2020/2021

CSC5002 Micro Project: VlibTour

Tutored by:

Chantal Taconet, Denis Conan,
Sophie Chabridon

Realized by:

Amir Gharbi

Marouane Kachouri

Table of contents:

1. Introduction :

VLibTour, is a new mobile application with which groups of tourists can visit different points of interest in Paris by bike, while having the freedom of choosing their path to the different POIs of their group. VLibTour also gives the tourists the possibility to communicate their position to the members of their group at all time. Consequently, every tourist knows where the different members of the group are every time he needs to know. In this report, we present our POC of the VLibTour system and we defend our design, data model and technology choices along the realization of this micro project,

At first, we begin by the design patterns of some key components in our system. We proceed by illustrating the general architecture of our model. Subsequently, we explore the different modules in the system and explain their functionalities along with the advantages of the used technologies. Consequently, we analyse our system with regard to two quality attributes notably interoperability and security. Finally we present the difficulties encountered during the project and we propose possible addings that were not implemented,

2. Architectural design :

2.1 Design patterns :

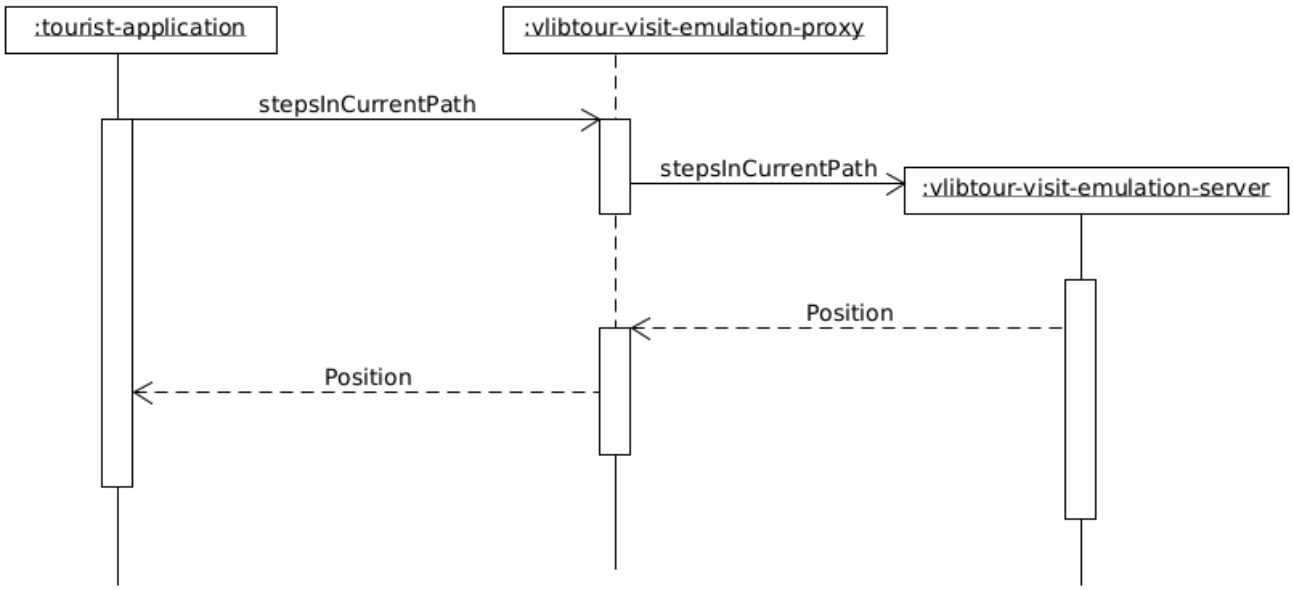

Figure 1: Sequence diagram of steps in current path

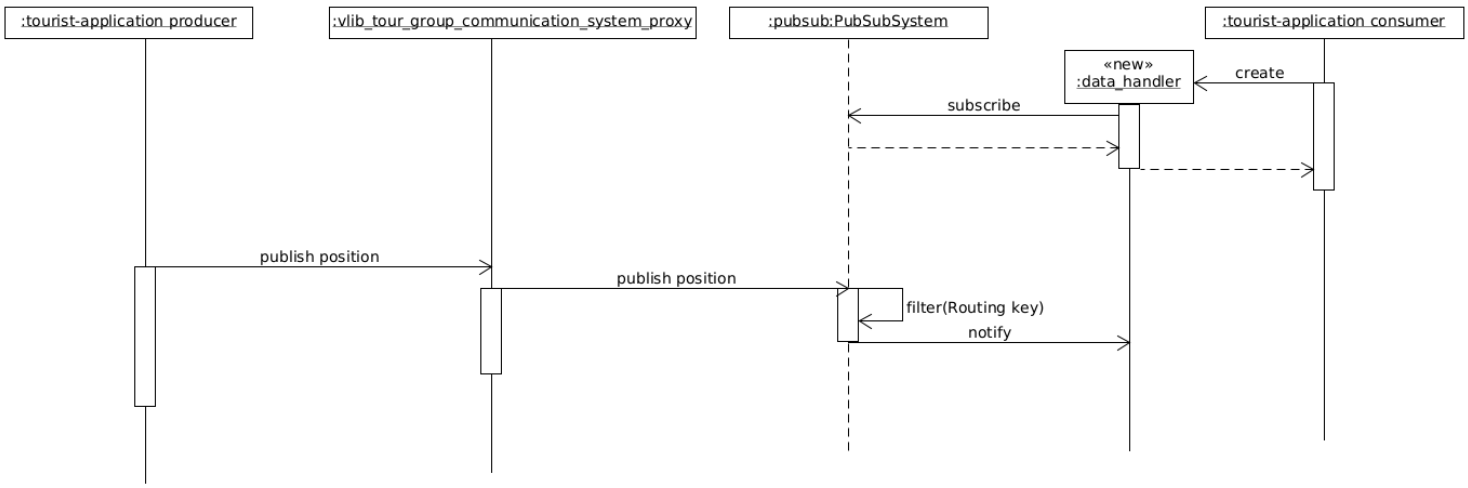

Figure 2: Sequence diagram of group communication system

2.2 Architecture of the POC :

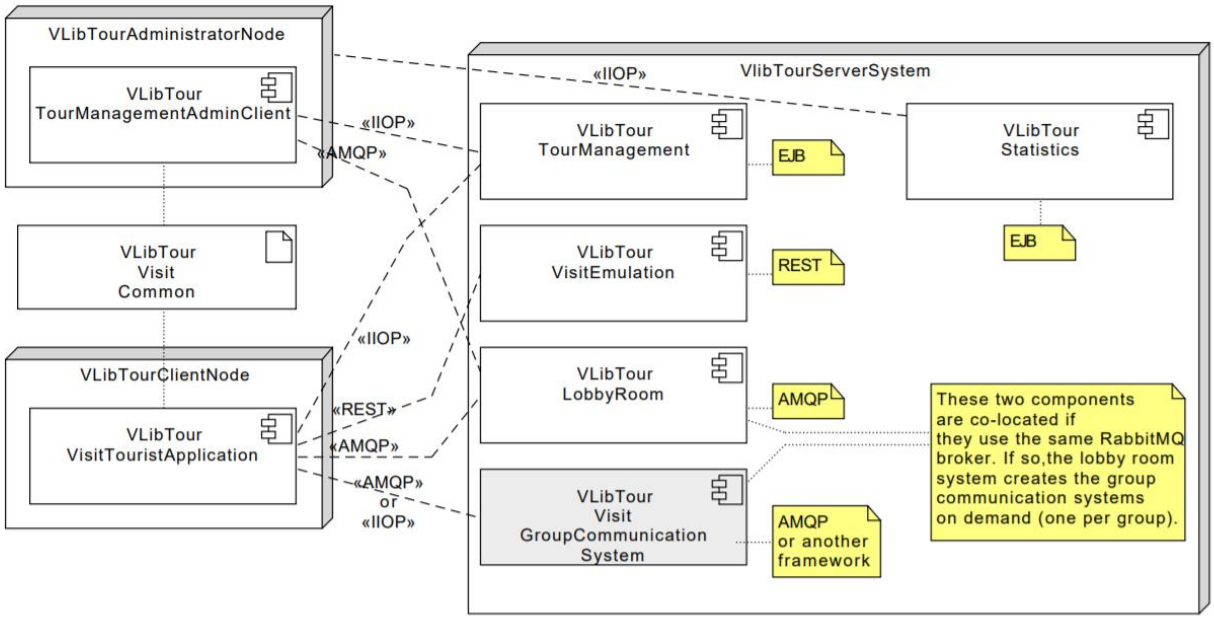

Figure 3: Architecture implemented in the POC

The modules in the VlibTourServerSystem represent different functionalities and use different technologies. The reason behind the choice of technologies will be highlighted further in the report.

3. VlibTour Modules :

3.1 VlibTour TourManagement :

This module is responsible for the creation, update and deletion of tours and POIs. We also have the possibility to assign different POIs to different tours (ManyToMany).

Technology wise, we opted for an Enterprise Java Beans (EJB) oriented architecture. We create a glassfish server instance containing the EJB container and then we deploy our Bean server on the container. We used EJB because of the various services it offers like security and concurrency control.

In order to create and persist data, we used a relational database because it is more fitted to our application and easy to manipulate.

3.2 VlibTour Statistics :

The statistics module allows the users to view representative numbers concerning the different entities in a given period of time. The module contains various statistics concerning the groups, tours and POIs. Examples of such information are the average group size, the most popular tour and the total number of groups.

The following module wasn't implemented in our project but we know it should use the EJB technology for the same reasons listed in the above module

3.3 VlibTour VisitEmulation :

This module allows the user to locate himself on the map at a any given moment. The user can check the next location to reach on his current path and he can check his next POI.

We used the REST technology in the implementation of this model. The client's requests are transmitted to the server that implements the api methods desired through a proxy.

The scalability that REST offers along with its stateless property facilitate the task of handling the increasing number of clients through the introduction of new servers since they function without client context.

3.4 VlibTour VisitGroupCommunicationSystem :

This module allows the user to locate himself on the map at a any given moment similar to the VisitEmulation module. However, this module adds the possibility for a tourist to check the location of every other member in his group simultaneously on the same map. This is important since tourists of the same group, although they have different paths, have to wait for each other to depart from every POI.

We used an AMQP architecture that allows every tourist to be either a producer that posts his location to the exchange or a consumer that receives the location of other tourists from a queue.

Our choice of AMQP is justified because it uses a publish/subscribe architecture for data sharing and it conserves messages in queues waiting to be consumed. We can add to the advantages of AMQP the fact it sends data as a stream of bytes thus allowing interoperability between different senders and receivers.

3.5 VlibTour LobbyRoom :

This module is meant to complement the VisitGroupCommunicationSystem and add the security aspect to it. In this module a client application has the option to contact the lobby room to either create or join a group. In the case of choosing to create a group, the lobby room is sollicited to create a dedicated virtual host to the group. For joining a group, the client has to provide the identifiers of the group and its creator.

Specifically, our lobby room server is an RPC server. The RPC server receives the tourist's request to create or join a group through a RPC client that represents a proxy. The server then returns a url to the RPC client. The url is generated from the received parameters (groupId, userId, tourId, generated password). Finally, the url is transmitted to the tourist through the RPC client.

This module uses the same technology of AMQP.

4. Extrafunctional requirements :

4.1 Interoperability :

Since our system relies on a AMQP architecture we can be sure that a certain degree of interoperability is maintained. According to www.rabbitmq.com, a lot of interoperability testing took place a some of the tests were successful. Successful tests of interoperability include but are not limited to: RabbitMQ clients and Qpid Java broker, Python clients and RabbitMQ broker.

4.2 Security :

Although our lobby room implementation provides a relatively secure system wuth randomly generated passwords and different hosts for each group, nothing prevents a user from requesting and getting access to a group he's not part of. This is because in order to join a group, a client only needs the creator's identifier and the group's identifier that is a concatenation of the tour identifier and the creator's Identifier. It is clear that this information is not hard to obtain even when the user doesn't belong to the group. Additionnally, another flaw in the system consists of the fact that a user can intercept the returned url and use it to identify himself as the victim user thus gaining access to the group. Both possible scenarios are illustrated below.

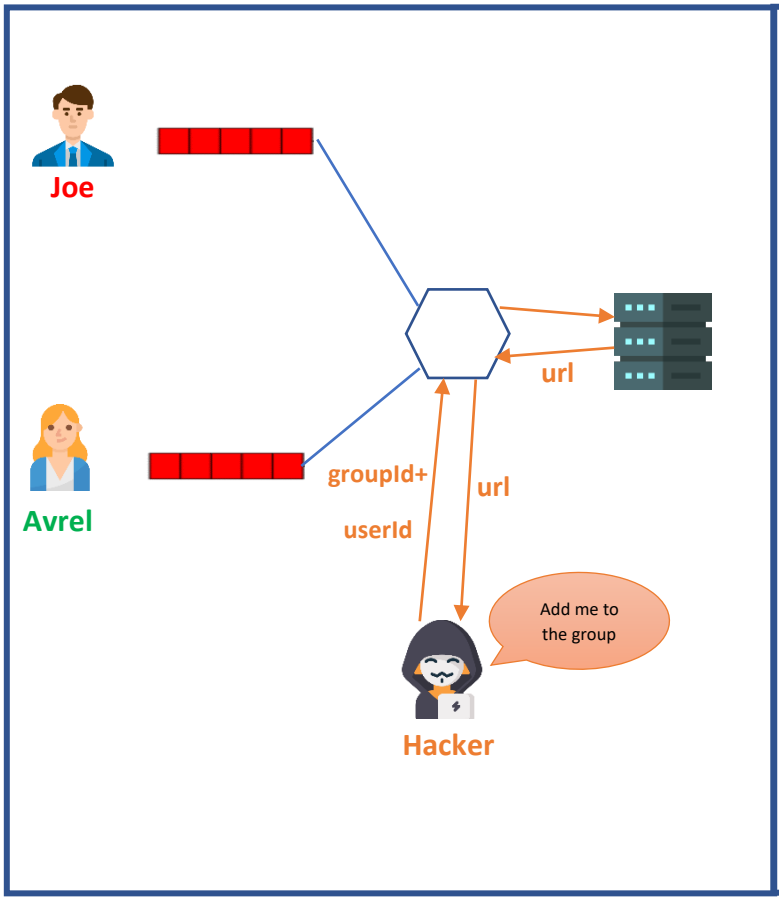

Figure 4: Security flaw scenario 1

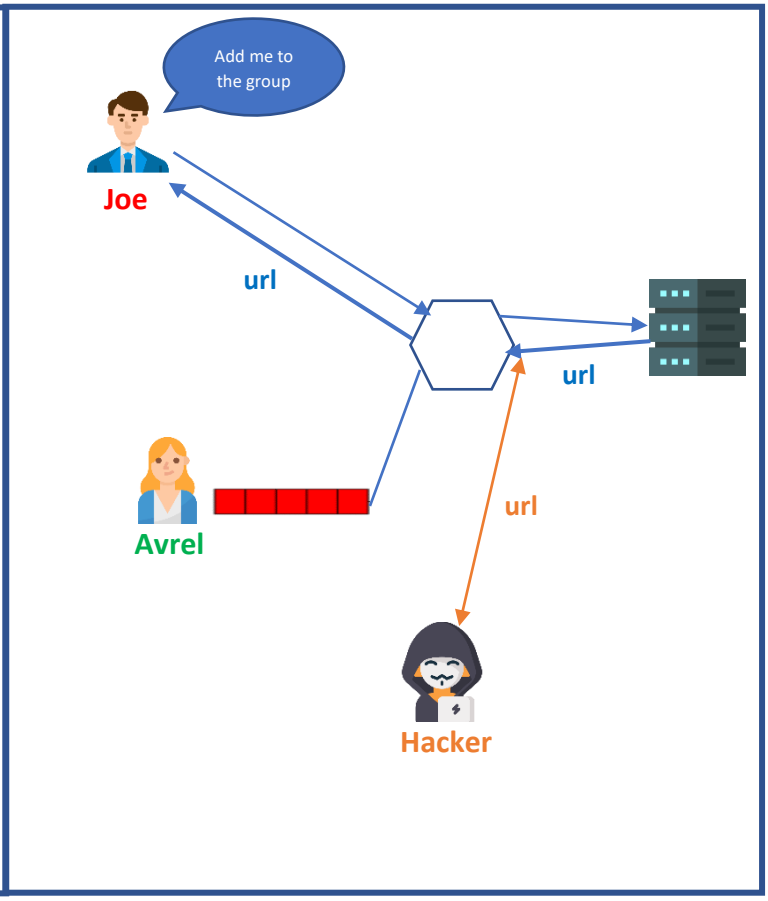

Figure 5: Security flaw scenario 2

5. Difficulties encountered :

Although we expect a lot of people faced difficulties with working from a distance and dividing work with their partner, this wasn't a problem for us since we lived in the same building and could meet at any time to discuss our ideas and implement them together.

However, we thought at times that the information provided in the labs was too vague and that generated some confusion. Thankfully this wasn't a major setback since we could ask the tutors and our colleagues for help when uncertain about something.

Another point we want to make is that we really appreciated continuous feedback on our git projects in last semester's learning unit : CSC4509. Pointing out bugs and issues really helped us with our projects. This point isn't much of a difficulty and we understand it's a micro project but we wanted to make point of how much we appreciated the continuous feedback.

6. Going forward :

Going forward, we can think of so many functionalities we can add to better our system. Most noticeably, we think the statistics module would be a good add so that users have a more global view of the different tours and their ratings. Additionnally, as of now, once the visit is over the client session closes and thus he is not able to reconnect to his group since he doesn't have the password and login. This is why implementing the external part where the client receives his login and password along with the possibility to connect to a group when there isn't a visit going on seems like a good addition to our POC.

Finally, it would be profitable to further explore the security aspect of the POC hence prevent the scenarios we mentioned in section 5.2.