RÉPUBLIQUE DU CAMEROUN
Paix – Travail – Patrie
- . - . - . -
UNIVERSITÉ DE YAOUNDÉ I
Faculté des Sciences
Département d'Informatique
B.P. 812 Yaoundé

REPUBLIC OF CAMEROON
Peace – Work – Fatherland
- . - . - . -
UNIVERSITY OF YAOUNDÉ I
Faculty of Science
Department of Computer Science
P.O.Box 812 Yaoundé

INF 4178 - SOFTWARE ENGINEERING

PROJECT REPORT :Group 5

Project Members

Students' ID	Students' names	Participations rate
18T2610	FENYEP WANGUE	20%
19M2441	FOTSO WAFO Omer Élysé	20%
19M2557	KANA ZANLEFACK Blondelle	20%
19M2069	KOMBOU MBIANDA Armel Dilane	20%
18U2355	NGIMDOCK ZEMFACK	20%

Instructors:

Dr Kimbi Xaveria
Mr Tekoh Palma

Table de Matières

1. Topic

Mobile app to search for an appropriate hospital in order to receive treatment for malaria.

2. Research Problem

Due to the highest rate of malaria cases in Africa (95%) and the highest rate of malaria death (96%) observed in 2021, the access to treatments about malaria is very difficult and so closed in the African region. So we have identified the following research problem : "**Lack of possibilities and incapability to receive good treatments for malaria.**"

3. Objectives

a. General objectives

This project focuses on development of a mobile application (app) to help people to find and localize an appropriate hospital to receive effective health care for malaria . We also aimed to implement a Web application for the administrator who will add information about hospitals.

b. Specific objectives

Using this app, our goal is to allow people to get numerous benefits like:

- give the possibility to all users to view all the hospitals in their location area
- Search the appropriate hospital to treat malaria for low cost
- Add criterias for searching hospitals in order to obtain a hospital according to them

4. System requirements

For the system to be developed we will specify the functional and non-functional requirements.

a. Functional requirements

The functional requirements of the application are grouped by actor.

As a user, I have to :

- See all the hospitals available in my locality on a map in order to have a large choice of hospitals that specialize in the treatment of malaria.
- Get the path from my location to a hospital in order to follow it.
- Add criterias for searching hospitals in order to filter the result.

As an administrator, I have to:

- Login to the system in order to access the dashboard and manage hospitals.
- Register hospitals where there is effective treatment for malaria into the system in order to recommend it to people.
- Update hospital information in order to share up-to-date hospital information.
- Delete hospital in order to remove it in the user search result list.

b. Non-functional requirements

The non-functional requirements of the application are chosen by criteria.

- **Easy to use** : All the screens of the mobile application must be accessible in a few clicks for quick use by a patient.
- **Efficiency** : The update of the path from my current position to the hospital must be in real time .
- **Reliable** : Hospital information displayed to users must be correct.
- **Portability**: The application produced could be used on android or ios and must be able to install in less than 3 minutes;

5. Application of Scrum

a. Presentation of scrum team

Scrum is an agile way to manage a software development project . In this project, we are 5 members each having a very specific role in the team

Roles	Team members Names	Responsibilities
Product Owner	FOTSO WAFO Omer	A team member who represents the voice of the customer and prioritizes user stories
Scrum Master	KOMBOU MBIANDA Armel Dilane	A team member who: – Acts as buffer between the Team and external distractions – Keeps team focused on task at hand – Enforces team rules – Removes impediments that prevent team from making progress
Developer team	- FENYEP WANGUE - FOTSO WAFO Omer - KANA ZANLEFACK Blondelle - KOMBOU MBIANDA Armel Dilane - NGIMDOCK ZEMFACK	The development team's responsibilities include: - Ensure shippable items are delivered on time during project iterations - Resolving project issues in a realistic way

b. Description of how you applied scrum to your specific project

i. Explanation of how Sprints were carried out

To effectively run an organization, complete every task on time and work efficiently as a team, it is necessary to understand what sprint means in project management. In Scrum, a sprint is a set of objectives, benchmarks, or outputs that a production team focuses on for a set period of time. Our Scrum teams create products through sprints or iterations, which divide apart large, complex projects into smaller, more manageable chunks. To enhance the effective execution of our project planning , we chose Sprint planning in **Jira** (a software tool) so that the end result is high-quality and features.

ii. Team organization and roles

A Scrum Team is a small cross-functional and self-managing group of people able to create value during each Sprint. It is the Scrum Team that decides what is being done, by who, and when. For this project, we decided to form a team composed of : **Product Owner**, **Scrum Master** and **Developers**.

- ❖ **Product Owner** is a team member (not the ScrumMaster) who represents the voice of the customer and prioritizes user stories
- ❖ **Scrum Master** offers process and method coaching to the product owner, developers, and stakeholders. Additionally, the scrum master acts as a change agent and facilitates organizational development.
- ❖ **Developers** creates products that offer a great user experience and have the right quality

iii. Daily scrum Agenda

iv. Scrum conflict Resolution

We didn’t have conflicts while working on this project.

v. Scrum workflow management

Concerning the workflow management of the project we decided to use many tools:

- ❖ **Version Control System (Git | GitHub):** To easily control the version of our application and work in a group, it was a great idea to add this tool into our tools box. That makes the development of our application easier.

- ❖ **Task management (Jira):** To organize our tasks and easily follow what we want to develop without any problem. This helps us to know our progression and also helps us to work well.

- ❖ **Communication**: For communication, we decided to create a **Whatsapp group** where we exchanged every time when we wanted to talk about the project.

vi. Product Backlog

Already Done

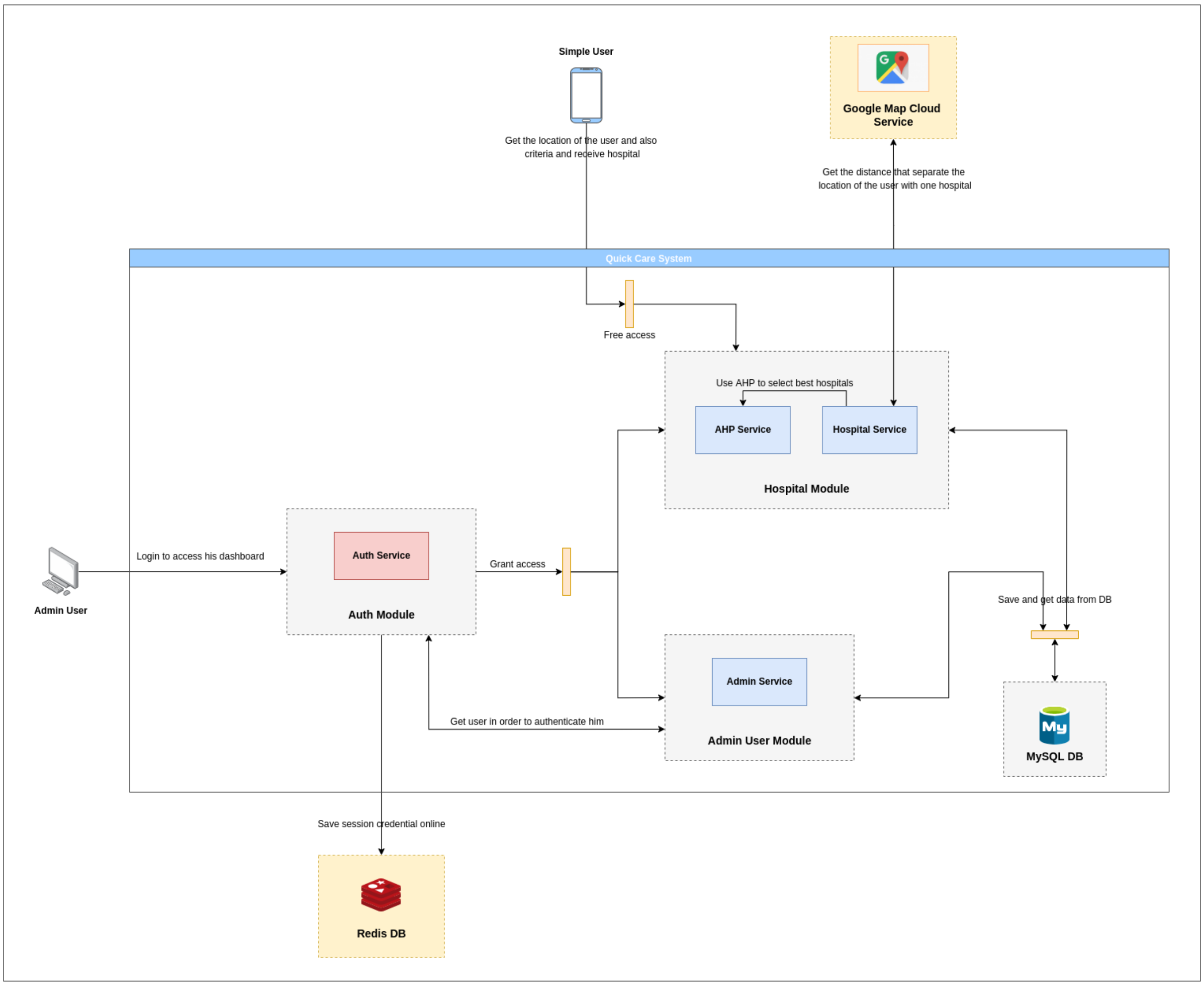
vii. Sprint Backlog

Already Done

6. Methodology

a. Architecture of your system

i. Logical architecture


Simple User
Google Map Cloud Service
Get the location of the user and also criteria and receive hospital
Get the distance that separate the location of the user with one hospital
Quick Care System
Free access
Use AHP to select best hospitals
AHP Service
Hospital Service
Hospital Module
Login to access his dashboard
Auth Service
Auth Module
Grant access
Admin User
Save and get data from DB
Admin Service
Admin User Module
Get user in order to authenticate him
MySQL DB
Save session credential online
Redis DB

ii. Description of Architecture

The entire system is divided into modules (three in total) and one internal database such as:

- **Authentication module**: secure the system
- **Hospital Module**: manage hospital and implement the **AHP** logic
- **Admin user module**: manage administrator user data
- **MySQL Database**: store data of the system about hospital, users

Each module has at least one service which is responsible to carry out the business logic of the module.

And we have some external services like:

- **Google Map cloud service**: used to calculate the distance that separates a simple user and a hospital.
- **Redis Database**: used to save session credential of the connected user

How are things working ?

About the **Administrator** of the system, he has to first login into the system in order to start working and accessing data and then manage them. That's what the **Authentication module** job is, and this module has to connect to the **Admin User module** in order to retrieve the user data and verify it. After the identity of the administrator is verified the system will grant his access to the rest of the system and his credentials will be stored into a **Redis DB** online.

About the **Simple User** of the system, his main need is to get access to the best hospital to go and receive treatments about malaria. So, firstly we get his location from the mobile phone and ask him to provide some criteria in order to filter the result.
He doesn't need to have an account so that he has free access to the **Hospital module** as presented on the architecture. When his request is received by the Hospital module, the **Hospital service** will first get the hospital in his country and then call the **AHP service** to get the best that fits his need and at the end the list of hospitals will be sent back to the user.

iii. Architectural Drivers

- **Scalability** : APIs should be designed to be scalable, so that they can handle increasing demand without impacting performance.

- **Security:** APIs should be designed to be secure, so that they are protected from unauthorized access and attacks. This can be achieved by using a variety of security measures, such as authentication, authorization, and encryption.

- **Performance:** APIs should be designed to be performant, so that they can respond to requests quickly and efficiently. This can be achieved by using a variety of techniques, such as caching, compression, and load balancing.

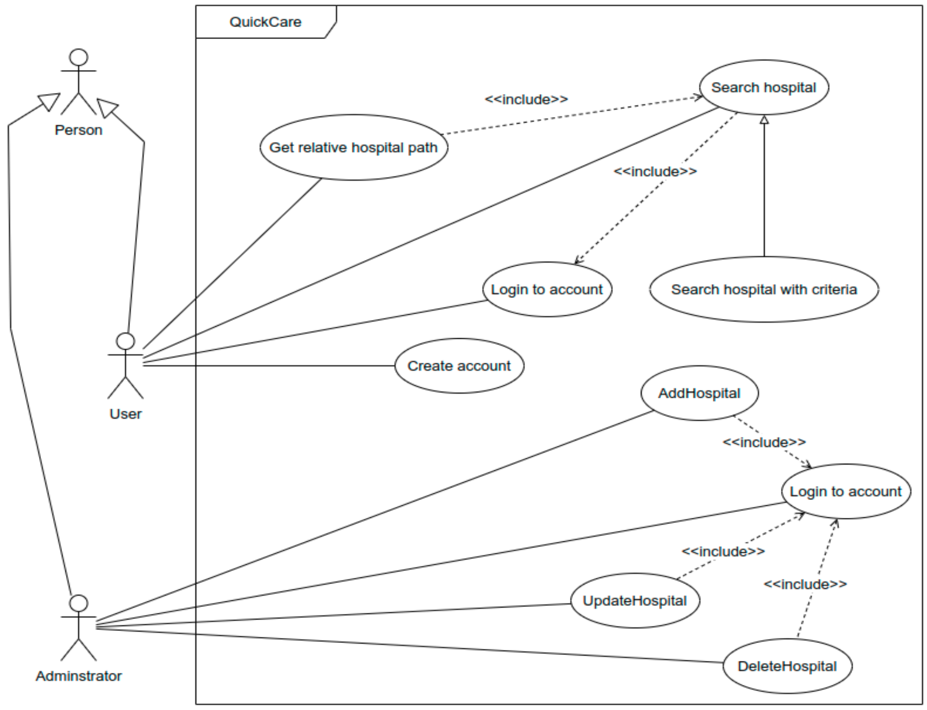
- **Reliability:** APIs should be designed to be reliable, so that they are available when needed.

- **Maintainability:** APIs should be designed to be maintainable, so that they can be easily updated and changed. This can be achieved by using a variety of techniques, such as using well-defined APIs, using versioning, and using documentation.

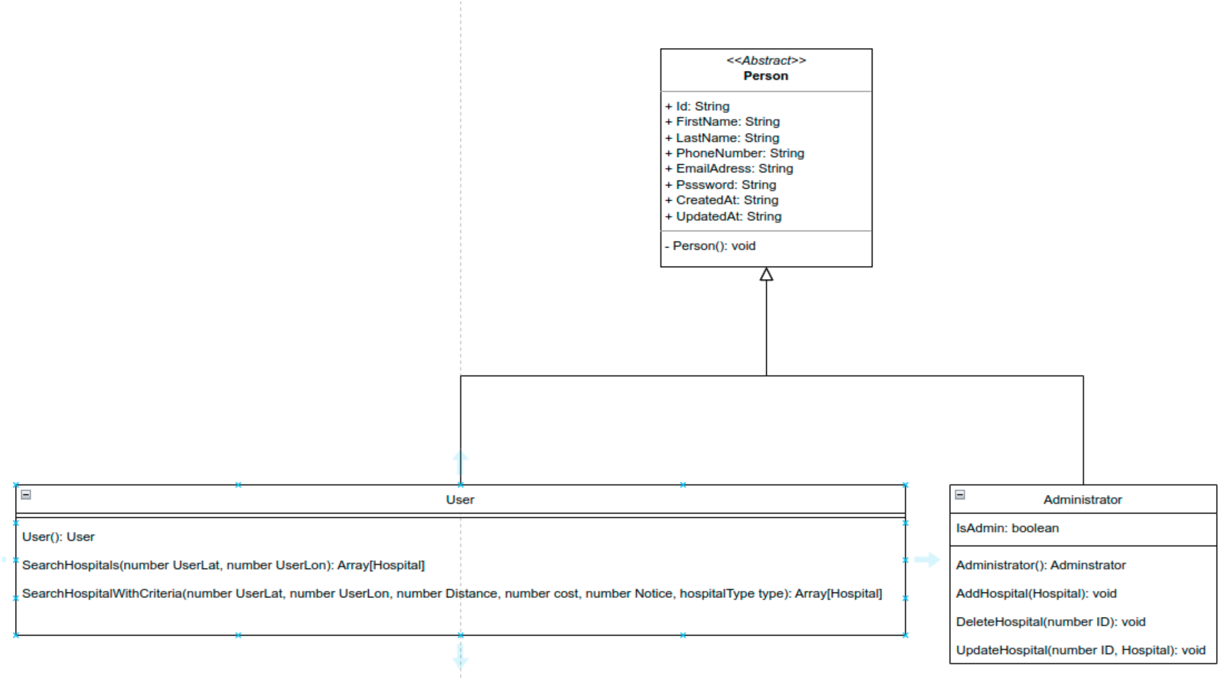
b. Model of your system

i. Model UML

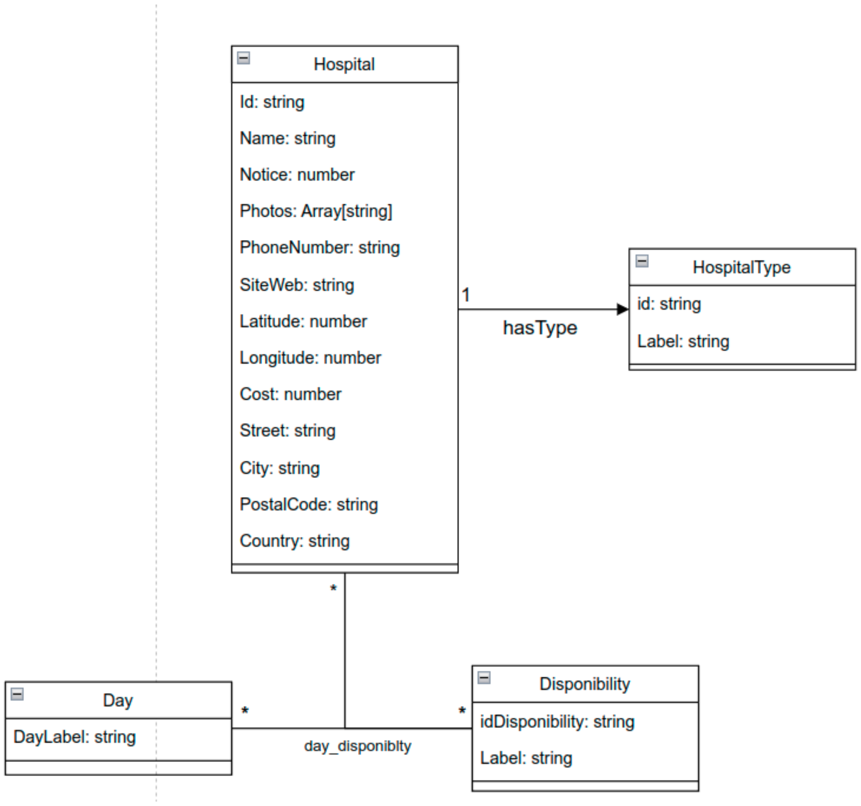
1. Use case diagram



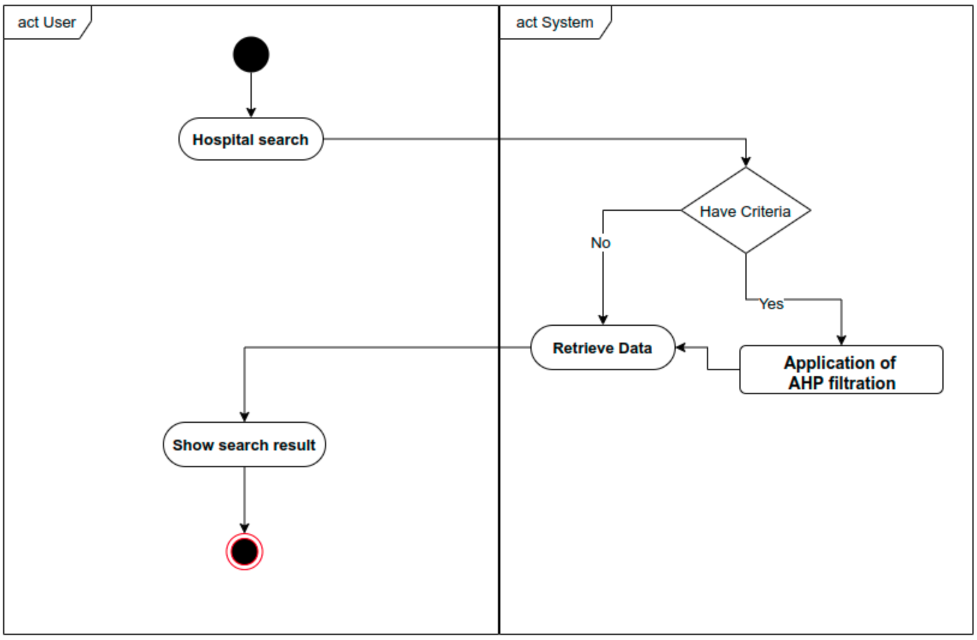
2. Class diagram



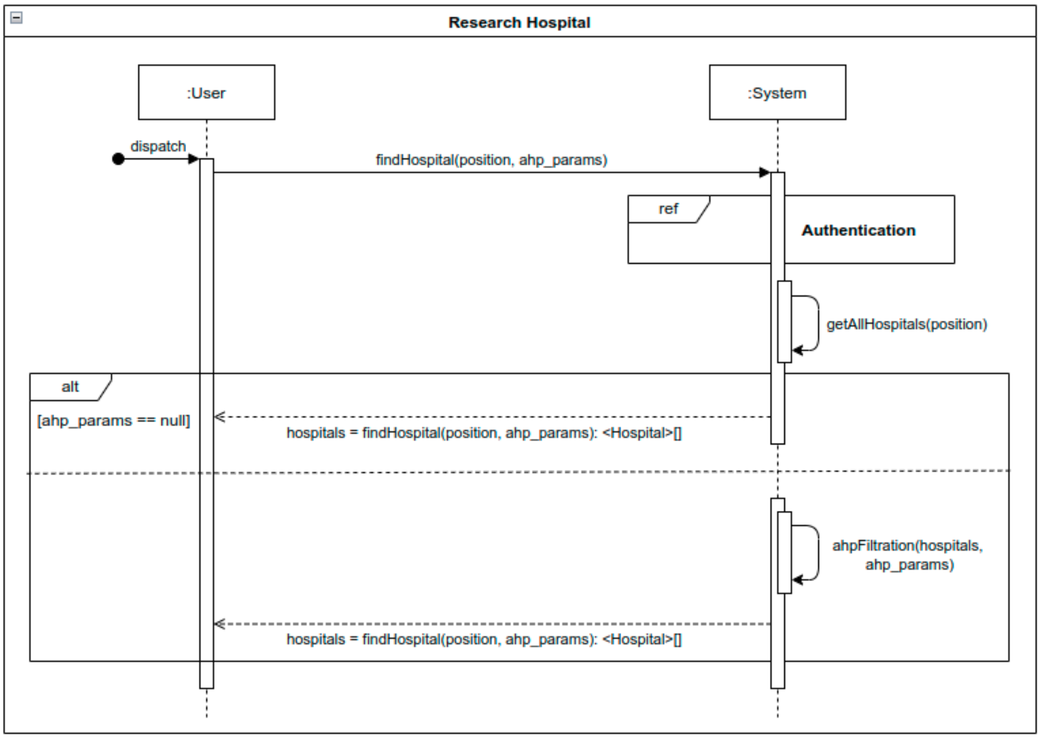
<<Abstract>>
Person
+ Id: String
+ FirstName: String
+ LastName: String
+ PhoneNumber: String
+ EmailAdress: String
+ Pssword: String
+ CreatedAt: String
+ UpdatedAt: String
- Person(): void
User
User(): User
SearchHospitals(number UserLat, number UserLon): Array[Hospital]
SearchHospitalWithCriteria(number UserLat, number UserLon, number Distance, number cost, number Notice, hospitalType type): Array[Hospital]
Administrator
IsAdmin: boolean
Administrator(): Adminstrator
AddHospital(Hospital): void
DeleteHospital(number ID): void
UpdateHospital(number ID, Hospital): void



3. Activity diagram



4. Sequence diagram


Research Hospital
:User
:System
dispatch
findHospital(position, ahp_params)
ref
Authentication
getAllHospitals(position)
alt
[ahp_params == null]
hospitals = findHospital(position, ahp_params): <Hospital>[]
ahpFiltration(hospitals, ahp_params)
hospitals = findHospital(position, ahp_params): <Hospital>[]

ii. Mathematical Model

Mathematical Model

The mathematical model of the software system for searching hospitals.

let G=(V,E) a weighted directed graph where V is the set of nodes and E is the set of edges.

The set of nodes V contains three types of nodes:

1. User nodes u ∈ V, representing the starting nodes where the search for the hospital begins.
2. Hospital nodes h ∈ V, representing the destination nodes where users want to go.
3. Intermediate nodes c ∈ V, representing intermediate nodes (Crossroad, Street, hotel, restaurant…) that are important for navigation and pathfinding.

The graph G can also include additional information such as:

1. Node attributes: Each node in V can have additional attributes such as its geographical location, type (user, hospital, Intermediate).
2. Edge attributes: Each edge in E can have additional attributes such as its length.

iii) Algorithm

Once the graph is constructed, Dijkstra's algorithm can be applied to find the shortest path from a user to a hospital as follow:

- Initialize the distance to all nodes as infinity except for the starting node u , which is set to 0.
- Set a priority queue to hold the nodes to be visited, with the starting node as the first entry.
- While the priority queue is not empty, do the following:
  a. Remove the node with the smallest distance from the priority queue.
  b. For each of its neighboring nodes (c or s), calculate the distance to that node as the sum of the distance to the current node and the weight of the connecting edge.
  c. If the calculated distance is smaller than the current distance to that node, update the distance and add the node to the priority queue.
- When the priority queue is empty, the distances to all nodes have been calculated. The shortest path from the starting node (user u) to the hospital can then be determined by backtracking through the nodes with the smallest distance until the hospital is reached.

c. Analytical Hierarchical process (AHP) algorithm applied to your project.

Step 1: Define goal, Criteria and some alternatives

Goal: The user want the best hospital to receive treatment about malaria

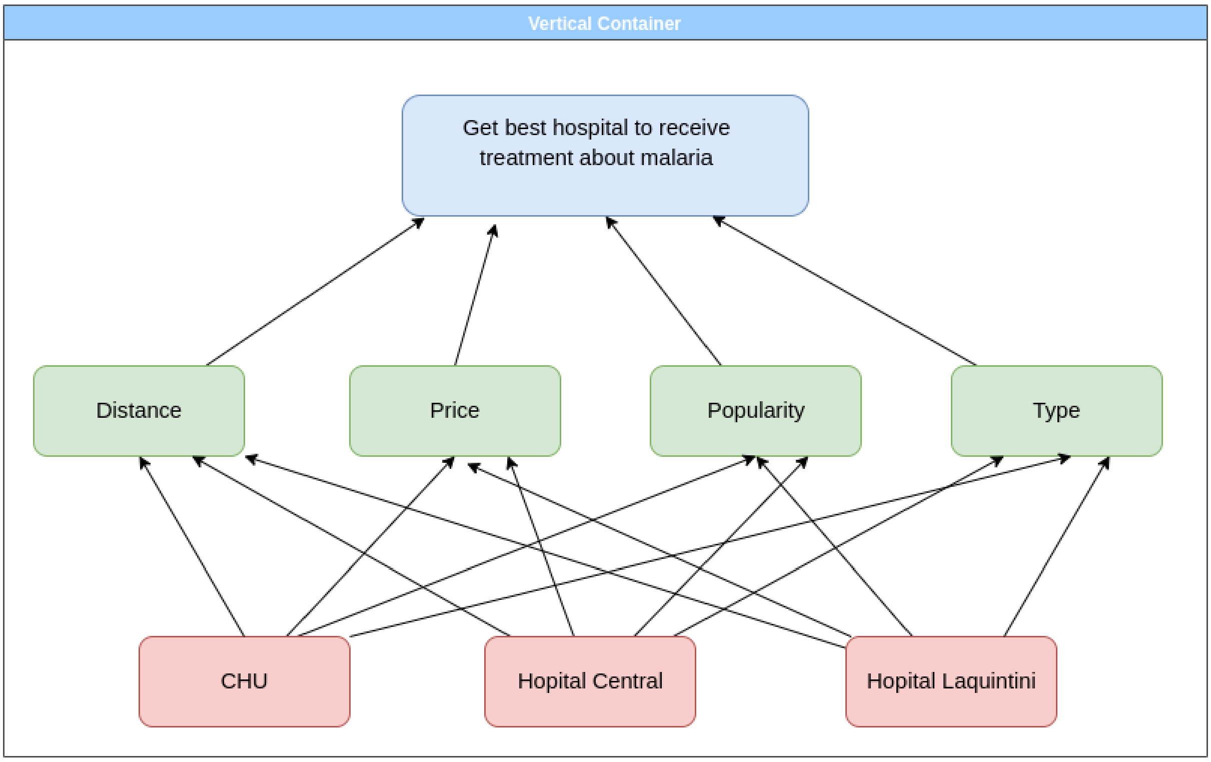
Criterias:

- Distance that separate the user with hospitals
- Price (minimal coast to receive treatment about malaria)
- Popularity (obtained by calculating the efficiency of each hospital)
- Type of hospital (public, private, clinic, etc.)

Alternatives:

- Hôpital central
- CHU
- Hôpital des soeurs de vogt-béti
- L'hôpital Laquintini
- etc.

Step 2: Draw the architecture of AHP



Step 3: Relative scale of preference

- 1: Equally important
- 3: Moderately important
- 5: Very important
- 7: Strongly important

- 9: Extremely important

Step 4: Pairwise matrix

Let D be Distance
Let C be Cost (Price)
Let P be Popularity
Let T be Type of Hospital

	D	**C**	**P**	**T**
D	1	3	5	7
C	1/3	1	3	5
P	1/5	1/3	1	3
T	1/7	1/5	1/3	1
Sum	1.676	4.533	9.333	16

Step 5: Normalized pairwise matrix

	D	**C**	**P**	**T**
D	0.597	0.661	0.536	0.438
C	0.197	0.22	0.321	0.313
P	0.119	0.073	0.107	0.188
T	0.085	0.044	0.035	0.063
Sum	1	1	1	1

Step 6: Calculare criteria weight

	D	**C**	**P**	**T**	**Criteria weight**

D	0.597	0.661	0.536	0.438	0.558
C	0.197	0.22	0.321	0.313	0.263
P	0.119	0.073	0.107	0.188	0.122
T	0.085	0.044	0.035	0.063	0.057

Step 7: Check consistency

	D	**C**	**P**	**T**	**Criteria weight sum**	**Lambda i**
D	0.558	1.052	0.61	0.399	2.619	4.693
C	0.186	0.263	0.366	0.285	1.1	4.182
P	0.112	0.086	0.122	0.171	0.491	4.024
T	0.08	0.053	0.04	0.057	0.23	4.035

Lambda max = 4.233, R = 0.9, CI = 0.077

CR = CI / R
= 0.077 / 0.9
= 0.086

CR = 0.086

As Cr is less than 0.1 that means our data is **consistent**