RÉPUBLIQUE DU CAMEROUN
****
Université de Yaoundé 1
****
Département d'Informatique
****

REPUBLIC OF CAMEROON
****
University of Yaounde 1
****
Department of Computer Science
****

Project report

INFO 4178 : Software Engineering I (Génie Logiciel I) TP

Membres de l'équipe

- **MELIE YEMELONG URIEL** 19M2222
- **MFOUT BANGMO YVANNA** 18T2864
- **NGOUTSOP RAINSONG** 19M2299
- **FATIA KOLOSSOUM** 22W2335
- **BIKOY ELIE EMMANUEL** 19M2520

Instructor: Dr Kimbi Xaviera

Plan of the document

1. Topic
2. Research Problem
3. General objective
    a. Specific objectives
4. System requirements
    a. Functional requirements
    b. Non-functional requirements
5. Application of Scrum
    a. Presentation of scrum team
    b. Description of how you applied scrum to your specific project
        i. Explanation of how Sprints were carried out
        ii. Team organization and roles
        iii. Daily scrum Agenda
        iv. Scrum conflict Resolution
        v. Scrum workflow management
        vi. Product Backlog
        vii. Sprint Backlog
6. Methodology
    a. Architecture of your system
        i. Architectural Diagram
        ii. Description of Architecture
        iii. Architectural Drivers
    b. Model of your system
        i. Model UML
            1. Use case diagram
            2. Class diagram
        ii. Mathematical Model

1. Topic

After a careful reading of the 5 problems to choose from, we opted for the third. From this problem we have drawn the following Topic: "**A platform to match job providers and workers based on needed skills**"

2. Research Problem

There are many research problems that could be addressed in the context of a platform to match job providers and workers based on needed skills.

This is a complex and challenging problem, but it has the potential to revolutionize the way that jobs are found and filled in the African context.

So, this is the problem we carried out:
Use AHP and mathematical model to build a platform for job providers in the aim to reduce unemployment rate .

3. General objective

A platform to match job providers and workers based on needed skills can have a positive impact on both the economy and society. It can help to improve the efficiency of the labor market, and it can also help to reduce unemployment and poverty.

- **Matching job providers with workers who have the skills and experience needed.** This can help to reduce the time and cost of hiring, and it can also lead to a better match between the skills of the workers and the needs of the job providers.

- **Providing workers with access to a wider range of job opportunities.** This can help workers to find jobs that are a better fit for their skills and interests, and it can also help them to earn a higher wage.
- **Increasing the transparency of the job market.** This can help workers to make better decisions about their careers, and it can also help job providers to find the best candidates for their open positions.

a. Specifics objectives

A platform to match job providers and workers based on needed skills can help to achieve these objectives for both workers and job providers. It can be a valuable tool for both parties in the labor market.

- **For workers:**
  - Find a job that is a good fit for their skills and interests.
  - Earn a fair wage for their work.
  - Have access to a wider range of job opportunities.
  - Be able to work from home or on a flexible schedule.
- **Job Providers**
  - Find qualified workers who have the skills and experience they need.
  - Reduce the time and cost of hiring.
  - Improve the efficiency of their operations.
  - Attract and retain top talent.
  - Stay ahead of the competition.

4. System requirements

a. Functional requirements

- Allow visitors to authenticate themselves (login or registration)
- Allow a user to fill in their professional profile
- Allow a service provider to search for service requests and filter them by several criteria (salary, city, applicant, etc...).

- Allow a service requester to search for a suitable service provider and filter them by several criteria (experience, city, etc...).
- Match service seekers with job offers in a single click.

a. Non-functional requirements

- The application must be easy to use and intuitive, with a clear and attractive interface.
- The application must be available on different devices (computers, tablets, smartphones) and function correctly on all common web browsers.
- The application must be able to handle multiple users (Workers, Job providers) and manage access rights accordingly.
- The application must be secure and protected against cyber-attacks and data leaks.
- The application must be easily extensible and customizable to meet the specific needs of each dermatology department.

5. Application of Scrum

a. Presentation of scrum team

Names	Rôles
MELIE YEMELONG URIEL	Scrum Master
NGOUTSOP RAINSONG	Product Owner
MFOUT BANGMO YVANNA	Member
FATIA KOLOSSOUM	Member
BIKOY ELIE EMMANUEL	Member

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

1. Team organization and roles

• “2 Pizza” team size (1 to 5 people)

• “Scrum” inspired by frequent short meetings

•15 minutes every day at same place and time

2. Daily scrum Agenda

➢ Answers 3 questions at “daily scrums”:

1. What have you done since yesterday?
2. What are you planning to do today?
3. Are there any impediments or stumbling blocks?

➢ Help individuals by identify what they need

3. Scrum conflict Resolution

➢ 1st list all items on which both sides agree

- vs starting with list of disagreements
- Discover closer together than they realize?
  - ➢ Each side articulates the other’s arguments, even if don’t agree with some
- Avoids confusion about terms or assumptions, which may be real cause of conflict
  - ➢ Constructive confrontation (Intel)

- If you have a strong opinion that a person is proposing the wrong thing technically, you are obligated to bring it up, even to your bosses
    - ➢ Disagree and commit (Intel)
- Once decision made, everyone needs to embrace it and move ahead
- I disagree, but I am going to help even if I don't agree.

4. Scrum workflow management

- At the beginning of the project, the roles are assigned and the team is assembled.
- A meeting is held between the Product Owner, the user, the Scrum Master and the Development team to extract the project Product backlog.
- The Project Product Backlog is a list of User Stories. User Stories are self-contained entities which define a functionality required by the customer for the project or the service. They can be grouped into a component which define a higher-level functionality
- of the system that cannot be delivered in a single sprint, and which are normally divided into smaller User Stories.
- For every User Story, a series of Acceptance Criteria are agreed between the customer (user) and the Product Owner (occasionally they might also be written between developers and Product Owners to increase transparency).

- A Sprint kick-off meeting is held between the **Product Owner, the Scrum Master and the Development team**. During the Sprint kick-off meeting the team decides which User Stories will be implemented during the sprint. The selected User Stories compose **the Sprint Backlog.**

5. Product Backlog

Product Backlog

ID	Description of users stories	Acceptance Criteria	priority	Initial estimate	Adjustment Factor	Adjustment estimate
01	As a user of the system, I should be able to create an account in order to have access to the system functionalities	When a user of the system creates an account, they are provided with an interface with various options that enables them perform functionalities based on their	01	10	1.5	10

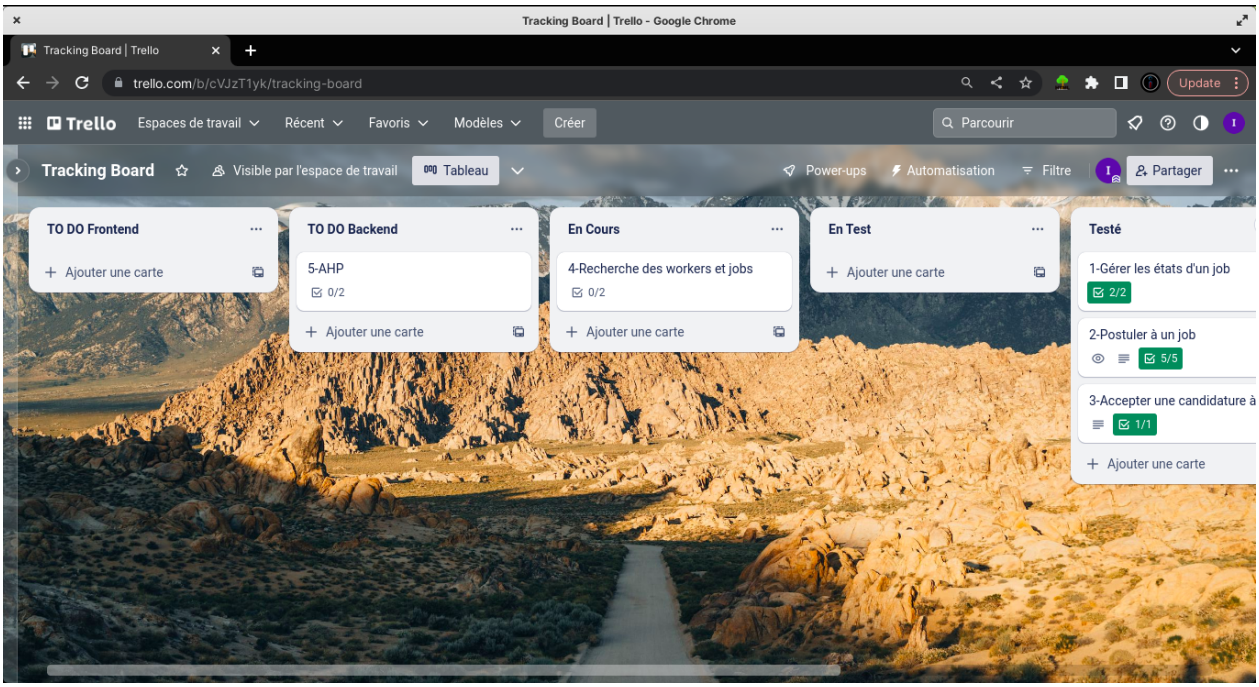
		role in the system				
02	As a fire fighter unit, I should be able to get notifications of a fire disaster in order to be able to respond	When a fire fighter logs into system, they can receive emergency fire request/Notification	02	15	1	16
03	A user can connect to the platform to apply for a job	I would be possible to apply,to apply, to list the applicants and to delete an applican	03	10	1	12
04	A user can modify his profil	Provided that the latter is accepted	04	10	2	13
05	For a user display all these jobs	Jobs will be listed by state	05	11	1	12

	with their status					

6.Sprint Backlog

release	sprint	ID of users stories	period
Release: API-Gateway	Sprint1	1.2.3	1st June - 20 june
	Sprint2	3.5	21 June-30 June
Release2:mobile Application	Sprint3	6.8	1st July- 15 July
	Sprint4	3.7	16 July -25 July

The tracking of the different development tasks was done using the online tool Trello.



An overview of our sprints tracking board

6. Methodology

a. Architecture of your system

i. Architectural Diagram

ii. Description of Architecture

Client: The client is a web client, that any user can use to access to the different features of the platform. It uses HTTP requests to communicate with the backend on which it will realize different requests types : GET, PUT, DELETE, POST.

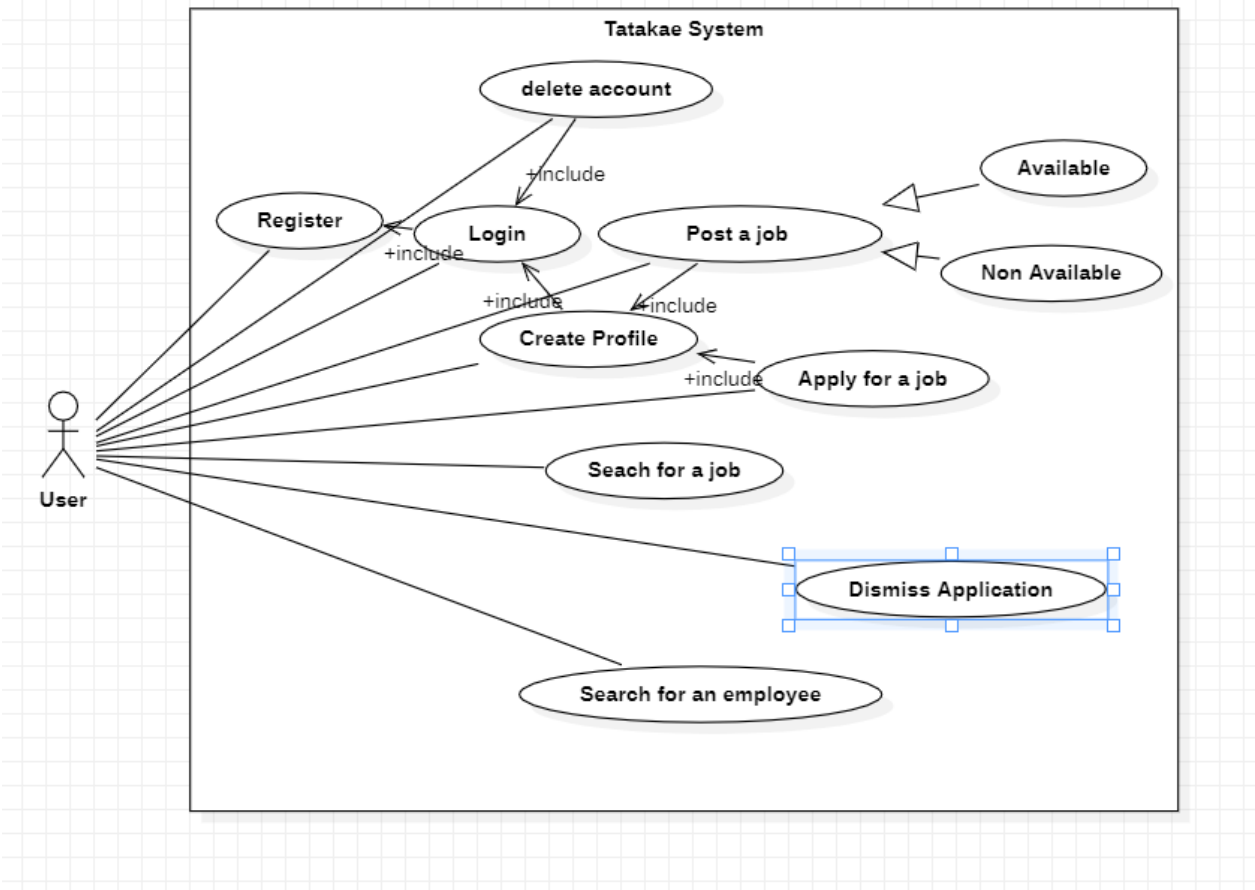
Backend Service: The backend service serves to link the client and the database. It contains the business features of the system. It interacts with the frontend by providing a REST api, build with the Backend Framework FastAPI.

iii. Architectural Drivers

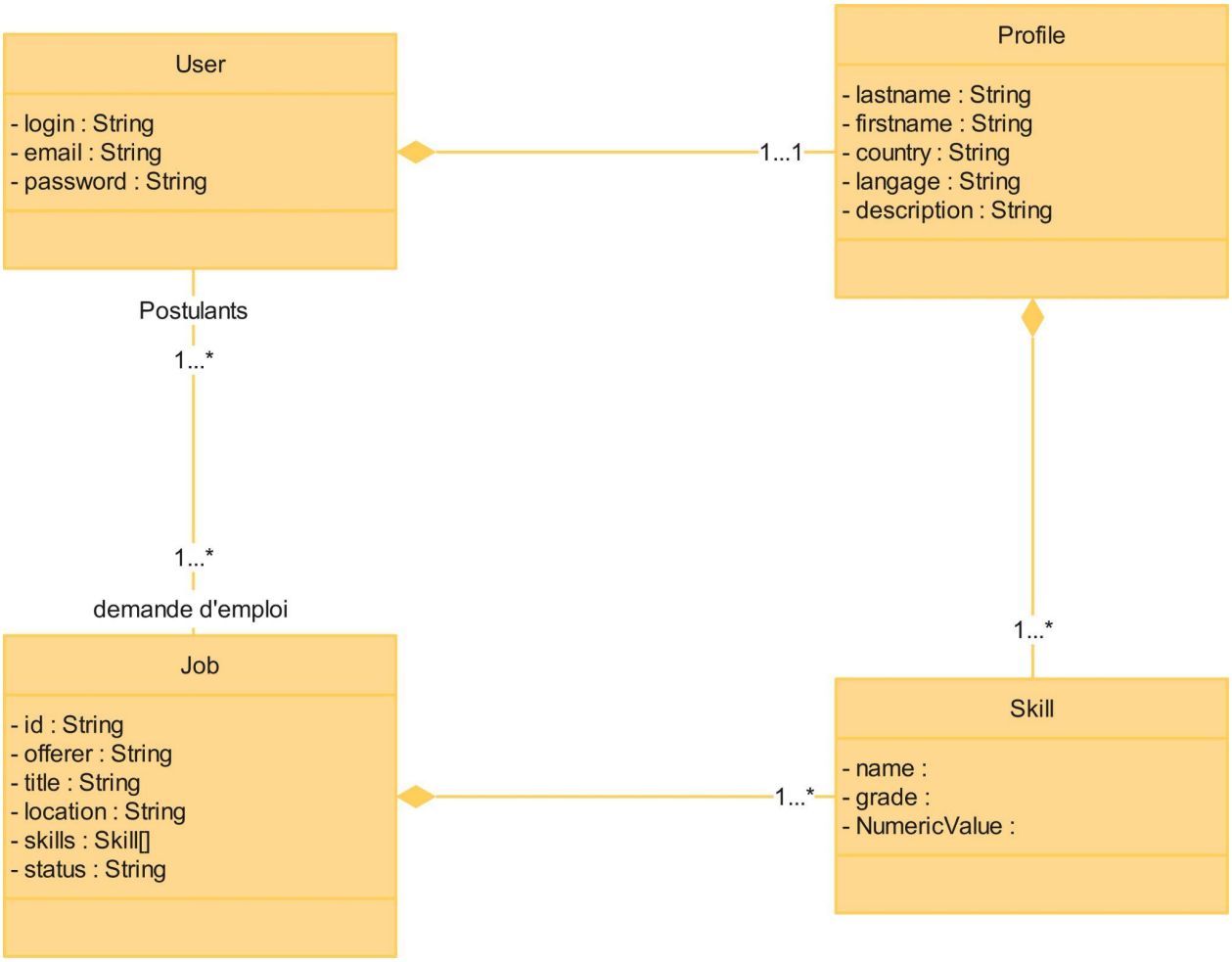
b. Model of your system

iii. Model UML

1. Use case diagram



2. Class diagram



iv. Mathematical Model

```python
@router.get('/user/{job_id}', response_model=List[ProfileModel])
def get_recommended_profiles(job_id:str, user_login=Depends(get_current_user)):
    job = client['Jobs'].find_one({'_id' : ObjectId(job_id)})
    if not job:
        raise HTTPException(status_code=status.HTTP_404_NOT_FOUND, detail="Unexistant job")

    job = Job(str(job['_id']),job['offerer'],job['title'],job['description'],job['location'],[Skill(skill['name'],skill['grade'],skill['numeric_va

    profiles = [Profile(owner = profile['owner'], skills = [Skill(skill['name'],skill['grade'],skill['numeric_value']) for skill in profile['skills

    sorted_profiles = sorted(profiles, key=lambda profile: len(intersection(flatten_skills(job.skills),flatten_skills(profile.skills))), reverse=Tr
    sorted_profiles_dict = [profile.__dict__ for profile in sorted_profiles]
    return sorted_profiles_dict

def flatten_skills(skills:List[Skill]):
    return [skill.name for skill in skills]


def intersection(list1, list2) -> List:
    return list(set(list1) & set(list2))
```

The mathematical model of our system is based on **sets.** In order to get the most interesting profiles for a given job, we evaluate the corelation between the skills of a worker and the skills required for a job. These two are considered as sets.

To calculate the corelation, we evaluate the **intersection** between the two skills sets, and then, we sort the different workers, depending on size of the intersection.

The bigger the intersection is, the better the profile.