Partial Project Delivery

1st Juan Diego Lozada
Systems Engineering
Universidad Distrital
Bogotá, Colombia
20222020014

2nd Edison
Systems Engineering
Universidad Distrital
Bogotá, Colombia
20222020023

***Abstract*—**
The goal project aims to clone an application that exists but create all from the beginning, implementing all the concepts learnt on class such as, design patterns, programming oriented-objects, good practices like SOLID and combine others concepts of classes seen previously like databases foundations. On the first check we need to create all the diagrams with the purpose to make a clear idea about how and what steps follow to make the project in a organize way. After that the code is the part more simple, due to the simplicity of python, but the project it cannot be only on python, applying URL we could connect other files that have other lenguages. How is a big project is important start to use Github and know the basic things for now, e.g, how to pull,push,commit. This tool is one of the most important for a programmer, show the commit and changes that all the collaborators were doing during the process. Despite of look like a big problem clone Pinterest, we realize that if you organize and try to make more simple all is not too hard

***Index Terms*—**

I. Introduction

Stakeholders

The stakeholders are the people who will be participating in an active way on the app, impulsing the app to be better, those are:

1) Users

The users improve the app, making the app more visible, generating posts that serve to the rest of the community, making more attractive the "boards", where the people get motivation with ideas found it on it. The users can upload and create boards and pins as they want (respecting rules)

2) Manager

The manager will be checking the correct function of the app-user, where the users must respect the rules that the app contain, for that reason in the business rules on *Readme* from the folder Backend, show the rules that all users must follow. Following those rules, the app will be for everyone

Business Model

The main concept of our app is to help users discover photos shared by others on various boards Through these visual collections, individuals can find inspiration and ideas for their own projects and plans.

The purpose of creating this app is to assist people who encounter difficulties in generating ideas, experiencing moments where their minds go blank

Imagine having all communities gathered in one place, where users can easily explore and discover photos relevant to their interests. Whether it's cooking, art, fashion, home decoration, or any other passion, our platform ensures users can find images with their preferences.

Our app's primary goal is to offer a seamless experience, allowing users to dive into a world of visual inspiration with just a few clicks. By providing access to a vast collection of high-quality images, we aim to reignite users' creative spark and empower them to bring their ideas to life.

Tools to use

The tools we going to use for the build of the app are:

As a main programming lenguage will be python, due to its simplify and his libraries who can complement the app successfully, such as Pandas, Numpy and SQLalchemy. The last one is one of the most interesting libraries we will use, SQLalchemy is an ORM (Object Relational Mapper), which serve to map and convert data between oriented objects and relational data. We checking if it is necessary use another library such as Faker, which generate database with fake information with the purpose to fill the table and test properly the funcionality of the program. Meanwhile, we use some tools we manage previously due to database foundations, so, we use PowerDesign to make the Diagram ER and generate an file SQL for create the tables and relationships on PostgreSQL. We test the tables making queries of *INSERT*

User Stories

The user stories help to know what are the objectives that people want for the app.

As a user, I want to see boards, so what I can get inspiration.
As a user, I want upload pins, so what I can share my art.
As a user, I want have an account, so what I can save my boards.
As a user, I want to see the catalogs, so what I can see other people's work.
As a manager, I want to see all the users have the app, so what I can make analysis about it.
As a manager, I want to see boards, so what I can decide delete depending on business rules.
As a manager,I want to have an special account, so what I can have special permissions such as delete or view the content of users.

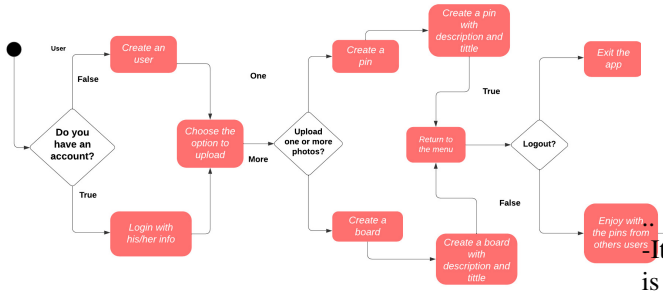
II. Methods and Materials

The steps we use to organize and design the project is through diagrams that let us understand and interpretate how the project will been build

The diagrams we used are:

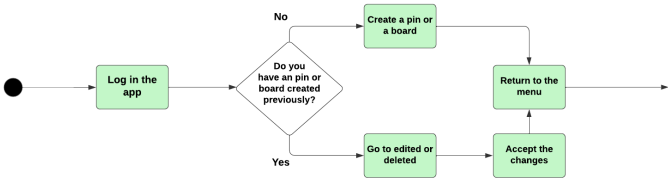
- **Activity Diagram**

1) The user login and upload a pin or create a board



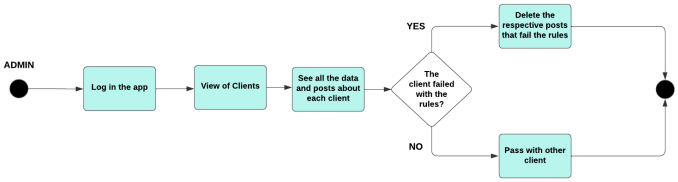
.

2) The user want to delete a photo that upload previously



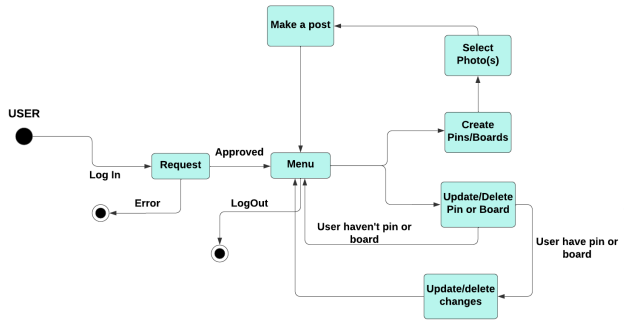
..

3) The manager will see the clients of the app

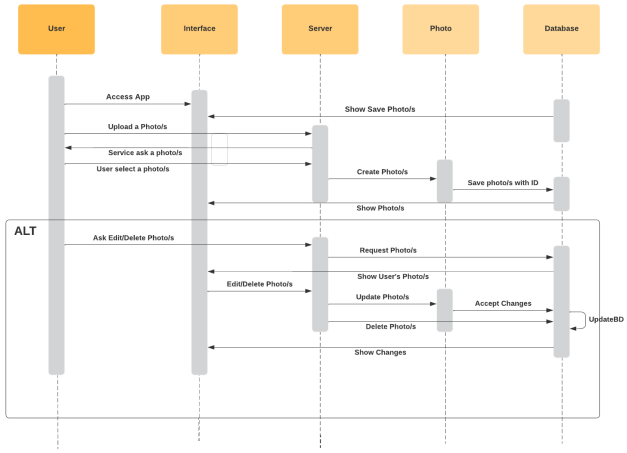


..

The Activity Diagram complete of the users is the next one:



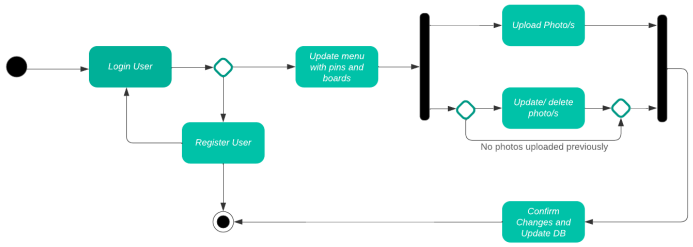
.. - **Sequence Diagram**



..

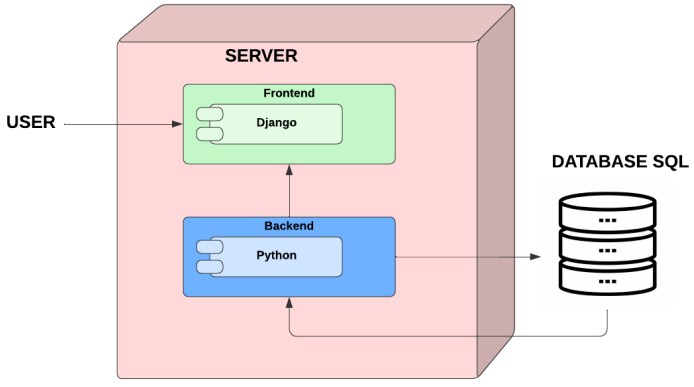
- It's necessary highlight that photo is the same Pin, but Photo is for the people understand more easy the business model.

- **State Diagram**



..

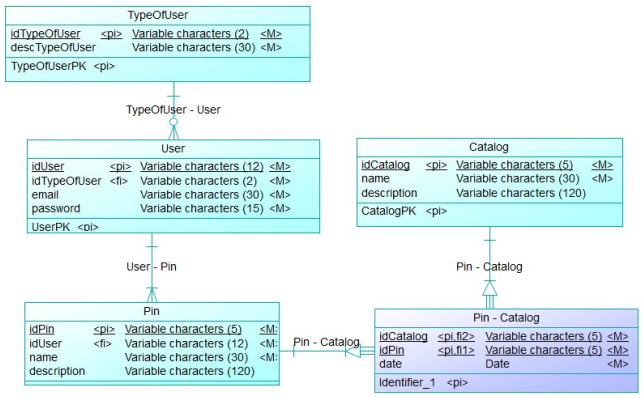
- **Deployment Diagram**



..

- Keeping in mind the time, the frontend part will be as an optional part.

- **ER Diagram**



..

- The table *TypeOfUser* will be an entity without more changes, due to, the only two type of users on the app will be Manager and Users, but, in a future companies may be part of the business.

- In the table *User* will be all the information of the users and managers

- The entity *Pin-Catalog* is the solution of the problem many-to-many between *Pin* and *Catalog*

Those diagrams let us guide how to make the scability for the develop of the program, also, getting the idea from the beginning is more easy find details for fix and make the code more clean

III. Experiments and Results

After develop all the necessary diagrams with the purpose to understand the business model and the steps we are going to take for the creation of the app, the next step is test, check if the diagrams and the database are created correctly.
First for the testing we insert some data on the BD

```
-- Insert data into table TYPEOFUSER:
INSERT INTO TYPEOFUSER (IDTYPEOFUSER,
DESCTYPEOFUSER)
VALUES ('T001', 'Client'),
       ('T002', 'Manager');
```

On the next link we put the queries we use to insert values

https://docs.google.com/document
/d/1WNIhCxo71T7ppMfgrrAYhSqzK5gKUje1B
Pa1vRLPVb4/edit?usp=sharing

It's important know information will be persistent at the moment to make the queries, such as:

Show all the users:
‘‘‘ **Consult 1:**

```
SELECT *
FROM "USER";
```

Consult 2:

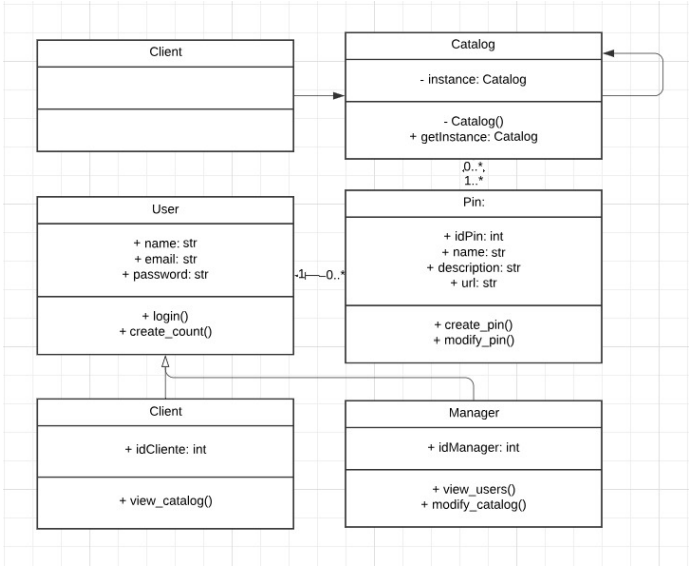
```
SELECT c.IDCATALOG, c.NAME AS CATALOG_NAME,
c.DESCRIPTION AS CATALOG_DESCRIPTION,
p.IDPIN, p.NAME AS PIN_NAME,
p.DESCRIPTION AS PIN_DESCRIPTION, p.URL
FROM CATALOG c
JOIN USER u ON c.IDCATALOG = u.IDCATALOG
JOIN PIN p ON up2.IDPIN = p.IDPIN;
```

Consult 3:

```
SELECT *
FROM PIN;
```

‘‘‘

Can we see, the database has already the queries to use in the program and the information persistent on the database is *TypeOfUser* as we said previously, allowing the database flexible and scalable, taking account the diagram and we see how the queries are correct we could make a diagram class based on:



..

The use of the Singleton pattern for the Catalog class is due to this class assuming the responsibility of managing the catalog of available pins in the system, and by implementing this pattern, it ensures the existence of a single instance of Catalog throughout the system. This provides several benefits, which are:

Instance Control: By restricting the creation of instances of the Catalog class to a single instance, the possibility of creating multiple instances that could lead to inconsistencies in the catalog data is avoided.

Global Access: The Singleton pattern provides a global access point to the Catalog instance, facilitating its use in different parts of the system without the need to pass the instance

from one object to another.

Resource Efficiency: By having a single instance of Catalog in memory, the system's resource consumption is reduced since multiple copies of the same information are not needed.

Data Consistency: By managing all operations related to the catalog through a single instance, data consistency is ensured, and potential update conflicts between different instances of the catalog are avoided.

After, in the code will implement more principles such as SOLID, design patterns structurals and design patterns behaviorals

IV. Conclusion

- We realized that think first on the structure about all the system that compose an application is better than start to code at the beginning, first in every project someone need to interpret the functionality of the system and then analyze with which tools someone can use
- Python has a lot of useful libraries, so, in the project there will be, like numpy, random, SQLalchemy, faker. - The design patterns help to reduce problems in a future, when the system start to receive many users and requests. Obviously, it is not necessary use them but it's a good guide to solve problems
- On the other hand, the application of version control through Git is revealed as a vital practice in software development. Allowing precise tracking of changes made to the code, facilitating collaboration between developers and ensuring the integrity of the project over time. Additionally, it offers the ability to revert unwanted changes, experiment with new functionality in separate branches, and maintain a clear history of software evolution.

References

[1] Pinterest, What is pinterest (2024) https://help.pinterest.com/es/guide/all-about-pinterest

[2] Atlassian, User Stories (2022) https://acortar.link/ymjOC9

[3] Unifed Model Lenguage, What is UML? (2022) https://www.uml-diagrams.org/

[4] Singleton (2024) https://refactoring.guru/es/design-patterns/singleton