OJ Logo

Received XX Month, XXXX; revised X Month, XXXX; accepted XX Month, XXXX; Date of publication XX Month, XXXX; date of current version XX Month, XXXX.

Digital Object Identifier 10.1109/OAJPE.2020.2976889

The creation of a posting network platform

Tomás Cárdenas Benítez, 20221020021
Juan Jesus Poveda, 20202020128
[1]Universidad Francisco Jose de Caldas

Software Designing

ABSTRACT In this project, it will be developed and explained the creation of an social media application/platform where an user can make post about a variety of topics, share multimedia files (images, videos, GIF, among others), interact with other users from the app, while also having moderation and regulation from the content posted, in order to make a respectful and enjoyable space and make new users more comfortable when joining the app.

INDEX TERMS Platform, Posting, User, Software, Pattern

I. INTRODUCTION

In the following paper, we will show the different decisions made, the structural design, and various analyses conducted during the creation of a posting platform network. The main focus is on the development to an application similar to Twitter/X, while improving the mistakes in the existing app.

First and foremost, we live in a society where internet access is not only necessary but arguably essential, given the various opportunities it provides and the doors it opens in diverse aspects of human life. For instance, it facilitates research on various topics without needing in-depth knowledge, thanks to resources like Wikipedia and Encyclopedia Britannica. It enables global communication through apps like WhatsApp and Telegram. It assists with navigation via apps like Google Maps and Waze. The list of practical applications of the internet in real life is extensive, and we can continue to explaining the importance of the internet, but that is not the main focus of this paper.

In this case, we will be discussing a ”Twitter clone”. To provide some context, Twitter is one of the most popular sites for posting various activities, allowing others to react to them.On Twitter, you have the freedom to use posts however you want. For example, a user can write a text expressing their opinion on a current controversy, or simply use it to share what’s happening in their day.

Twitter also offers the option to post different media files, including images, videos, GIFs, and audio files. This gives users the opportunity to express themselves freely in the way they prefer. Another advantage Twitter offers is the ability to choose who to follow or not, including muting content, blocking users, and reporting users/posts when they feel the Terms of Service (ToS) of the app have been violated. Unfortunately, not everything is as it seems. With the freedom of speech, many users have taken it too far, allowing hate speech to spread across the app and tarnishing Twitter’s reputation. Closing the idea, the purpose of making this project is basically trying to make a better managed posting network, where the users feel comfortable and safe, while expressing their opinion, while also being moderated.

II. THE PLATFORM DESIGN: MATERIALS AND METHODS

In order to make possible the creation of the platform, there should be some things that are necessary for, forgive the redundancy, making the platform, one of them can be the IDE that will be used for the project, in this case, we will be using the python language (v. 3.11.9), while making the use of different interpreters being those Visual Studio Code (easier and lighter program to code) and PyCharm (easier way to generate a virtual environment if it is not possible on VS Code).

With that being said now we use one of the most important libraries for this project, Faker, this library helps, as its name says, generating fake data (names, emails, directions, among others), this library will help when making the different users and posts, the application of this system can be generated like this:

FIGURE 1. **Creation of comments and possible passwords**

```python
def create_person_name():
    return fake.name()

def create_person_post():
    return fake.sentence()

def generate_likes():
    return random.random()

def generate_saves():
    return random.random()

def create_comments():
    return random.choice(comments_list)

def create_birhtdate():
    return fake.date_of_birth()

def create_password():
    return random.choice(password_list)
```

FIGURE 2. **Definition of functions based on the use of Faker and Random**

Now, with the use of the lists and the defined functions we can create a method that generates posts. This function is called *add_fake_users()*, it requires a number of users it can add to the JSON, this function asks **Faker** to generate different sentences, usernames and birth-dates, while also asking **Random** for a password and comments (from the lists created), and asking to generate a number of likes and saved posts, the explained can be seen in the image bellow:

```python
def add_fake_users(num_users=10):
    users = []

    for _ in range(num_users):
        user = {
            "username": create_person_name().replace(" ", "_"),
            "password": create_password(),
            "profile": {
                "bio": fake.text(),
                "pfp": fake.image_url(),
                "birthday": str(create_birhtdate()),
                "posts": [
                    {
                        "text": create_person_post(),
                        "likes": generate_likes(),
                        "comments": [create_comments() for _ in range(random.randint(0, 5))],
                        "saved": generate_saves(),
                    } for _ in range(random.randint(1, 5))]
            }
        }
        users.append(user)

    with open("users.json", "w", encoding="UTF-8") as file:
        json.dump(users, file, indent=4)
```

FIGURE 3. **Function *add_fake_users()* and its implementation on the project**

This method will facilitate the creation of users when testing the platform and its different functions. Going back to the project, when thinking about making a media platform, first there should be created a user, this user should have a username, a password and a bio, formed by a description, a profile picture, a birthdate and a list of posts made by its user; with that being said the project has the basic concept of a social media posting application, then it can be made the creation of different methods that help making the platform more enojoyable for the user. With this explained, we can understand the model of creating the platform, that by building a UML diagram presented next:

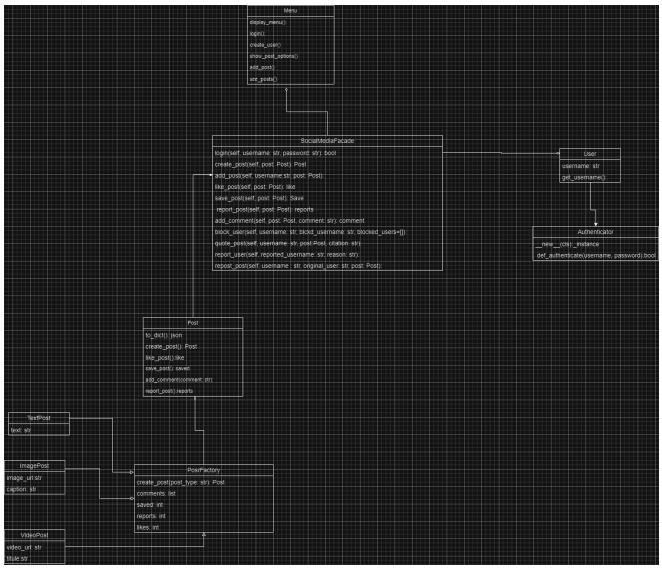

FIGURE 4. **Function *diagram representing the relationships in the different classes***

In the diagram above, it can be seen the UML diagram of this project. When first thinking about a platform, its is always better for the user to only interact with a part of the program, especially when the project is really big, this reason is why in the project its decided to aplicate the use of a creational design pattern, this being the Facade Design; this pattern explains that all the logic of the program is irrelevant to the user, therefore the user only interacts with the GUI (since this project doesnt have a GUI, it can be interpreted as the menu.py), and the processes are thrown ”behind” the GUI, like a Facade. Another thing of consideration is the only use of Facade in the program, the reason of this can be argumented in various aspects: the potency of the machines used for the programs werent as optimal as it should be, so reforming the application took extended periods of time; the use of other patterns incremented the aplication complexity, that was one of the main reasons, since it was argued about using singleton (create multiple threads of user processes) and command (assign tasks), but when trying to apply them, it became really complex to reform the application (especially command), and one of the main principles of pattern dessigns is it isnt really necessary to apply them if it costs the application resources and time, which happened with these patterns.

Below are the relationships that they have in each relationship table and the goals of each one explained, using the CRC by or which one presents the following tables:

User	
Get_username: request username	Authenticator y SocialMediaFacede

FIGURE 5. *Diagram representing the class User*

Authenticator	
__new__: requests proof of the program and creates it if it does not exist y authenticate: review user information to enter	User

FIGURE 6. *Diagram representing the class Authenticator*

Post	
to_dict: returns the action performed, create_post: generates new Post data, like_post: stores the number of likes, save_post: add a save to the post, add_comment: add a comment to the post, report_post: add a save to the post	Authenticator y SocialMediaFacede

FIGURE 7. *Diagram representing the class Post*

PostFactory	
create_post: generates the posts to publish	SocialMediaFacede TextPost, ImagePost, VideoPost

FIGURE 8. *Diagram representing the class PostFactory*

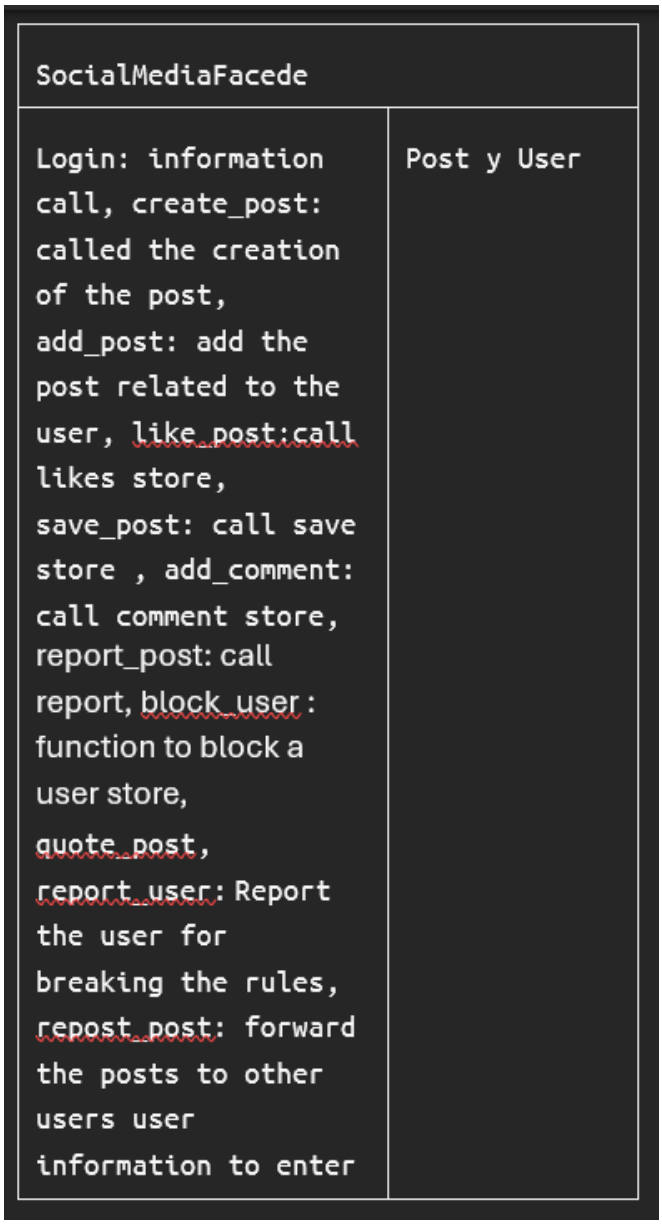

FIGURE 9. *diagram representing the class SocialMediaFacedes*

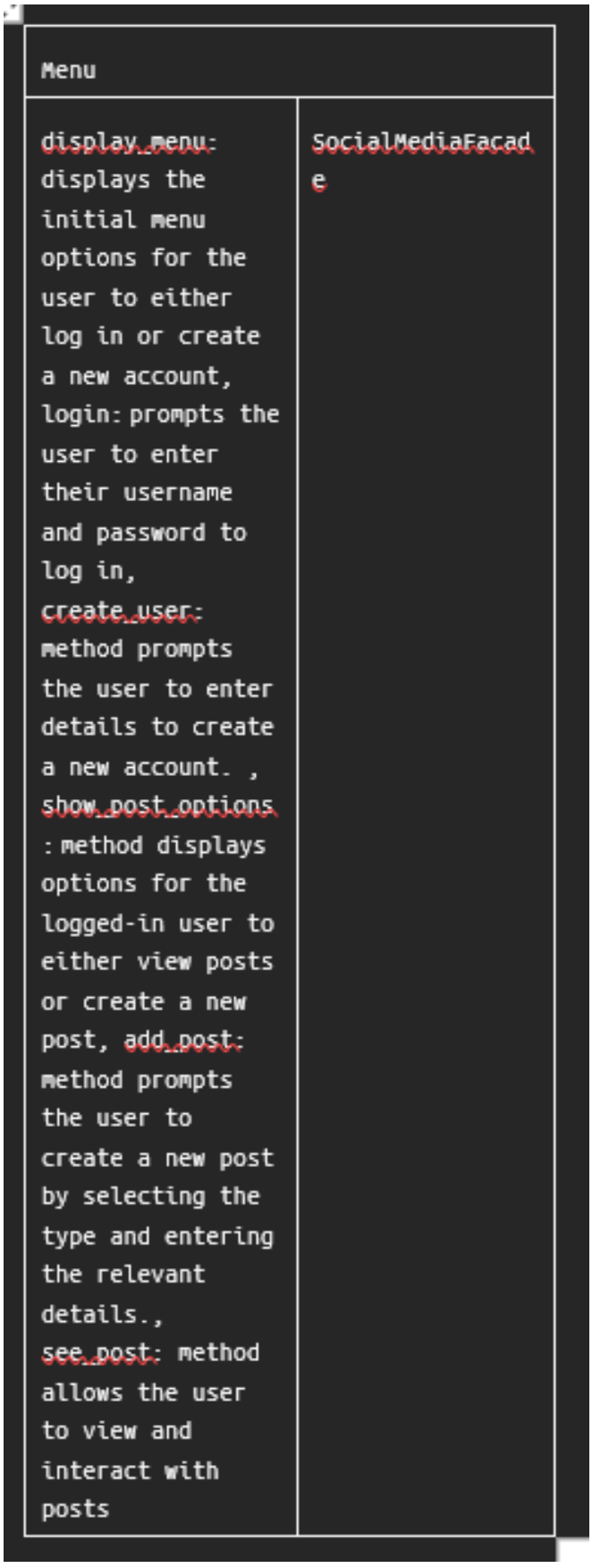

FIGURE 10. *diagram representing the class Menu*

III. EXPERIMENTS AND RESULTS

To better emphasize the operation, we give examples of consoles, which are divided into two:

```
Bienvenido al sistema de redes sociales!
1. Iniciar sesión
2. Crear usuario
0. salir
Seleccione una opción: 2
Ingrese un nombre de usuario: juna
Ingrese una contraseña: juan
Ingrese su biografia:  juan
Ingresa la direccion de tu imagen de usuario: as
Ingrese su cumpleaños: 2000-12-23
Usuario posible
Usuario ya existente
Usuario añadido exitosamente
Usuario creado exitosamente.
Bienvenido al sistema de redes sociales!
1. Iniciar sesión
2. Crear usuario
0. salir
Seleccione una opción: 1
Ingrese su nombre de usuario: juan
Ingrese su contraseña: juan
¡Inicio de sesión exitoso!
```

FIGURE 11. *home and registration screen*

The start of loading the program and the completion of its registration are reflected, confirming the validations at each step, ending with the entry to the section start space.

```
1. Iniciar sesión
2. Crear usuario
0. salir
Seleccione una opción: 1
Ingrese su nombre de usuario: juan
Ingrese su contraseña: juan
¡Inicio de sesión exitoso!
¿Qué desea hacer?
1. Ver posts
2. Crear un nuevo post
Seleccione una opción: 2
Seleccione el tipo de publicación que desea crear:
1. Publicación de texto
2. Publicación de imagen
3. Publicación de video
Ingrese el número correspondiente al tipo de publicación: 1
Ingrese el texto de la publicación: jhgue
Post añadido exitosamente
Publicación agregada exitosamente.
```

FIGURE 12. Function *home and registration screen*

The previous image shows the saving of the post and the generation of the post by the user

```
{
    "username": "juan",
    "password": "juan",
    "profile": {
        "bio": "soy una persona",
        "pfp": "12233",
        "birthday": "2002-05-03",
        "posts": [
            {
                "text": "jhgue",
                "likes": 0,
                "comments": [],
                "saved": 0
            }
        ]
    }
},
```

FIGURE 13. Function *home and registration screen*

Mainly you see a json file that was generated by the program, which is related to the previous image of the user registration program, thus showing the validity of the program and its operation.

IV. CONCLUSIONS

- The simplest relationships are necessary because everything depends on them and they cannot be altered.
- Virtual environments like GitHub make it easy to move programs and collaborate properly among programmers. also generating sufficient changes to solve problems, thus avoiding problems of loss of information since each one has a copy of their Code in their own cloud
- Not having the necessary organization of the program can generate headaches, but by managing the fundamentals of the program you can understand the process very easily.