ST. TERESA'S COLLEGE (AUTONOMOUS), ERNAKULAM
AFFILIATED TO MAHATMA GANDHI UNIVERSITY

PROJECT REPORT ON
SOCIOSPHERE

In partial fulfillment of the requirements for the award of the degree of

BACHELOR OF SCIENCE IN
COMPUTER APPLICATIONS [TRIPLE MAIN]

Submitted By

AGNES P J

III BSc Computer Applications [Triple Main]

Register No: SB21CA003

Under the Guidance of
Assistant Professor

Ms. Arunima P S

DEPARTMENT OF COMPUTER APPLICATIONS

2021- 2024

ST. TERESA'S COLLEGE
(AUTONOMOUS)
ESTD 1925

CERTIFICATE

This is to certify that the project entitled "**SOCIOSPHERE**", is a bona-fide record of the work done by **AGNES P J** [Reg. No. SB21CA003] during the year 2021-2024 and submitted in partial fulfillment of the requirements of the degree of Bachelor of Science in Computer Applications (Triple Main) under Mahatma Gandhi University.

Head of the Department

ST. TERESA'S COLLEGE
DEPT. OF COMPUTER SCIENCE
ERNAKULAM

Internal Examiner

External Examiner

Date: 21/3/24

DECLARATION

I, AGNES P J (Register no: SB21CA003), B.Sc. Computer Applications [Triple Main] final year student of St. Teresa's College (Autonomous), Ernakulam, hereby declare that the project submitted named "SOCIOSPHERE" for the Bachelor's Degree in Computer Applications [Triple Main] is my original work. I further declare that the said work has not previously been submitted to any other university or academic body.

Date: 21/03/2024

Place: ERNAKULAM

Agnes

AGNES P J

ACKNOWLEDGEMENT

Firstly, I thank God, the Almighty, with whose blessings, this project has been successfully completed.

I would like to convey my heartfelt gratitude to Rev. Dr. Sr. Vinitha (CSST) Manager, Director Rev. Sr. Emeline (CSST) and Principal Dr. Alphonsa Vijaya Joseph for providing me with this wonderful opportunity to work on a project with the topic Sociosphere.

I would like to express my profound gratitude to the Head of the Department of Computer Applications Ms. Remya C.J, my project guide Ms. Arunima P S and all other faculty of the department for their contributions to the completion of my project. The completion of the project would not have been possible without their help and insights.

I would also like to thank my project guide at LCC Computer Education Institute, Ms. Akhila for training me well to develop this project.

Finally, I take this opportunity to thank all of them who has directly or indirectly helped me with my project.

AGNES P J

SYNOPSIS

Sociosphere outlines the implementation of a comment moderation system within a social media app, aimed at preventing the posting of abusive comments in the comment section. Sociosphere revolutionizes social media engagement by integrating sentiment analysis technology to curate a positive and supportive online environment. Through advanced sentiment analysis algorithms, Sociosphere categorizes user-generated content into positive, negative, or neutral sentiments, enabling proactive moderation to prevent the dissemination of negative comments. This innovative approach fosters constructive discourse among users, promoting mutual respect and understanding. Sociosphere's commitment to fostering positivity and meaningful interactions enhances the overall user experience, cultivating a vibrant and inclusive online community.

Contents

1.INTRODUCTION

1.1 About Project

Sociosphere is a dynamic social media platform designed for modern connectivity and expression. With its intuitive interface and vibrant community, Sociosphere offers users a seamless experience to share moments, connect with friends, and discover new interests. From captivating visual content to engaging discussions, Sociosphere fosters a welcoming environment where individuals can express themselves authentically and forge meaningful connections. Sociosphere empowers users to curate their digital space and engage in enriching interactions while prioritizing user safety and well-being.

1.2 Objectives of the Project

The objective of the Sociosphere project is to develop a user-friendly social media platform that promotes positive interactions and cultivates a safe and inclusive online community by implementing effective comment moderation features to prevent the posting of offensive or abusive comments..

2. SYSTEM ANALYSIS

2.1 Introduction

The primary objective of system analysis is to identify any problems, weaknesses, or inefficiencies in the system and to develop strategies for improving its functionality. This may involve gathering and analysing data, conducting interviews and surveys, creating mathematical models, and developing computer simulations to test different scenarios. It can also be used to design and implement new systems or to integrate existing systems with new technologies or processes.

Overall, system analysis plays a critical role in improving the performance, reliability, and effectiveness of complex systems, and it continues to be an essential tool for organizations and businesses seeking to stay competitive in today's fast-paced and rapidly changing environment.

2.2 Existing System

Existing social media platforms have struggled to effectively prevent the proliferation of negative comments, creating an environment where abusive, offensive, and harassing content often goes unchecked. Despite efforts to implement comment moderation systems, these platforms continue to face challenges in adequately identifying and addressing harmful comments. The reliance on user reporting mechanisms places the burden of content moderation on individuals, leading to inconsistencies and delays in addressing harmful content. Consequently, the failure to effectively prevent negative comments not only contributes to a hostile and unwelcoming online environment but also poses significant risks to user well-being and mental health.

2.3 Proposed System

The proposed system is a negative comment prevention system designed for a social media web application. It utilizes sentiment analysis techniques to automatically evaluate the sentiment of user-generated comments. By analyzing the sentiment of user-generated

content, the system aims to identify and mitigate the impact of negative sentiments in discussions and interactions among users. The system utilizes the VADER (Valence Aware Dictionary and sEntiment Reasoner) sentiment analysis tool, which is implemented through the SentimentIntensityAnalyzer class from the vaderSentiment library. This tool is specifically designed to analyze sentiments expressed in social media texts. The polarity_scores() method provided by the SentimentIntensityAnalyzer class is used to analyze the sentiment of text data. It calculates a sentiment score for the text, indicating the intensity of positive, negative, and neutral sentiments present. By assessing the sentiment of user comments, the system can classify them as positive, negative, or neutral, allowing it to identify potentially harmful or negative content. Once the sentiment of a comment is analyzed, the system can take preventive actions if negative sentiment is detected. The system may implement rules or algorithms to automatically filter out or flag comments with negative sentiment before they are published or displayed to other users.. Overall, the proposed negative comment prevention system leverages sentiment analysis techniques to assess the sentiment of user-generated content, with the aim of creating a more positive and constructive social media experience for users.

2.4 System Specification

A system specification is a formal document that outlines the functional and non-functional requirements for a proposed system. It provides a comprehensive description of the system's behaviour, features, and limitations. A system specification is used as a roadmap for the design, development, and implementation of the system. System specification specifies the hardware and software configuration of the new system. It helps to define the operational and performance guidelines of the system.

2.5 Operating System

The operating system required for proper execution of the system is 64 -bit Microsoft® Windows® 8/10/11. Windows 11 includes improved network, application, and Web services. It provides increased reliability and scalability, lowers your cost of computing with powerful, flexible management services, and provides the best foundation for running business application.

2.6 Languages or Software Packages

- **Visual Studio Code** : Visual Studio Code (VS Code) is a popular source-code editor developed by Microsoft. It's widely used for programming and web development, known for its flexibility, extensive customization options, and a rich ecosystem of extensions. VS Code supports a wide range of programming languages and frameworks, and it's available for Windows, macOS, and Linux. Many developers appreciate its performance, intuitive interface, built-in Git integration, and the ability to collaborate with others using Live Share. Overall, it's a versatile tool for developers of all levels.

- **HTML**: Hypertext Markup Language or HTML is the standard markup language for documents designed to be displayed in a web browser. It defines the content and structure of web content.
- **Python**: Python is a high-level, general-purpose programming language. Its design philosophy emphasizes code readability with the use of significant indentation. Python is dynamically typed and garbage-collected. It supports multiple programming paradigms, including structured, object-oriented and functional programming.

- **SQLite**: SQLite is a C library that provides a lightweight disk-based database that doesn’t require a separate server process and allows accessing the database using a nonstandard variant of the SQL query language. Some applications can use SQLite for internal data storage. It’s also possible to prototype an application using SQLite and then port the code to a larger database such as PostgreSQL or Oracle.

  The sqlite3 module was written by Gerhard Häring. It provides an SQL interface compliant with the DB-API 2.0 specification described by PEP 249, and requires SQLite 3.7.15 or newer.
- **Django**: Django is a free and open-source, Python-based web framework that runs on a web server. It follows the model–template–views architectural pattern. It is maintained by the Django Software Foundation, an independent organization established in the US as a 501 non-profit

2.7 Hardware and Software Specifications

Hardware Requirements:

- Main Processor: Intel core i3 or above
- RAM 8 GB or Above
- ROM: 10 GB or above
- Drive: 512 GB SSD space

Software Requirements:

- Operating System 64-bit Microsoft® Windows® 8/10/11
- Programming Language: Python
- Database Management System: SQLite
- Development tool: HTML

3. SYSTEM DESIGN

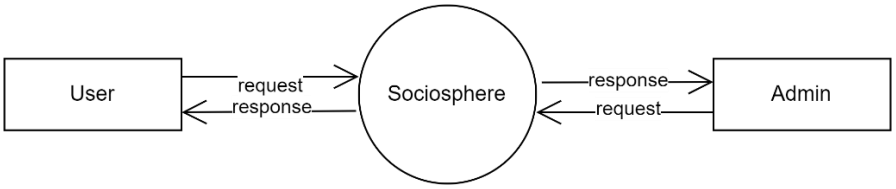
3.1 Introduction

System design is the process of defining the architecture, components, modules, interfaces, and data for a system to satisfy specified requirements. It is a critical step in the software development lifecycle that bridges the gap between the requirements analysis and implementation phases.

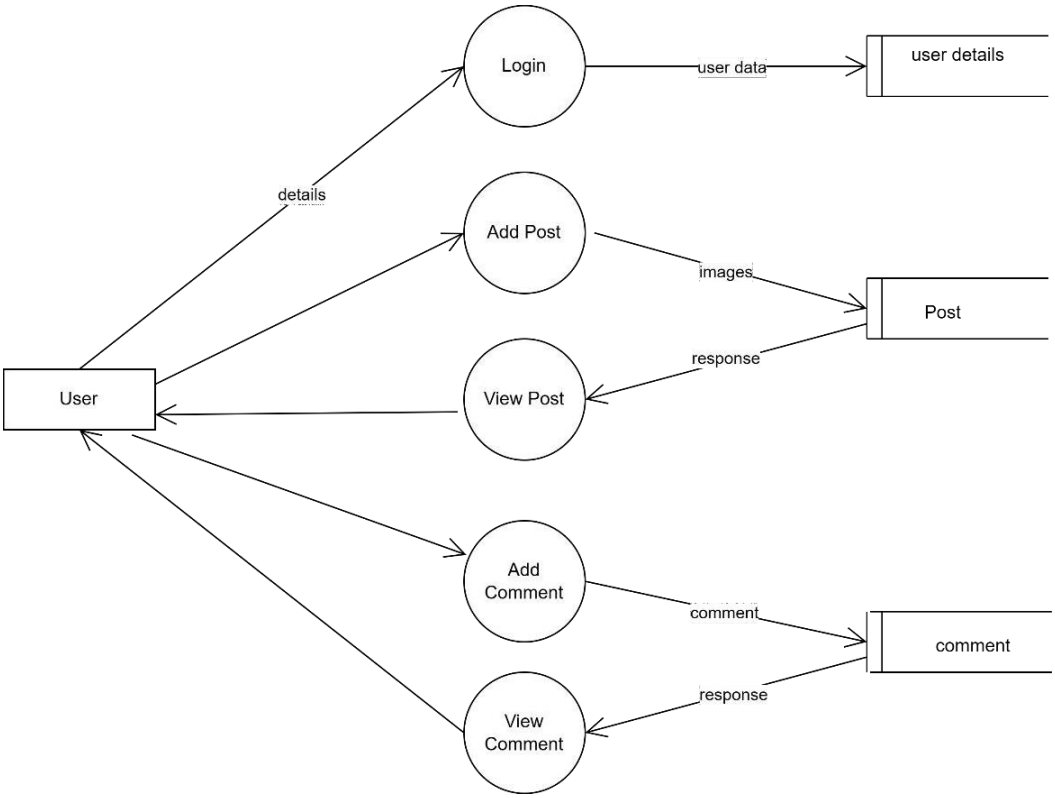
3.2 Data Flow Diagram

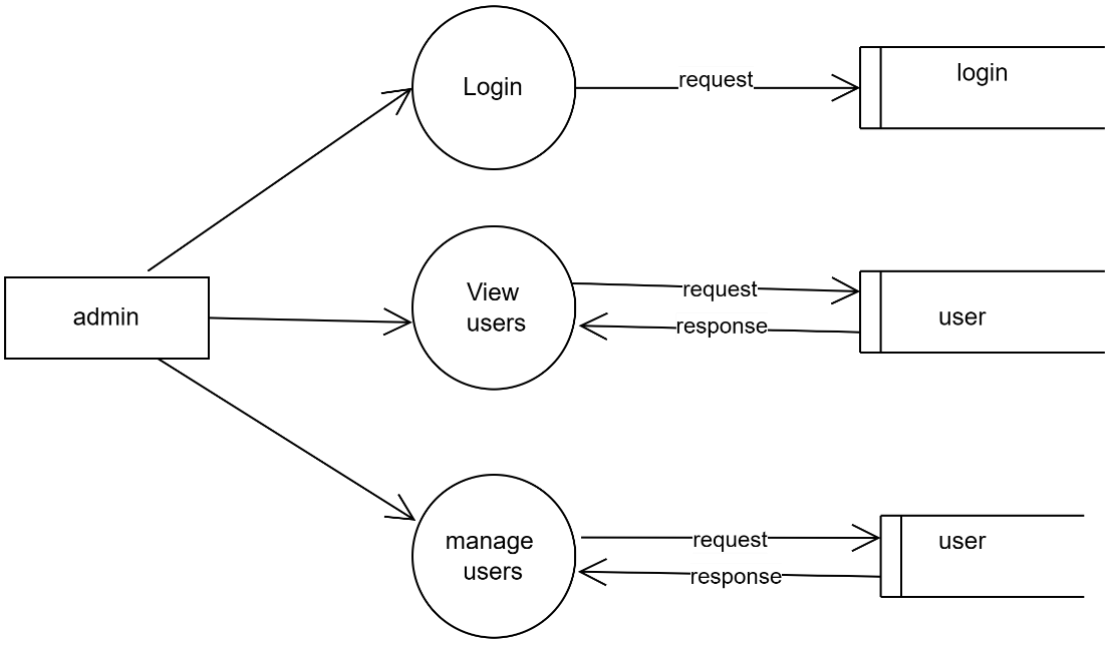
A data flow diagram (DFD) is a graphical representation of the flow of data through an information system. A DFD is often used as a primary step to create an overview of the system, which can later be elaborated. A DFD shows what will be the input of the system as well as the output. It clearly represents where the data will come from and go to, and where the data will be stored.

DFD Level 0



DFD Level 1


Login
user data
user details
details
Add Post
images
Post
response
User
View Post
Add
Comment
comment
comment
response
View
Comment

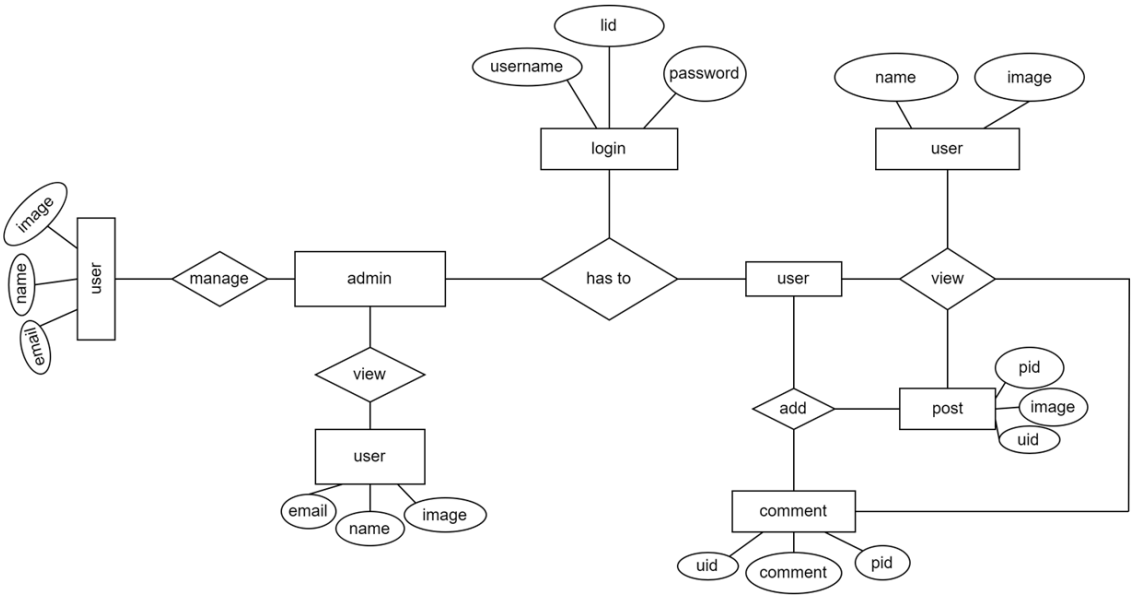

Login
request
login
admin
View
users
request
response
user
manage
users
request
response
user

3.3 E-R Diagram

An E-R (Entity-Relationship) diagram is a visual representation used in database design to illustrate the relationships between different entities in a system. It depicts the logical structure of a database by showing the entities, their attributes, and the relationships between them.

In an E-R diagram:

1. Entities: Entities represent real-world objects or concepts that can be distinguished from one another. Each entity is depicted as a rectangle in the diagram and typically corresponds to a table in the database.
2. Attributes: Attributes are the properties or characteristics of entities. They are depicted as ovals connected to the corresponding entity rectangle. Attributes describe the features of an entity and help define its properties.
3. Relationships: Relationships illustrate the associations or connections between entities. They describe how entities interact with each other in the database. Relationships are represented by lines connecting the relevant entities, with labels indicating the nature of the relationship (such as "belongs to," "has," or "is related to")



3.4 Data Dictionary

A data dictionary is a collection of metadata that provides information about the data used in a database or information system. It serves as a reference guide for data elements and their definitions, as well as their relationships with other data elements in the system. A data dictionary is useful for ensuring data accuracy, consistency, and completeness. It can also help facilitate communication and collaboration among stakeholders who are involved in designing, developing, and maintaining a database or information system.

3.5 Data Design

Database design is the process of creating a well-organized and structured database that efficiently stores and retrieves data. A well-designed database ensures data accuracy, consistency, and completeness and provides an intuitive and user-friendly interface for interacting with the data. Data design is the first design activity, which results in less complex, modular, and efficient program structure. The information domain model developed during analysis phase is transformed into data structures needed for implementing the software. The data objects, attributes, and relationships depicted in entity relationship diagrams and the information stored in data dictionary provide a base for data design activity.

Register

Field	Type	Description
reg_id	int	Primary key
name	varchar	Not Null
email	varchar	Not Null
mobile	varchar	Not Null
password	varchar	Not Null
image	blob	Not Null
loginid	int	Foreign key

Login

Field	Type	Description
login_id	int	Primary key
username	varchar	Not Null
password	varchar	Not Null

Post

Field	Type	Description
p_id	int	Primary key
text	varchar	
image	blob	Not Null
user_id	int	Foreign key
likes_count	int	

Comment

Field	Type	Description
id	int	Primary key
comments	varchar	Not Null
date	date	Not Null
Sentiment	varchar	Not Null
user_id	int	Foreign key
p_id	int	Foreign key

4. SYSTEM DEVELOPMENT

4.1 Introduction

Software Development is the process of analysing, designing, testing, implementation, and maintenance. It is called Software Development Life Cycle (SDLC). Different SDLC include waterfall, prototyping, iterative, incremental, spiral development, rapid application development and agile methodology.

4.2 Process Description

Different processes of each module are given below:

- **Registration**

  The user must enter their name, email id, mobile number and password in the registration fragment.

- **Login**

  The user must enter the existing username and password to log into their account in the login fragment.

- **Post**

  This process represents the action of creating and submitting a new post. When a user creates a new post, it is processed and stored within the system.

- **Comment**

  This process represents the action of creating and submitting a comment on a social media platform. This process involves users interacting with the system by composing and posting comments in response to content shared by others.

4.3 Code Design

```
from django.shortcuts import render,redirect,HttpResponse,HttpResponseRedirect
from django.contrib import messages
from django.contrib.auth import authenticate, logout,login
from .models import *
from datetime import datetime
from datetime import datetime, timedelta
from vaderSentiment.vaderSentiment import SentimentIntensityAnalyzer
sid_obj = SentimentIntensityAnalyzer()



def index (request):
    return  render(request,"index.html")




def userhome(request):
    return  render(request,"user/index.html")

def adminhome(request):
    return  render(request,"admin/index.html")

def logins(request):
    if  request.POST:
        email=request.POST['email']
        passw=request.POST['password']
        print(email,passw)
        user=authenticate(username=email,password=passw)
        print(user,"Hii")
        if user is not None:
            login(request,user)
            if user.type=='Admin':
                print(user.type,"ihygcueycfgueycfg")
                messages.info(request,"Login  Success")
```

```
                return redirect("/adminhome")
            elif user.type =="User":
                print(user.type,"jhbfcuyhb")
                uid=user.id
                request.session['uid']=uid
                messages.info(request, "login  Success")
                return redirect("/userhome")
            else:
                messages.error(request, "Invalid Username/Password")
                return redirect("/login")
        else:
            messages.error(request, "Invalid Username/Password")
            return redirect("/login")
    return  render(request,"login.html")

def register(request):
    if request.POST:
        name=request.POST['name']
        email=request.POST['email']
        password=request.POST['password']
        mobile=request.POST['mobile']
        image=request.FILES['image']


log=Login.objects.create_user(username=email,password=password,type="User",viewpassw
ord=password,is_active="1")
        log.save()


reguser=Register.objects.create(name=name,email=email,password=password,mobile=mobil
e,image=image,loginid=log)
        reguser.save()

        messages.success(request,"Registered   Successfully")
```

```
    return  render(request,"register.html")

def addposts(request):
    uid=request.session["uid"]
    print(uid)
    userid=Register.objects.get(loginid=uid)
    if request.POST:
        file=request.FILES['file']
        text=request.POST['text']

        post=Posts.objects.create(image=file,text=text,tweets="nil",userid=userid)
        post.save()
        messages.success(request,"Post Uploaded")
    return  render(request,"user/Add_posts.html")

def viewpost(request):
    uid=request.session["uid"]
    view=Posts.objects.all()

    tweet=Posts.objects.filter(userid_id=uid)[:4]

    return    render(request,"user/view_posts.html",{"view":view,"tweet":tweet})

def Addcomments(request):
    uid=request.session["uid"]
    print("idddddddddddddddddd",uid)
    logid=Register.objects.get(loginid=uid)
    print(logid.id)
    pid=request.GET.get("pid")
    # current_date = datetime.date.today()
    curdate = datetime.today().strftime('%Y-%m-%d')
    # print(current_date)
    print(curdate)
```

```
    view=Posts.objects.all()
    if request.POST:
        tweets=request.POST['tweet']
        # title=request.POST['title']
        userid=request.POST['uid']
        sentiment_dict = sid_obj.polarity_scores(tweets)
        sentiment=""
        if sentiment_dict['compound'] >= 0.05 :
            sentiment="Positive"

        elif sentiment_dict['compound'] <= - 0.05 :
            sentiment="Negative"

        else :
            sentiment="Neutral"

        tweet=Tweets.objects.create(tweets=tweets,userid_id=logid.id
                                    ,Pid_id=pid,description=tweets,date=curdate,Sentiment=sentiment)
        tweet.save()
        return  redirect("/viewpost")

    return    render(request,"user/add_tweets.html",{"view":view,"uid":uid})




def view_more(request):
    pid  =  request.GET.get("id")
    print(pid)
    view  =   Tweets.objects.filter(Pid_id=pid).exclude(Sentiment="Negative")

    if view:
        tweet = view[0]
```

```
        image = tweet.Pid.image
        print(image)
        print(view, "dfhvidrhvriue")
        return render(request, "user/view_more.html", {"view": view, "image": image})
    else:
        # Handle the case where view is empty, e.g., display an error message or redirect
        # to a different page.
        messages.success(request,"No comments found")
        return redirect("/viewpost")
```

```
def adm_view_users(request):

    view=Register.objects.all()
    return    render(request,"admin/view_users.html",{"view":view})

def delete_user(request):
    id=request.GET.get('id')
    Login.objects.filter(id=id).delete()
    messages.success(request,"deleted")
    return redirect("/adm_view_users")

from django.contrib.auth import get_user_model
from django.shortcuts import render, get_object_or_404
from django.http import JsonResponse
from django.db import transaction

def like_post(request):
    uid = request.session.get('uid')
    post_id = request.GET.get('pid')
    post = get_object_or_404(Posts, id=post_id)
```

```
    user = Register.objects.get(loginid=uid)

    with transaction.atomic():
        # Check if the user already liked the post
        if user not in post.likes.all():
            # User didn't like the post, like it
            post.likes.add(user)

            # Update the likes_count in the Posts model
            post.likes_count = post.likes.count()
            post.save()
            messages.success(request,"liked successfully")
    return redirect("/viewpost",{'liked': True, 'likes_count': post.likes_count})
```

```
def managepost(request):
    uid = request.session['uid']
    user=Register.objects.get(loginid=uid)
    posts = Posts.objects.filter(userid=user)
    return render(request,'user/manageposts.html',{'posts':posts})

def delete(request):
    id=request.GET.get('id')
    Posts.objects.filter(id=id).delete()
    messages.success(request,"Post deleted")
    return redirect("/manage_post")
```

5. SYSTEM TESTING AND IMPLEMENTATION

5.1 Introduction

Software testing is a crucial process in the software development life cycle that involves evaluating the quality and functionality of software applications. The main goal of software testing is to identify defects or errors in the application before it is released to the end-users. The software testing process typically involves several steps, including test planning, test design, test execution, and test reporting.

5.2 System Implementation

Implementation is the action that must follow any preliminary thinking for something to happen. Software/hardware implementations should always be designed with the end user in mind and the implementation process usually benefits from user involvement and support from managers and other top executives in the company. If users participate in the design and implementation of the system, ideally it will serve their business objectives more accurately and reflect their priorities and the ways in which they prefer to work.

5.3 Debugging

Debugging is the process of identifying and resolving issues or errors in software applications. It is a critical step in the software development life cycle and is aimed at improving the quality and functionality of the application. During the development process, errors or bugs can occur in the code that can cause the application to behave in unexpected ways or not work at all. Debugging involves identifying and diagnosing these errors, tracing their root cause, and then making the necessary changes to fix them.

Different types of debugging methods used in this system are:

- **Unit Testing:** The application was divided into smaller components and tested individually. Each code was executed separately to ensure accuracy.

- **Integration Testing:** Each small component was integrated or combined into a module to ensure that each module works properly when put together. This was done to check connectivity between modules.
- **System Testing:** The system was tested by combining every module. This was to ensure that each process have a particular order. This was to ensure that the system does not crash while using.

5.4 System Security

Password encryption is used to protect each user's details.

5.5 Scope for future enhancement

For future enhancement, the social media platform could consider implementing advanced comment moderation with sophisticated algorithms and machine learning models, to create a safer and more positive environment. Real-time comment translation could be introduced for linguistic inclusivity. Personalization features can be introduced to allow users to tailor their moderation preferences and filters, fostering a more tailored and engaging experience.

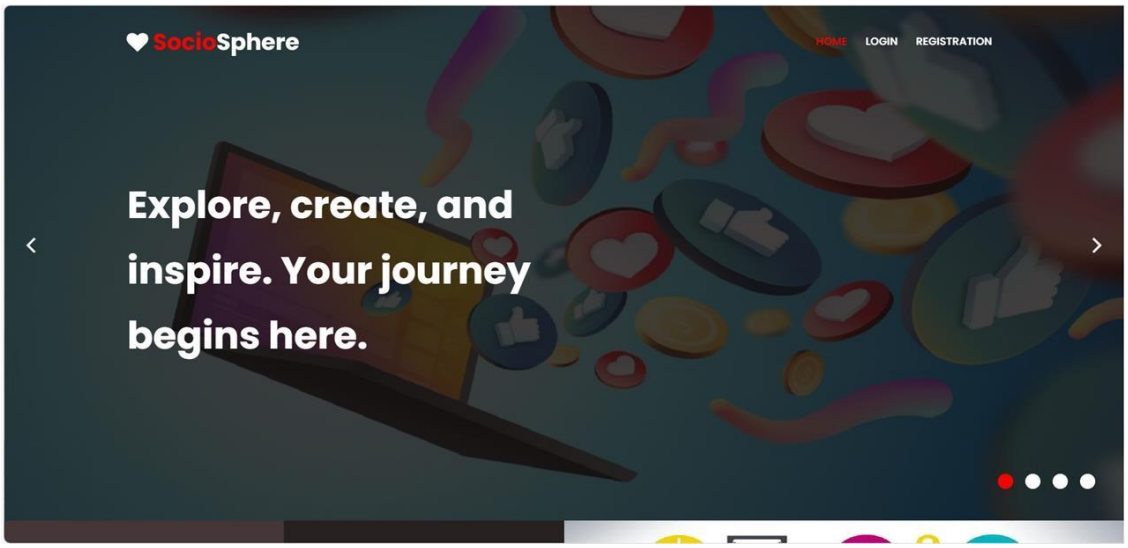
6. CONCLUSION

Conclusion

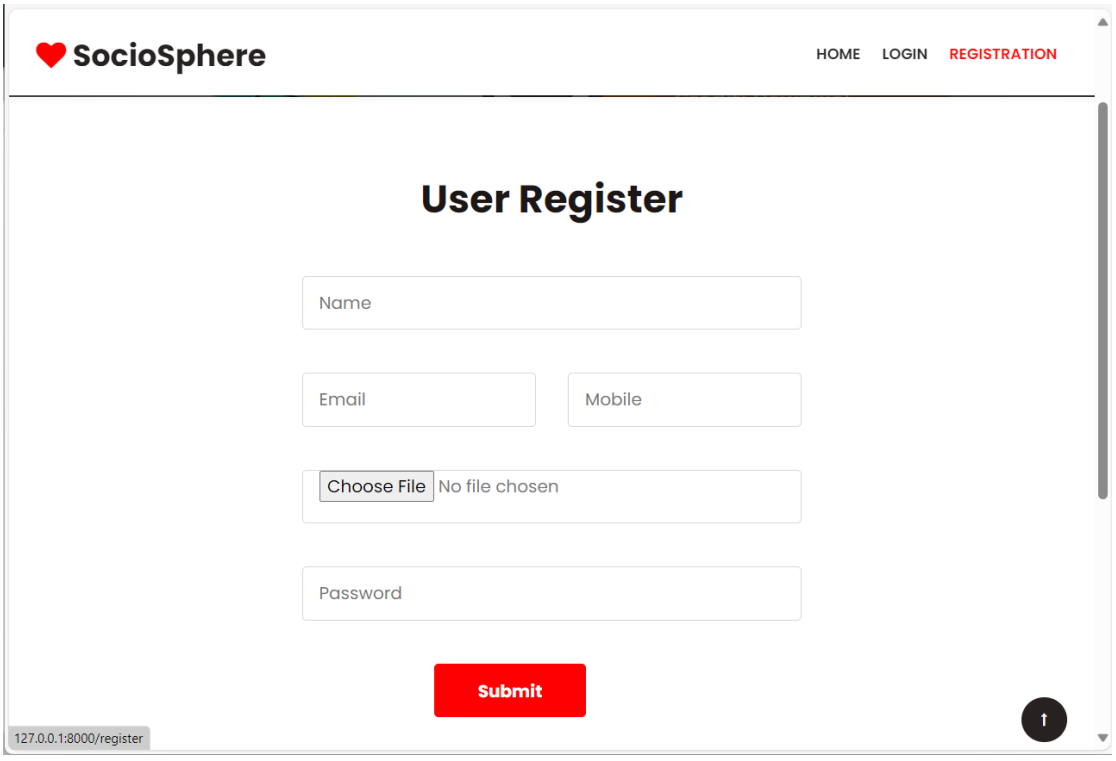
Sociosphere represents a groundbreaking advancement in social media platforms, harnessing sentiment analysis technology to cultivate a positive and inclusive online community. By automatically filtering out negative sentiments, Sociosphere promotes constructive interactions among users, fostering mutual respect and understanding. With its innovative approach to sentiment-aware moderation, Sociosphere sets a new standard for online engagement, prioritizing positivity and meaningful discourse. As a result, Sociosphere stands as a beacon of innovation in the digital landscape, paving the way for a more harmonious and supportive online environment.

7. APPENDIX

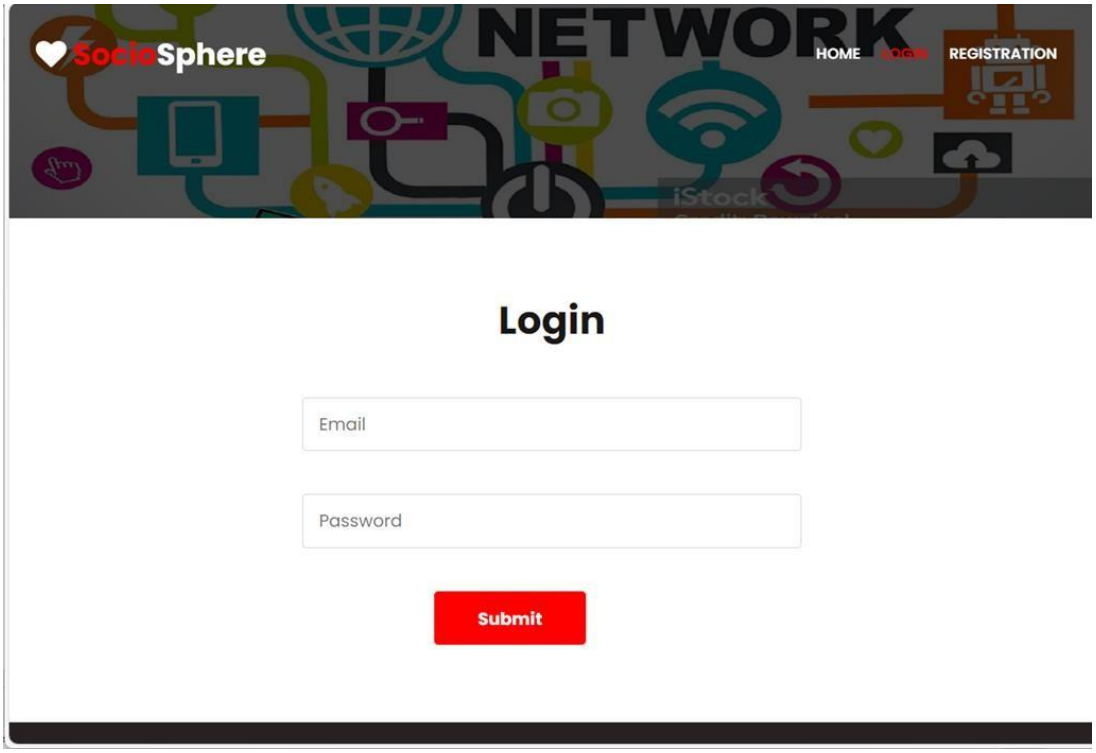
Home page



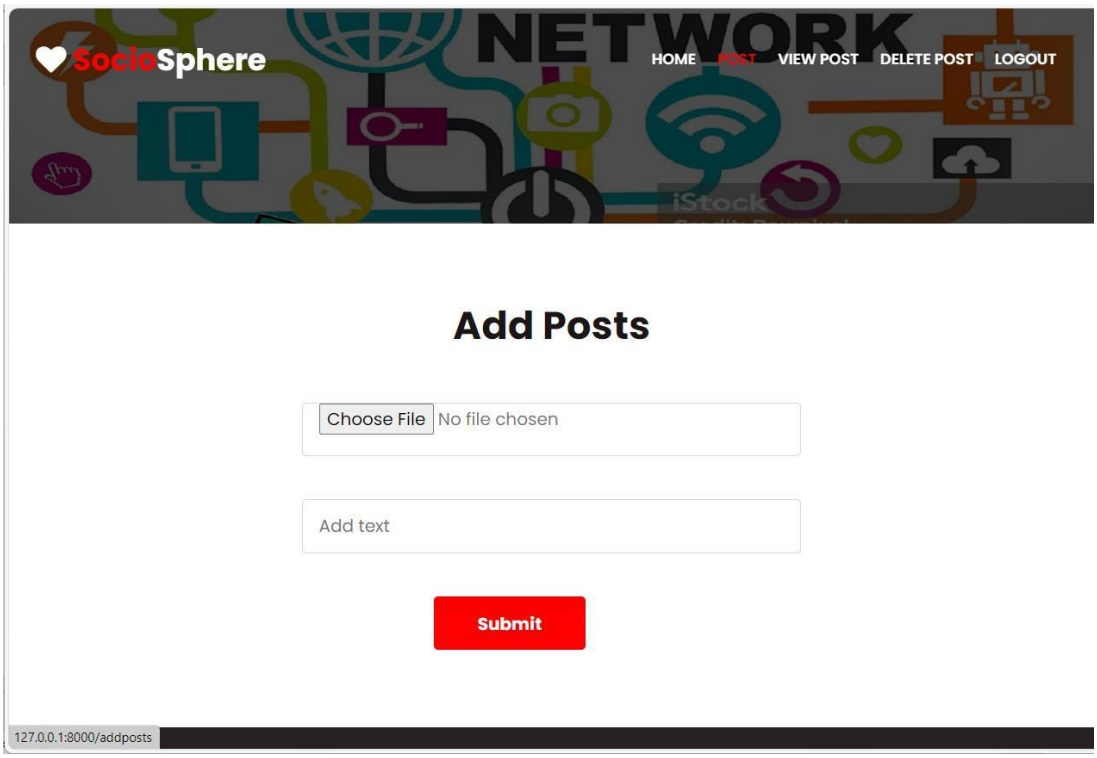
Registration page



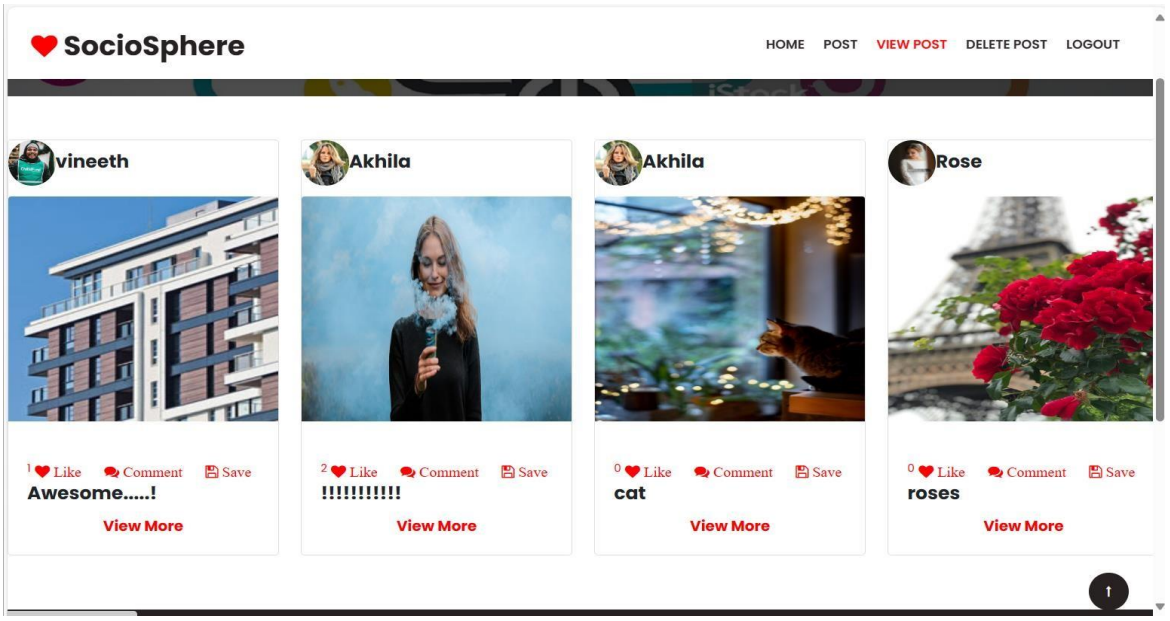
Login Page



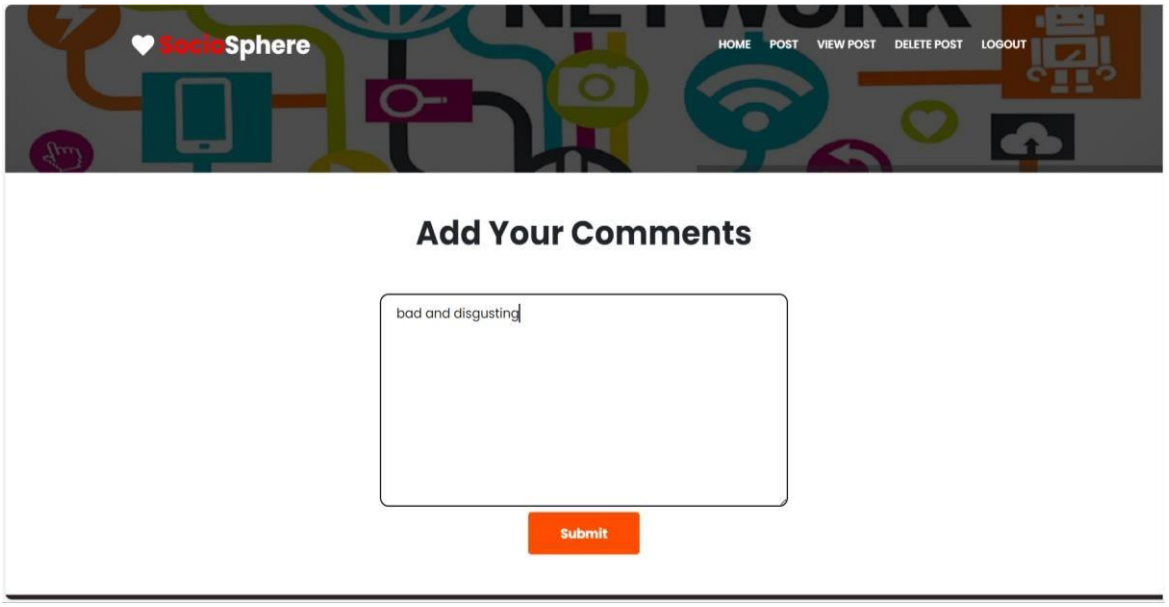
Upload Posts



View Post

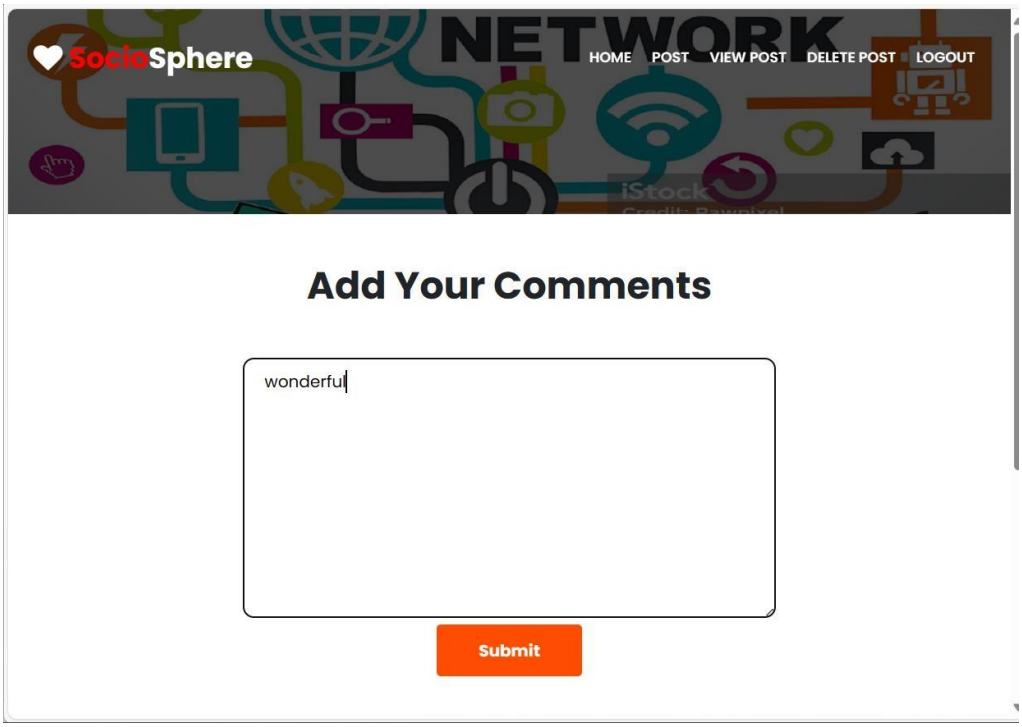


Add negative comment

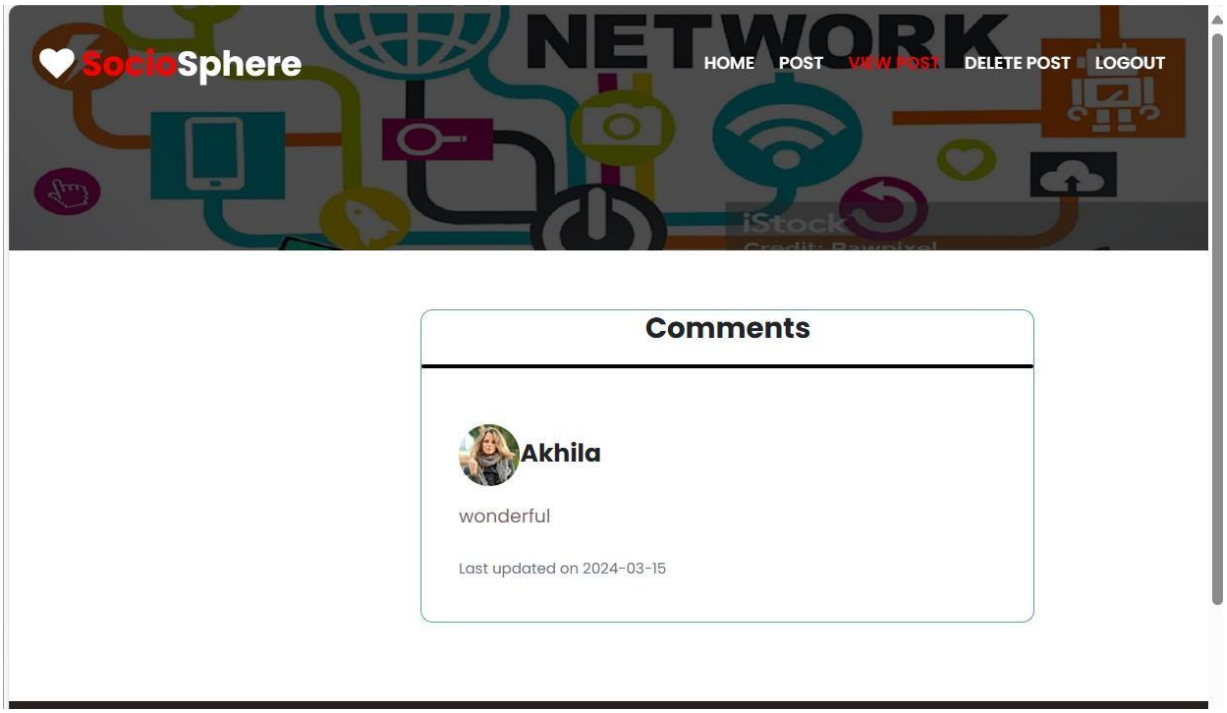


View Comment

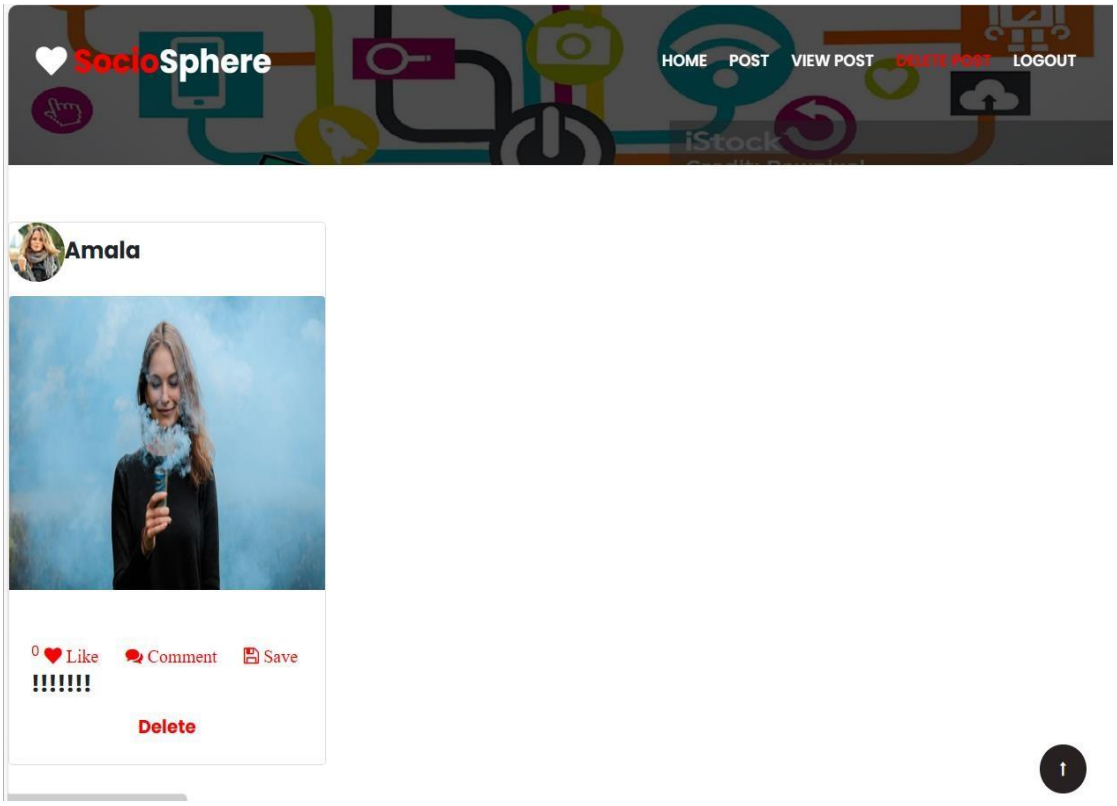
Add non negative comment



View Comment



Delete Post



8. BIBLIOGRAPHY

8.1 References

- Silva, Leandro, et al. "Analyzing the targets of hate in online social media." *Proceedings of the International AAAI Conference on Web and Social Media*. Vol. 10. No. 1. 2016.

- Castaño-Pulgarín, Sergio Andrés, et al. "Internet, social media and online hate speech. Systematic review." *Aggression and Violent Behavior* 58 (2021): 101608.