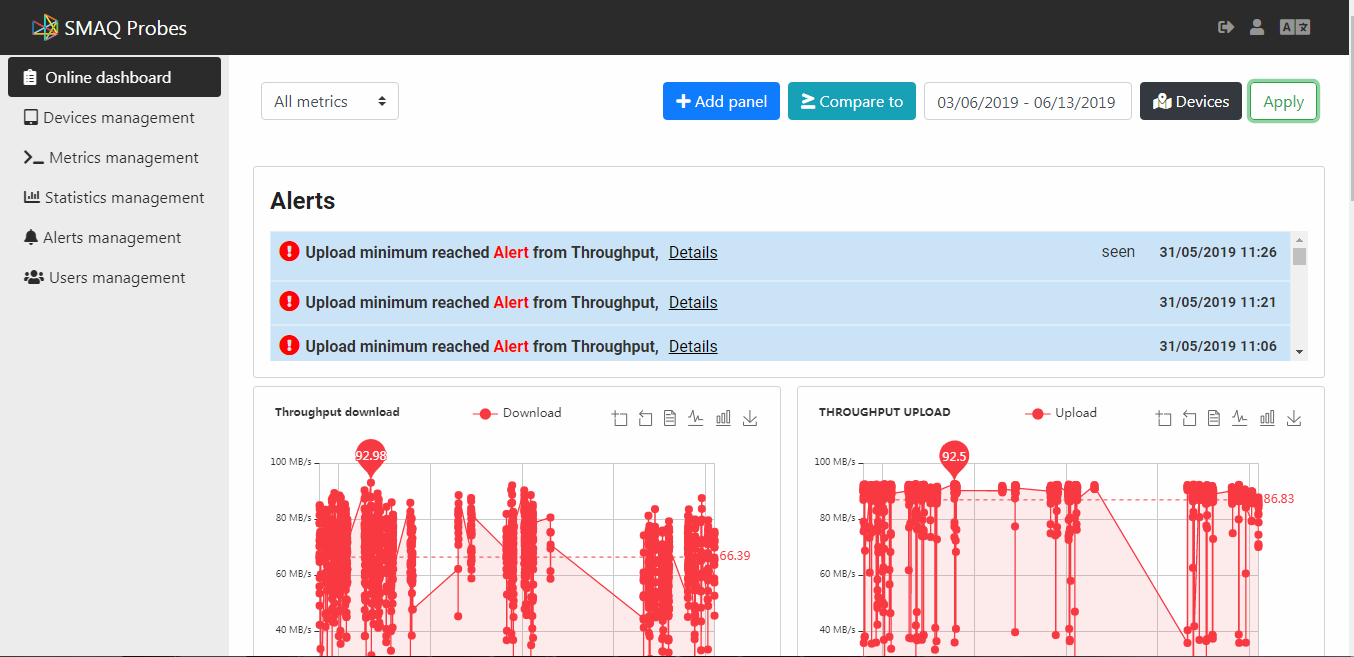
**4.3.1 Statistics and dashboard management:**

The following screen figure 789 presents the views and charts configuration:



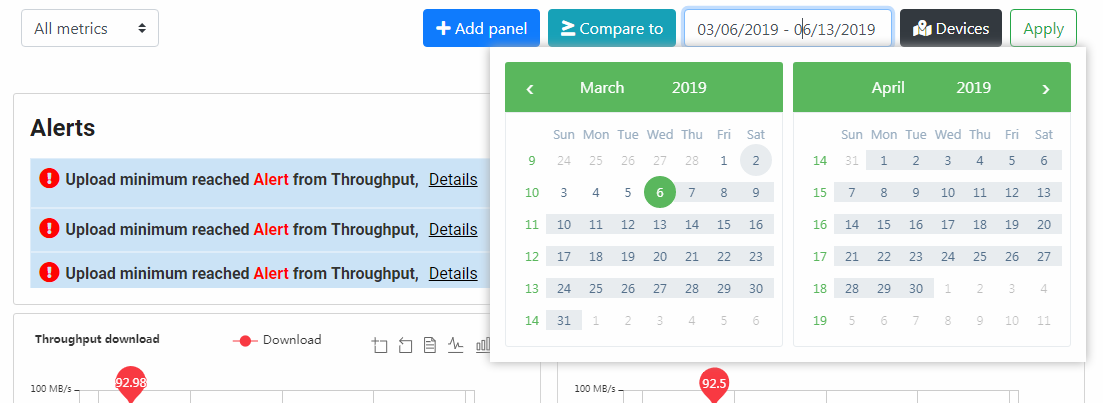
A chart is configured according to a test configuration. In the left side we have the list of the tests’ configuration with its metrics. Also, we have two fields, the first one is titled “Metrics to visualize” and the second one is titled “Metrics with aggregation to visualize. If the network and broadband administrator drops the metric in the first field, the chart shows the metric values by time. If the network and broadband administrator drop the metric in the second field, the chart shows the values according to an aggregation operator, the sum, the mean, the maximum, or the minimum, with a specified period. If the network and broadband administrator desires to add the chart to the dashboard, he clicks on the “Add to dashboard” button. The configuration of the chart is persisted to MongoDB. So, with this configuration we can generate the views on the dashboard.

The figure figure 345 presents an overview of the dashboard:



As we can see in the dashboard, we have the views that we saved its configurations. Also, we have the triggered alerts that we saved its configurations. To generate the different views, we have to use the available filters, the time filter, the zone filter and the test filter.

The following screen figure 456 presents the time filter:



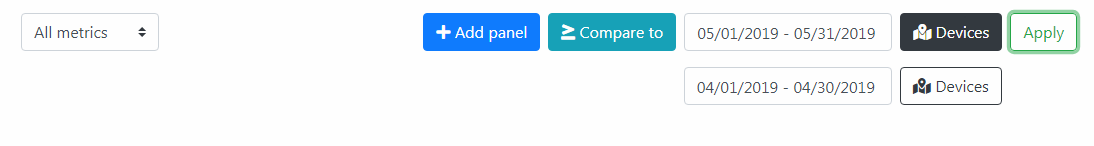
This filter specifies the period of time that we use to generate the statistics according to it. We use for this a time range picker. With this filter we can get relevant statistics about the QoS.

The following figure figure 234 presents the zone filter:



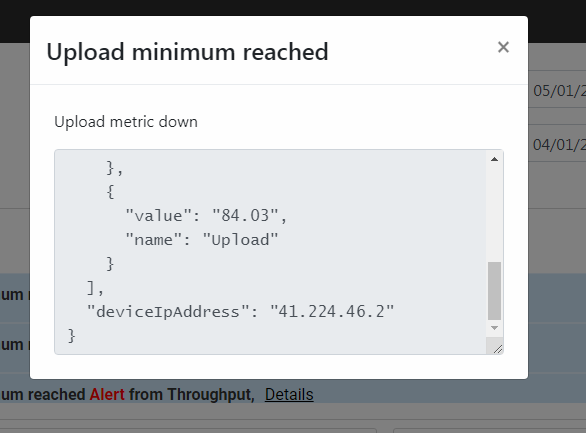
As mentioned before, we have the information about the devices’ positions. Using the position information, we can locate the probes on the map. The network supervisor can select an area on the map. SMAQ Probes generates the statistics using the devices that are located in the specified area. This filter allows the network supervisor to evaluate the QoS in a specified area.

The following figure figure 753 shows the comparison filters.



By clicking the “Compare to” button, an additional filter is shown, using these filters, the system generates charts that includes a comparison between two different results. This feature gives the possibility to the network supervisor to compare between different QoS information, this helps him improve the network service.

The following screen figure 741 shows a sample of a triggered alert information:



In the real-time processing, when an alert is triggered, the alert holds within it the detailed information about the metric value that satisfied the constraint. This feature helps the network supervisor to get quick information about the broadband network problems.

**4.4 Conclusion:**

During this chapter, we presented the technologies that were chosen to implement the project features and we argued that choice for the different components of the project. We also presented the used frameworks and tools, and we finished by displaying the main features and interfaces from SMAQ Probes application.

Sofrecom has been involved in an RFP from a regulator where part of the request was to massively assess and monitor the QoS of fixed networks access (ftth).

None satisfying “on the shelf “tool solution has been found either inside or outside the Orange Group.

Taking into account this situation, Sofrecom has brainstormed about a tool which will be internally developed, and which will fulfill the following needs:

remotly

**Conclusion and future work**

We were focused on solving the broadband supervision and monitoring issue by designing and developing a flexible and easy-to-use platform. As a matter of fact, we implemented a web application that helps network supervisors to assess the quality of service of their networks. This information (quality of service) is provided by data analytics, this data is coming from widespread probes. We designed the platform in order to give users the flexibility they need to manage their tests and devices online.  
For the graphic user interface, we developed many advanced graphic options to give users better experience with our application.

During the project, we were aware of data security, so we implemented many security techniques.

As every project can be enhanced and expanded, various perspectives are imminent for our project. We can mention the enhancement of the real-time processing component using machine learning algorithms to predict the quality of experience, there is some technologies that can be integrated with our platform in order to achieve these enhancements. With this empowering we can predict unusual quality of experience.