Project Report Format

1. INTRODUCTION

1.1 Project Overview

The Meme Museum is an ambitious project that aims to create a digital museum showcasing a curated collection of memes from various online platforms. This project recognizes the cultural significance and impact of internet memes as a form of communication and entertainment in modern society. By preserving and celebrating memes, the museum provides a unique opportunity to explore the evolution of internet culture over time. The museum's collection will encompass a wide range of memes, including both classic and popular ones as well as niche and obscure gems. These memes will cover a diverse array of themes and topics, reflecting the rich tapestry of internet culture. Through the collection, visitors will be able to witness the transformative power of memes and their ability to capture and convey ideas, humour, and emotions. The project strives to create a lasting legacy of internet culture for future generations to explore and enjoy.

1.2 Purpose

1. Preserve Internet Culture: The project aims to safeguard the history and cultural significance of internet memes by curating and archiving a diverse collection.
2. Celebrate Creativity and Entertainment: The museum intends to highlight the creative and entertaining aspects of memes by showcasing a wide range of memes.
3. Educate and Engage Visitors: The museum aims to provide visitors with an educational and interactive experience. By presenting the evolution of memes over time, it offers insights into their origins, influences, and societal impact.
4. Bridge Generational and Cultural Divides: The project strives to bridge generational and cultural gaps by bringing together diverse audiences. By showcasing memes from different eras and communities.

5. Promote Research and Analysis: The museum aspires to become a valuable resource for researchers, scholars, and students interested in studying internet culture and the social phenomena surrounding memes
6. Collaborate and Partner with Online Platforms: The project seeks to collaborate with online platforms, meme creators, and internet communities to ensure a comprehensive, representative collection and growth of the internet meme ecosystem.

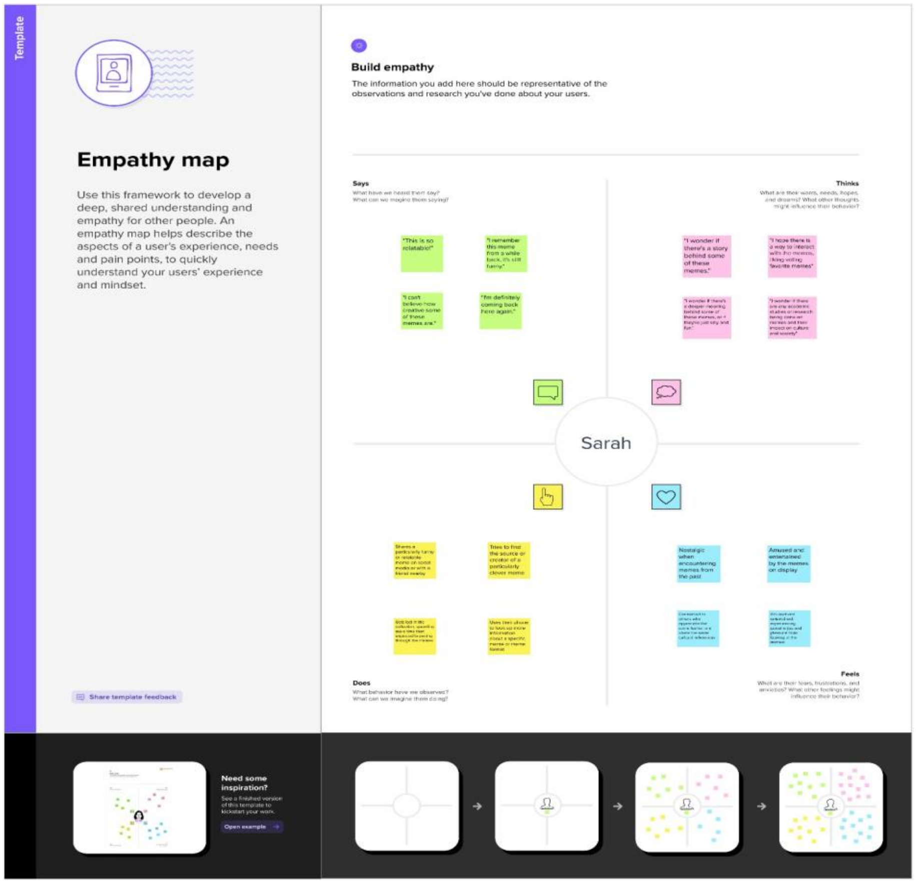
2. IDEATION & PROPOSED SOLUTION

2.1 Problem Statement Definition

Problem Statement (PS)	**I am (Customer)**	**I'm trying to**	**But**	**Because**	**Which makes me feel**
I am someone who is highly engaged with social media and internet culture ,which makes me a potentially valuable visitor to the Meme Museum	Student	approach the Meme Museum with a positive attitude and give it a fair chance to impress me	I'm also skeptical that it will be able to provide a truly unique and engaging experience that justifies the price of admission.	I'm considering visiting the Meme Museum because I'm always on the lookout for new and entertaining memes to share with my friends and followers on social media	excited to see what the Meme Museum has to offer and whether it can live up to my expectations

As a content creator, I am always on the lookout for new and interesting memes and trends to incorporate into my work, which could make the Meme Museum a valuable resource for me.	Content Creator	stay up-to-date with the latest memes and trends with my skepticism that the Meme Museum will be able to provide a truly unique and valuable experience	explore the Meme Museum's collection, but I'm also worried that it will not have anything truly new or innovative that I haven't already seen online.	I'm always on the lookout for new sources of inspiration and ideas for my content.	curious and excited to see what the Meme Museum has to offer and whether it can inspire me with new ideas for my content.

2.2 Empathy Map Canvas



Empathy map

Use this framework to develop a deep, shared understanding and empathy for other people. An empathy map helps describe the aspects of a user's experience, needs and pain points, to quickly understand your users' experience and mindset.

See a finished version of this template to kickstart your work.

Build empathy

The information you add here should be representative of the observations and research you've done about your users.

Says
What have we heard them say?
What can we imagine them saying?

"This is so relatable!"
"I remember this meme from a while back, it's still funny."
"I can't believe how creative some of these memes are."
"I'm definitely coming back here again."

Thinks
What are their wants, needs, hopes, and dreams? What other thoughts might influence their behavior?

"I wonder if there's a story behind some of these memes."
"I hope there is a way to interact with the memes, liking voting 'favorite memes'"
"I wonder if there's a deeper meaning behind some of these memes, or if they're just silly and fun."
"I wonder if there are any academic studies or research being done on memes and their impact on culture and society"

Sarah

Shares a particularly funny or relatable meme on social media or with a friend nearby
Tries to find the source or creator of a particularly clever meme
[illegible]
Uses their phone to look up more information about a specific meme or meme format

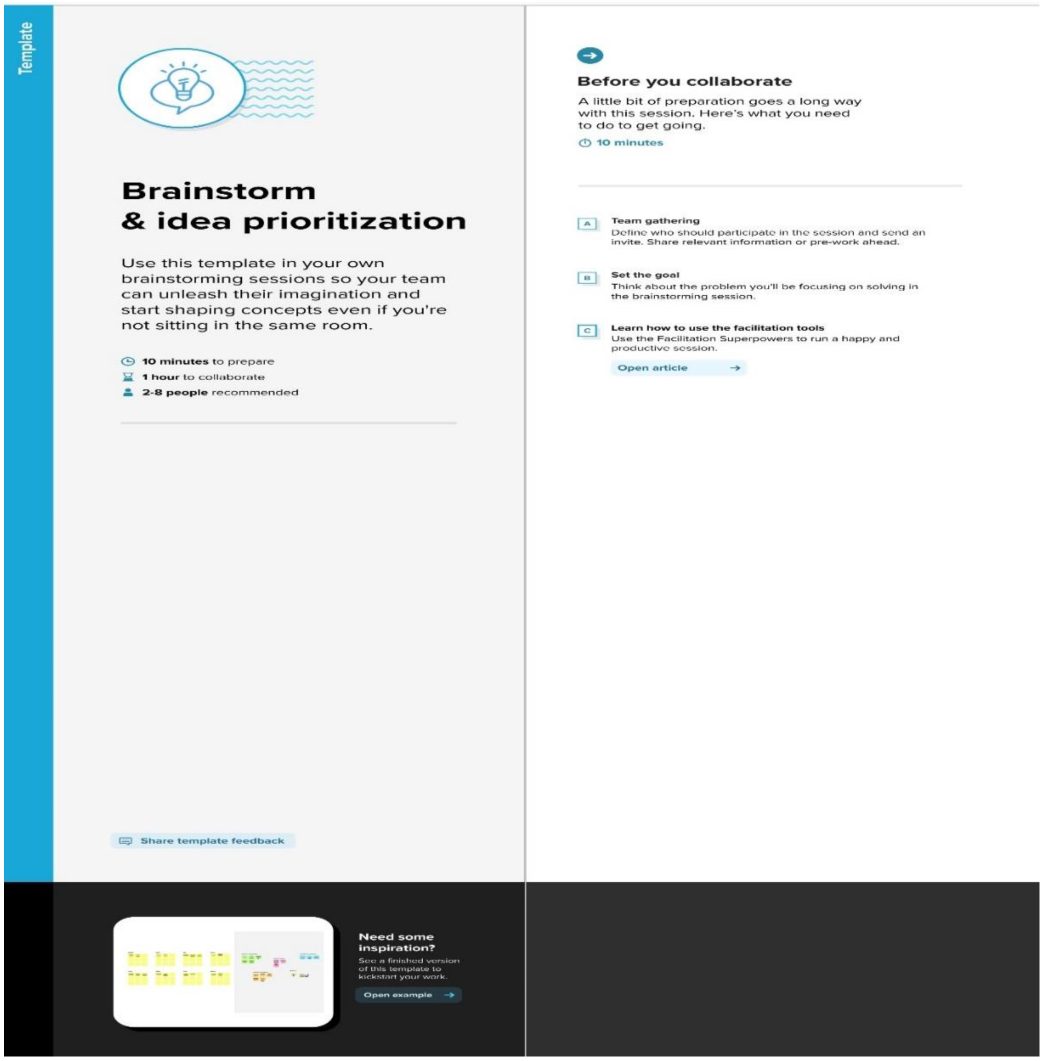
Nostalgic when encountering memes from the past
Amused and entertained by the memes on display
[illegible]
[illegible]

Does
What behavior have we observed?
What can we imagine them doing?

Feels
What are their fears, frustrations, and anxieties? What other feelings might influence their behavior?

2.3 Ideation & Brainstorming

Step-1: Team Gathering, Collaboration and Select the Problem Statement



Template

Brainstorm & idea prioritization

Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.

10 minutes to prepare
1 hour to collaborate
2-8 people recommended

Share template feedback

Before you collaborate

A little bit of preparation goes a long way with this session. Here's what you need to do to get going.

10 minutes

A **Team gathering**
Define who should participate in the session and send an invite. Share relevant information or pre-work ahead.

B **Set the goal**
Think about the problem you'll be focusing on solving in the brainstorming session.

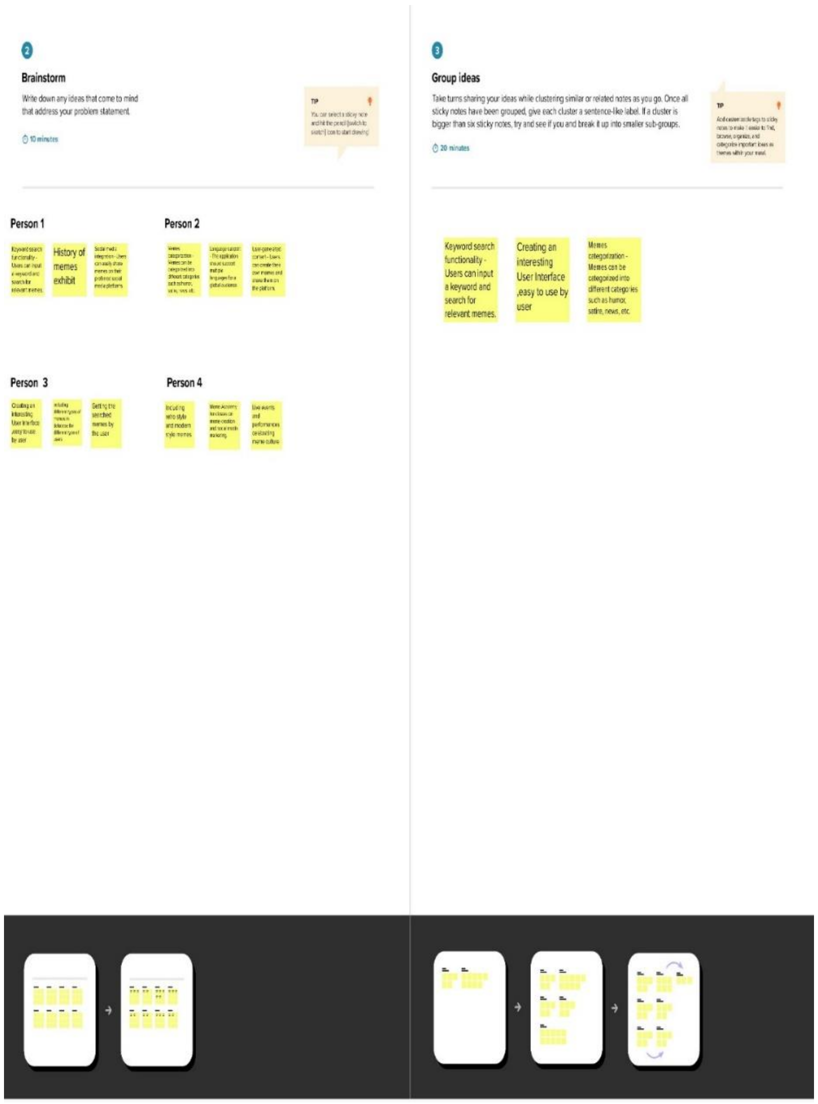
C **Learn how to use the facilitation tools**
Use the Facilitation Superpowers to run a happy and productive session.

Open article →

Need some inspiration?
See a finished version of this template to kickstart your work.

Open example →

Step-2: Brainstorm, Idea Listing and Grouping



2

Brainstorm

Write down any ideas that come to mind that address your problem statement.

10 minutes

TIP

You can select a sticky note and hit the pencil [switch to sketch] icon to start drawing!

Person 1

Keyword search functionality - Users can input a keyword and search for relevant memes.

History of memes exhibit

Social media integration - Users can easily share memes on their preferred social media platforms

Person 2

Memes categorization - Memes can be categorized into different categories such as humor, satire, news, etc.

Language support - The application should support multiple languages for a global audience

User-generated content - Users can create their own memes and share them on the platform.

Person 3

Creating an interesting User Interface ,easy to use by user

including different types of memes in [illegible] for different types of users

Getting the searched memes by the user

Person 4

Including retro style and modern style memes

Meme Academy [illegible] classes on meme creation and social media marketing.

Live events and performances celebrating meme culture

3

Group ideas

Take turns sharing your ideas while clustering similar or related notes as you go. Once all sticky notes have been grouped, give each cluster a sentence-like label. If a cluster is bigger than six sticky notes, try and see if you and break it up into smaller sub-groups.

20 minutes

TIP

Add customizable tags to sticky notes to make it easier to find, browse, organize, and categorize important ideas as themes within your mural.

Keyword search functionality - Users can input a keyword and search for relevant memes.

Creating an interesting User Interface ,easy to use by user

Memes categorization - Memes can be categorized into different categories such as humor, satire, news, etc.

Step-3: Idea Prioritization

4

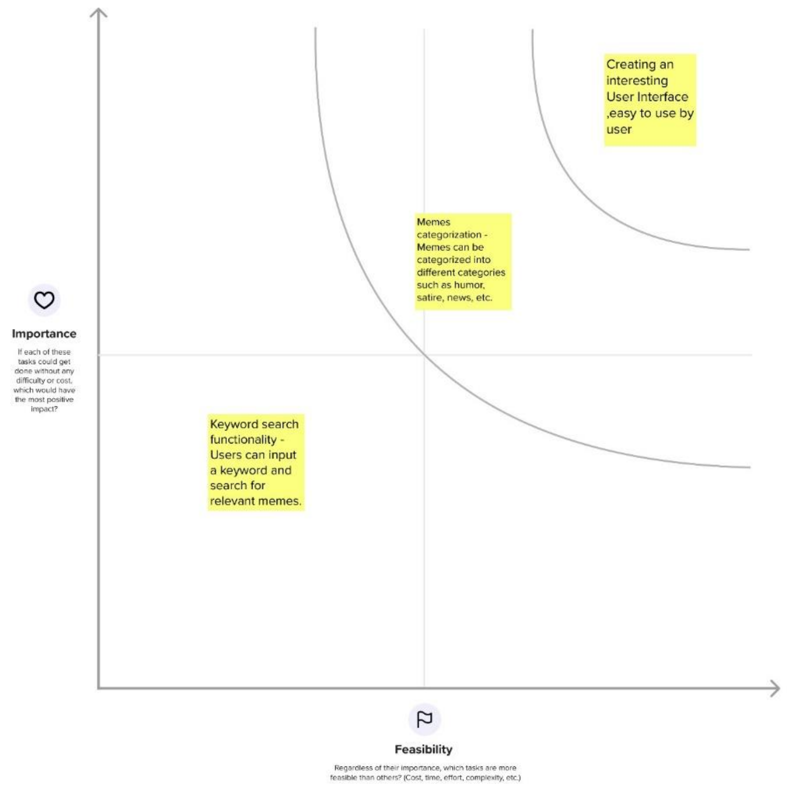
Prioritize

Your team should all be on the same page about what's important moving forward. Place your ideas on this grid to determine which ideas are important and which are feasible.

20 minutes

TIP

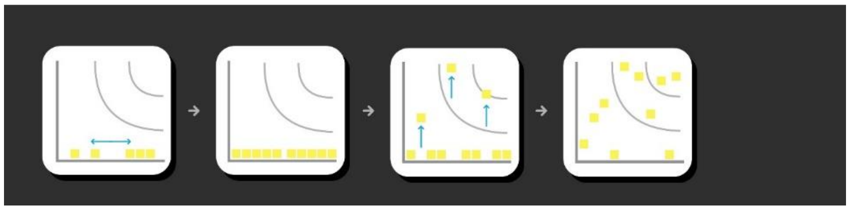
Participants can use their cursors to point at where sticky notes should go on the grid. The facilitator can confirm the spot by using the laser pointer holding the **H key** on the keyboard.


Importance
If each of these tasks could get done without any difficulty or cost, which would have the most positive impact?
Feasibility
Regardless of their importance, which tasks are more feasible than others? (Cost, time, effort, complexity, etc.)

Creating an interesting User Interface ,easy to use by user

Memes categorization - Memes can be categorized into different categories such as humor, satire, news, etc.

Keyword search functionality - Users can input a keyword and search for relevant memes.



2.4 Proposed Solution

1. Web Application: Develop a user-friendly web application as the central hub of the Meme Museum. The application will allow visitors to access and explore the curated collection of memes.
2. API Integration: Collaborate with various online platforms and utilize APIs to fetch memes from different sources. This integration will enable the museum to gather a diverse and extensive collection of memes, ensuring representation of classic, popular, niche, and obscure memes.
3. User Instructions for Meme Creation: Implement a meme creation feature where registered users can provide instructions to generate custom memes. The application will utilize AI algorithms and meme generation libraries to process these instructions and generate memes accordingly. This feature will add an interactive and personalized touch to the museum experience.
4. User Authentication: Implement a secure user authentication system to protect user data and provide personalized experiences. Users can create accounts, log in, and access additional features such as saving favourite memes, submitting their own memes for consideration, and participating in community activities.
5. Curated Collections and Themes: Categorize and curate the meme collection into various themes and topics. Users can explore different sections of the museum based on their interests, allowing them to delve into specific categories or timelines that showcase the evolution of memes over time.
6. Engaging User Experience: Design an intuitive and visually appealing user interface that offers seamless navigation and search functionalities. Implement features like meme voting, comments, and social sharing to encourage user engagement and interaction within the museum community.
7. Educational Content: Accompany the meme collection with informative and educational content. Provide descriptions, historical context, and explanations of meme references to enhance visitors' understanding and appreciation of internet culture.
8. Community and Social Impact: Foster a sense of community by allowing users to interact with each other through comments, discussions, and meme sharing. Encourage meme creators to contribute their work, provide credit to original creators, and organize meme contests to celebrate the talent within the community.

3. **REQUIREMENT ANALYSIS**

3.1 Functional requirement

FR No.	**Functional Requirement (Epic)**	**Sub Requirement (Story / Sub-Task)**
FR-1	User Registration	Users can create accounts to access additional features and personalize their experience. Registration may require providing basic information such as username, email, and password. The registration process should include validation to ensure data integrity and security. Users should receive confirmation or verification emails to complete the registration process.
FR-2	Search functionality	Users can search for specific memes or topics using keywords or tags. The search feature should provide relevant and accurate results based on the user's input. Advanced search options such as filtering by date, popularity, or category can be implemented. The search functionality should be fast and responsive, allowing users to find memes easily..
FR-3	Curated Content	The Meme Museum will feature a curated collection of memes from various online platforms. Curated content should represent a diverse range of themes, topics, and time periods. Memes should be selected based on their cultural significance, historical relevance, and entertainment value .The curated content should be regularly updated to keep the museum fresh and engaging for visitors.
FR-4	Interactive experience	The web application should provide an interactive and engaging experience for users. Users can navigate through exhibits, exploring memes from different eras or themes.

		Memes can be displayed with additional information, such as descriptions, historical context, or explanations. Users can interact with memes by voting, rating, commenting, or sharing them on social media platforms. The application can offer features like meme creation or customization to encourage user participation and creativity. These functionalities contribute to cre

3.2 Non-Functional requirements

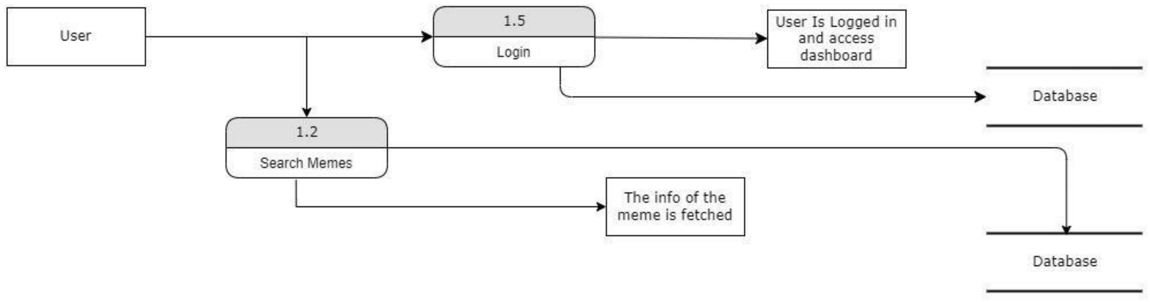
FR No.	**Non-Functional Requirement**	**Description**
NFR-1	**Usability**	The web application should have an intuitive and user-friendly interface. Navigation should be easy, allowing users to browse and explore memes without confusion. The application should provide clear instructions and guidance on how to interact with exhibits and features. Accessibility features should be implemented to ensure inclusivity for users with disabilities.
NFR-2	**Security**	User data should be securely stored and protected. Implement appropriate authentication and authorization mechanisms to ensure that only authorized users can access and modify sensitive information. Apply encryption techniques to protect sensitive user data, such as passwords.

		Regular security audits and updates should be conducted to identify and address potential vulnerabilities.
NFR-3	**Reliability**	The web application should be highly reliable, minimizing downtime and system failures. Error handling and exception management should be implemented to handle unforeseen issues gracefully. Regular backups and data recovery mechanisms should be in place to ensure the preservation of memes and user data.
NFR-4	**Performance**	The application should be optimized for efficient performance, providing fast loading times and responsive interactions. Minimize latency in fetching and displaying memes to ensure a smooth user experience. Conduct performance testing to identify and address any bottlenecks or performance issues.
NFR-5	**Availability**	The Meme Museum should be available to users at all times, with minimal maintenance-related downtime. Implement load balancing and failover mechanisms to handle high traffic and ensure continuous availability. Monitor server and network infrastructure to promptly address any issues that may impact availability.
NFR-6	**Scalability**	The application should be designed to handle increasing user demand and a growing meme collection. Implement a scalable architecture that allows for easy addition of new servers or resources as needed.

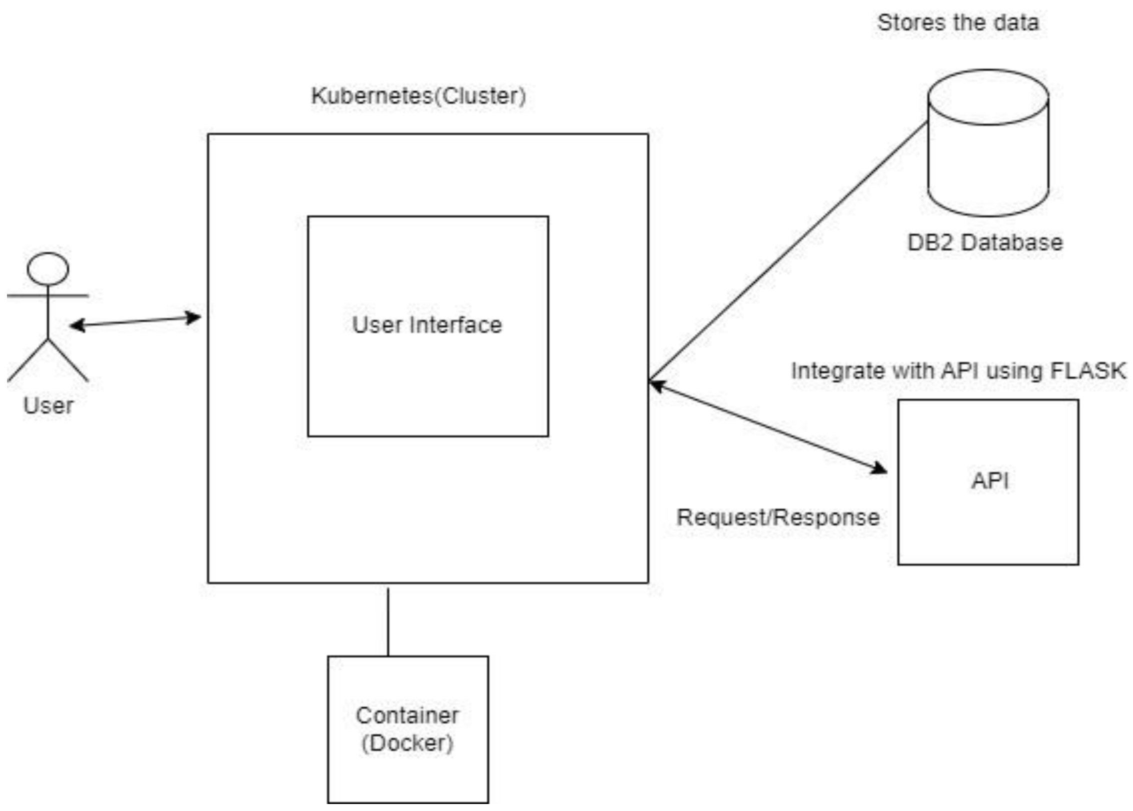
		Conduct load testing to ensure the application can handle concurrent user access and maintain performance.

4. PROJECT DESIGN

4.1 Data Flow Diagrams


User
1.5
Login
User Is Logged in and access dashboard
Database
1.2
Search Memes
The info of the meme is fetched
Database

4.2 Solution & Technical Architecture



4.3 User Stories

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Team Member
Customer (Mobile user)	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	I can access my account / dashboard	High	Krishna Prasad S
		USN-2	As a user so that I can access additional features and personalize my experience on the app.	I can receive confirmation & click confirm	High	Krishna Prasad S
		USN-3	As a user, I want the registration process to be quick and easy.	I can register & access the dashboard quickly without any buffering.	Low	Sai Ddiwakar
		USN-4	As a user, I can register for the application through Gmail		Medium	Kamesh
	Login	USN-5	As a user, I can log into the application by entering email & password	If incorrect showing the prompt.	High	Krishna Prasad S
Customer (Web user)	Overall application usage	USN-7	The website should be responsive design, adapting to different screen.	I can view at any type of device as the application should be compatible.	High	Krishna Prasad S
		USN-8	The website should be easy to navigate, with clear menus and links to relevant content.	I can view the relevant content when navigating to the page.	High	Baby Shree J
		USN-9	The user should be able to interact with website.	I can easily interact with the website by searching.	Medium	Sai Ddiwakar

Customer Care Executive		USN-10	The system should be able to provide real time data or impactful memes.	The memes I see should be realistic and learnable.	Medium	Kamesh E
		USN-11	The system should have a search functionality to help executives find customer information quickly and efficiently.	The searching should be fast and data fetching should be quick.	High	Krishna Prasad S
User Type	**Functional Requirement (Epic)**	**User Story Number**	**User Story / Task**	**Acceptance criteria**	**Priority**	**Team Member**
		USN-12	The system should provide a user-friendly interface for customer care executives to manage customer queries and requests.	The UI should be creative and appealing.	High	Sai Ddiwakar
Administrator	Login	USN-13	The system should provide a secure login mechanism for administrators to access the backend.	The login and logout should be secure and passwords should be encrypted.	High	Kamesh E
	Managing traffic	USN-14	The site should have minimal downtime.	The site should always be up.	High	Kamesh E

5. CODING & SOLUTIONING (Explain the features added in the project along with code)

5.1 Feature 1 : Login and SignUp

```
@auth.route("/sign-up", methods=["POST", "GET"])
def sign_up():
    if request.method == "POST":
        email = request.form.get("email")
        name = request.form.get("name")
        pwd = request.form.get("pass")
        pass_conf = request.form.get("pass_conf")
        user = User.query.filter_by(email=email).first()

        if user:
            flash("Email already exists.", category="error")
        elif len(email) < 4:
            flash("Email must be greater than 3 characters.",
category="error")
        elif len(name) < 2:
            flash("First name must be greater than 1 character.",
category="error")
        elif pwd != pass_conf:
            flash("Passwords don't match.", category="error")
        elif len(pwd) < 7:
            flash("Password must be at least 7 characters.",
category="error")
        else:
            new_user = User(
                email=email,
                name=name,
                password=generate_password_hash(pwd, method="sha256"),
            )
            db.session.add(new_user)
            db.session.commit()
            login_user(new_user, remember=True)
            flash("Account created!", category="success")
            return redirect(url_for("views.home"))

    return render_template("sign-up.html", user=current_user)
```

```
@auth.route("/login", methods=["POST", "GET"])
def login():
    if request.method == "POST":
        user =
db.session.query(User).filter_by(email=request.form.get("email")).first()

        if user:
            if check_password_hash(user.password,
request.form.get("pass")):
                login_user(user, remember=True)
                flash(
                    f"Logged in as {current_user.name.capitalize()}!",
                    category="success",
                )
                return redirect(url_for("views.home"))
            else:
                flash("Incorrect Password!", category="error")
        else:
            flash("Incorrect Email!", category="error")

    return render_template("login.html", user=current_user)
```

5.2 Feature 2

Generating the Meme.

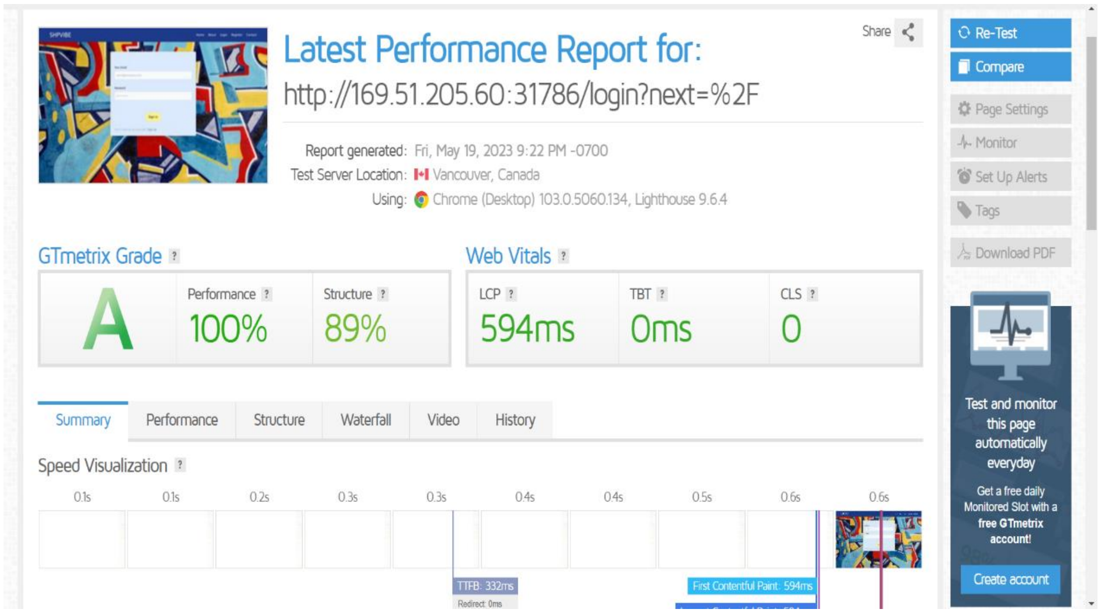
```
 meme = request.form.get("meme")
    print(meme)
    #  querystring = {
    #      "exclude-tags": "nsfw",
    #      "keywords": "rocket",
    #      "min-rating": "7",
    #      "include-tags": "one_liner",
    #      "number": "3",
    #      "max-length": "200",
    #  }
    querystring = {
        "keywords": meme,
        "media-type": "image",
        "keywords-in-image": "false",
        "min-rating": "3",
        "number": "3",
    }
    headers = get_api_data()
    response = requests.get(os.environ["API_URL"], headers=headers,
params=querystring)
    print("\n\nResponse: ", response.json(), "\n\n\n")
    json_response = response.json()
    # return json.dumps(json_response)
    # json_response = {
    #     "memes": [
    #         {
    #             "id": 6698,
    #             "description": "Bezos, Phallic Rockets, Taxes. Twitter
didn’t disappoint.:
https://twitter.com/TheDailyShow/status/1417478230068563973?s=20",
    #             "url": "https://i.imgur.com/8JTc5z3.jpg",
    #             "type": "image/jpeg",
    #         },
    #         {
    #             "id": 237030,
    #             "description": "Rocket Money is a scam.: Prevented a $30
charge of something I don't use anymore. Feels good.",
    #             "url": "https://i.imgur.com/q3cZlpv.png",
    #             "type": "image/png",
    #         },
    #         {
```

```
#             "id": 237032,
#             "description": "Rocket Money is a scam.: Cat tax.",
#             "url": "https://i.imgur.com/yaZoCFP.jpg",
#             "type": "image/jpeg",
#         },
#     ],
#     "available": 20,
# }

return render_template(
    "generatedmeme.html",
    user=current_user,
    json_response=json_response,
)
```

6. RESULTS

6.1 Performance Metrics


Latest Performance Report for:

http://169.51.205.60:31786/login?next=%2F

Report generated: Fri, May 19, 2023 9:22 PM -0700
Test Server Location: Vancouver, Canada
Using: Chrome (Desktop) 103.0.5060.134, Lighthouse 9.6.4

GTmetrix Grade: A — Performance 100% — Structure 89%

Web Vitals: LCP 594ms — TBT 0ms — CLS 0

Speed Visualization: TTFB: 332ms; Redirect: 0ms; First Contentful Paint: 594ms

7. ADVANTAGES:

- Cultural preservation: The Meme Museum provides a platform for the preservation and curation of popular internet memes, capturing a unique aspect of contemporary culture.
- Entertainment value: Visitors can enjoy a wide range of hilarious and relatable
- memes, creating an enjoyable and light-hearted experience.
- Social sharing: The museum encourages social interaction as visitors can discuss and share their favorite memes with others, fostering a sense of community.
- Education and reflection: Memes often reflect social and political issues, providing an opportunity for visitors to engage in discussions and gain insights into current events in a lighthearted manner.

DISADVANTAGES:

- Transient nature: Memes have a short lifespan and can quickly lose relevance or humor over time. The museum may struggle to keep up with the constantly evolving meme landscape.
- Subjectivity and taste: Humor is subjective, and not all memes may resonate with every visitor. Some memes may be offensive or controversial, leading to potential disagreements or discomfort among visitors.
- Copyright issues: Many memes are based on copyrighted material, which could pose legal challenges for the museum. Obtaining permissions and licenses to display certain memes could be complex and costly.
- Limited depth: Memes are often short and simple, lacking the complexity and depth found in traditional art forms. The museum may struggle to provide a profound and intellectually stimulating experience compared to other art museums.

8. CONCLUSION:

- On the positive side, it serves as a platform for cultural preservation, provides entertainment value, encourages social sharing, and offers educational opportunities.
- However, there are challenges to consider, including the transient nature of memes, subjective taste, copyright issues, and potential limitations in depth and intellectual engagement.
- Ultimately, the Meme Museum offers a unique and lighthearted experience, but it may not appeal to everyone and may require careful curation to stay relevant in a rapidly changing meme landscape.

9. <u>FUTURE SCOPE:</u>

- **Embracing emerging trends:**
  - The Meme Museum can stay relevant by continuously adapting to new meme trends and formats. It should actively monitor online platforms and engage with meme communities to identify and curate the latest hilarious content.

- **Interactive exhibits:**
  - To enhance visitor engagement, the museum could introduce interactive exhibits that allow visitors to create their own memes or participate in meme-related activities. This would provide a more immersive and hands-on experience.

- **Virtual and augmented reality experiences:**
  - The museum could explore the use of virtual and augmented reality technologies to offer unique and immersive experiences. Visitors could virtually interact with memes or step into a virtual meme world, further blurring the lines between online and offline humor.

- **Collaboration with meme creators:**
  - : Partnering with popular meme creators and influencers can add a dynamic element to the museum. They could host workshops, talks, or even temporary exhibits showcasing their work, fostering a sense of authenticity and bringing in a wider audience.

- **Global outreach:**

    - The Meme Museum could expand its reach beyond physical locations by creating a strong online presence. It could establish a comprehensive digital archive, develop a website or app for virtual visits, and leverage social media to share and promote its curated collection of hilarity worldwide.

- **Research and analysis:**
    - The museum could collaborate with researchers and sociologists to study the impact of memes on culture, society, and communication. This would provide valuable insights into the evolution and significance of internet humor.

- **Collaborations with other museums:**
    - The Meme Museum could collaborate with other art or history museums to create exhibitions that explore the intersection of memes with traditional art forms or historical events. This interdisciplinary approach could attract diverse audiences and broaden the museum's appeal.

10. APPENDIX:

Source Code

__init__.py

```python
from os import path, environ
from flask import Flask
from flask_sqlalchemy import SQLAlchemy
from flask_login import LoginManager


db = SQLAlchemy()
DB_NAME = "model.db"
```

```python
def create_app():
    app = Flask(__name__)
    app.config["SECRET_KEY"] = environ["SECRET_KEY"]
    app.config["SQLALCHEMY_TRACK_MODIFICATIONS"] = True
    # app.config["SQLALCHEMY_ECHO"] = True
    # app.config["SQLALCHEMY_DATABASE_URI"] = f"""sqlite:///{DB_NAME}"""
    app.config[
        "SQLALCHEMY_DATABASE_URI"
    ] =
f"""db2://{environ['USER']}:{environ['PASSWORD']}@{environ['HOST']}:{environ['PORT']}/{environ['DATABASE']};security=ssl;sslcertificate={environ['SSL_CERTIFICATE']}"""
    db.init_app(app)

    from .html_test import html_test
    from .auth import auth
    from .views import views

    app.register_blueprint(html_test, url_prefix="/test")
    app.register_blueprint(auth, url_prefix="/")
    app.register_blueprint(views, url_prefix="/")

    from .models import User

    with app.app_context():
        db.create_all()
        print("Created Database!")

    # create_database(app)

    login_manager = LoginManager(app=app)
    login_manager.login_view = "auth.login"
    @login_manager.user_loader
    def load_user(id):
        return db.session.query(User).get(int(id))

    return app


def create_database(app: Flask):
    with app.app_context():
        if not path.exists("website/" + DB_NAME):
            db.create_all()
            print("Created Database!")
```

auth.py

```
from flask import Blueprint, render_template, request, flash, url_for,
redirect
from flask_login import login_user, logout_user, login_required,
current_user
from werkzeug.security import check_password_hash, generate_password_hash
from .models import User
from . import db

auth = Blueprint("auth", __name__)

@auth.route("/about")
def about():
   return render_template("About.html",user=current_user)

@auth.route("/Home")
def home():
    return render_template("home.html",user=current_user)

@auth.route("/contact")
def contact():
    return render_template("contact.html",user=current_user)



@auth.route("/sign-up", methods=["POST", "GET"])
def sign_up():
    if request.method == "POST":
        email = request.form.get("email")
        name = request.form.get("name")
        pwd = request.form.get("pass")
        pass_conf = request.form.get("pass_conf")

        user = User.query.filter_by(email=email).first()

        if user:
            flash("Email already exists.", category="error")
        elif len(email) < 4:
            flash("Email must be greater than 3 characters.",
category="error")
        elif len(name) < 2:
            flash("First name must be greater than 1 character.",
category="error")
        elif pwd != pass_conf:
            flash("Passwords don't match.", category="error")
        elif len(pwd) < 7:
            flash("Password must be at least 7 characters.",
category="error")
        else:
            new_user = User(
                email=email,
                name=name,
                password=generate_password_hash(pwd, method="sha256"),
            )
            db.session.add(new_user)
            db.session.commit()
            login_user(new_user, remember=True)
            flash("Account created!", category="success")
            return redirect(url_for("views.home"))

    return render_template("sign-up.html", user=current_user)


@auth.route("/login", methods=["POST", "GET"])
def login():
    if request.method == "POST":
        user =
db.session.query(User).filter_by(email=request.form.get("email")).first()

        if user:
            if check_password_hash(user.password,
request.form.get("pass")):
                login_user(user, remember=True)
                flash(
                    f"Logged in as {current_user.name.capitalize()}!",
                    category="success",
                )
                return redirect(url_for("views.home"))
            else:
                flash("Incorrect Password!", category="error")
        else:
```

```python
            flash("Incorrect Email!", category="error")

    return render_template("login.html", user=current_user)




@auth.route("/logout")
@login_required
def logout():
    flash(
        f"{current_user.name.capitalize()} Logged out successfully!",
category="success"
    )
    logout_user()
    return redirect(url_for("auth.login"))
```

views.py

```python
from flask import Blueprint, redirect, render_template, request, url_for
from flask_login import login_required, current_user
import requests
import os, json
from .api import get_api_data


views = Blueprint("views", __name__)


@views.route("/", methods=["GET", "POST"])
@login_required
```

```
def home():
    return render_template("home.html", user=current_user)


@views.route("/memeinput", methods=["GET", "POST"])
@login_required
def memeinput():
    if request.method == "POST":
        json_response = get_meme()
        # meme=request.form.get("meme")
        # querystring = {"exclude-tags":"nsfw","keywords":meme,"min-
rating":"7","include-tags":"one_liner","number":"3","max-length":"200"}
        # headers=get_api_data()
        # response = requests.get(os.environ['API_URL'], headers=headers,
params=querystring)
        # print(type(response.json()))
        # return response.json()

        return redirect(url_for("views.get_meme"))
    return render_template("MemeInput.html", user=current_user)


@views.route("/showmeme")
@login_required
def generatememe():
    return render_template("generatedmeme.html", user=current_user)


@views.route("/meme")
def get_meme():
    meme = request.form.get("meme")
    print(meme)
    #  querystring = {
```

```
    #       "exclude-tags": "nsfw",
    #       "keywords": "rocket",
    #       "min-rating": "7",
    #       "include-tags": "one_liner",
    #       "number": "3",
    #       "max-length": "200",
    #   }
    querystring = {
        "keywords": meme,
        "media-type": "image",
        "keywords-in-image": "false",
        "min-rating": "3",
        "number": "3",
    }
    headers = get_api_data()
    response = requests.get(os.environ["API_URL"], headers=headers,
params=querystring)
    print("\n\nResponse: ", response.json(), "\n\n\n")
    json_response = response.json()
    # return json.dumps(json_response)
    # json_response = {
    #      "memes": [
    #          {
    #              "id": 6698,
    #              "description": "Bezos, Phallic Rockets, Taxes. Twitter
didn’t disappoint.:
https://twitter.com/TheDailyShow/status/1417478230068563973?s=20",
    #              "url": "https://i.imgur.com/8JTc5z3.jpg",
    #              "type": "image/jpeg",
    #          },
    #          {
    #              "id": 237030,
    #             "description": "Rocket Money is a scam.: Prevented a $30
charge of something I don't use anymore. Feels good.",
    #             "url": "https://i.imgur.com/q3cZlpv.png",
    #             "type": "image/png",
    #         },
    #         {
    #             "id": 237032,
    #             "description": "Rocket Money is a scam.: Cat tax.",
    #             "url": "https://i.imgur.com/yaZoCFP.jpg",
    #             "type": "image/jpeg",
    #         },
    #     ],
    #     "available": 20,
    # }

    return render_template(
        "generatedmeme.html",
        user=current_user,
        json_response=json_response,
    )
```

Base.html

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta http-equiv="Content-Security-Policy" content="img-src i.imgur.com
images.unsplash.com;">
    <meta name="viewport" content="width=device-width, height=device-
height, initial-scale=1.0" />
    <script src="https://cdn.tailwindcss.com"></script>
    <title>
      {% block title %}
        base
      {% endblock %}
    </title>

  </head>

  <body class="bg-[url('https://images.unsplash.com/photo-1596517335913-
66b7be20c921?ixlib=rb-
4.0.3&ixid=MnwxMjA3fDB8MHxwaG90by1wYWdlfHx8fGVufDB8fHx8&auto=format&fit=cro
p&w=2070&q=80')] bg-cover">

       <div >
        <div >
          {% if user.is_authenticated %}

              <header class="bg-blue-900 font-sans  flex items-center
justify-between  max-w-full">
                <p class=" text-3xl text-white px-3 py-4 lg:px-16
">SHPVIBE</p>

                <nav>
                  <ul
                    class=" text-center text-white transition-all py-4 flex
justify-center items-center gap-6 px-6 ">
                    <a href="/">
                      <li class="py-1 hover:text-orange-400">Home</li>
                    </a>
                    <a href="/memeinput">
                      <li class="py-1 hover:text-orange-400">Meme</li>
                    </a>
                    <a href="/logout" id="logout">
                      <li class="py-1 hover:text-orange-400">Logout</li>
                    </a>
                  </ul>

              </header>

          {% else %}

              <header class="bg-blue-900 font-sans relative lg:flex items-
center justify-between fixed">
              <p class=" text-3xl text-white px-6 py-4 lg:px-16
">SHPVIBE</p>
              <div class="absolute top-3 right-4 lg:hidden">
        <svg
          xmlns="http://www.w3.org/2000/svg"
          width="30"
          height="30"
          viewBox="0 0 16 16"
          id="hamburger"
        >
          <path
            fill="none"
            stroke="white"
            stroke-linecap="round"
            stroke-linejoin="round"
            stroke-width="1.5"
            d="M2.75 12.25h10.5m-10.5-4h10.5m-10.5-4h10.5"
          />
        </svg>
      </div>
      <nav>
        <ul
          id="navbar"
          class=" text-center text-white transition-all py-4 lg:flex
justify-center items-center gap-6 px-16"
        >
          <a href="/Home" id="home">
            <li class="py-1 hover:text-orange-400">Home</li>
          </a>
          <a   id="about" href="/about" >
            <li class="py-1 hover:text-orange-400">About</li>
          </a>
          <a  id="login" href="/login">
            <li class="py-1 hover:text-orange-400" >Login</li>
          </a>
          <a  id="signUp" href="/sign-up">
            <li class="py-1 hover:text-orange-400" >Register</li>
          </a>
          <a id="contact" href="/contact">
            <li class="py-1 hover:text-orange-400">Contact</li>
          </a>
        </ul>

              </header>

          {% endif %}

        </div>
      </div>
    </nav>
    {% block content %}

    {% endblock %}

    <!-- jQuery library -->
    <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></sc
ript>

    <!-- Popper JS -->
    <script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.16.0/umd/popper.min
.js"></script>


    <script>
      hamBurger = document
        .getElementById("hamburger")
        .addEventListener("click", function () {
          document.getElementById("navbar").classList.toggle("text-
center");
          document.getElementById("navbar").classList.toggle("hidden");
        });
    </script>

  </body>
</html>
```

Login.html

```
{% extends 'base.html' %} {% block title %}Login{% endblock %} {% block
content
%}

<main class="px-10">
  <form
    class="max-w-lg shadow-2xl mt-12 border rounded-md px-6 py-5 mx-auto
bg-blue-100 opacity-150"
    method="post"
  >
    <div class="pt-12">
      <label
        for="email"
        class="block mb-2 text-sm font-medium text-gray-900 dark:text-
white"
        >Your email</label
      >
      <input
        type="email"
        name="email"
        id="email"
        class="bg-gray-50 border border-gray-300 text-gray-900 sm:text-sm
rounded-lg focus:ring-primary-600 focus:border-primary-600 block w-full p-
2.5 dark:bg-gray-700 dark:border-gray-600 dark:placeholder-gray-400
dark:text-white dark:focus:ring-blue-500 dark:focus:border-blue-500"
        placeholder="name@company.com"
        required=""
      />
    </div>
    <div class="py-8">
      <label
        for="pass"
        class="block mb-2 text-sm font-medium text-gray-900 dark:text-
white"
```

```
        >Password</label
      >
      <input
        type="password"
        name="pass"
        id="pass"
        placeholder="••••••••"
        class="bg-gray-50 border border-gray-300 text-gray-900 sm:text-sm
rounded-lg focus:ring-primary-600 focus:border-primary-600 block w-full p-
2.5 dark:bg-gray-700 dark:border-gray-600 dark:placeholder-gray-400
dark:text-white dark:focus:ring-blue-500 dark:focus:border-blue-500"
        required=""
      />
    </div>
    <!-- <div class="flex items-center justify-between">
            <div class="flex items-start">
                <div class="flex items-center h-5">
                  <input id="remember" aria-describedby="remember"
type="checkbox" class="w-4 h-4 border border-gray-300 rounded bg-gray-50
focus:ring-3 focus:ring-primary-300 dark:bg-gray-700 dark:border-gray-600
dark:focus:ring-primary-600 dark:ring-offset-gray-800" required="">
                </div>
                <div class="ml-3 text-sm">
                  <label for="remember" class="text-gray-500 dark:text-
gray-300">Remember me</label>
                </div>
            </div>
            <a href="#" class="text-sm font-medium text-primary-600
hover:underline dark:text-primary-500">Forgot password?</a>
        </div> -->
    <div class="grid place-items-center py-8">
      <button type="submit" class="w-fit bg-yellow-200 rounded px-4 py-2">
```

```
        Sign in
      </button>
    </div>

    <p class="text-sm font-light text-gray-500 dark:text-gray-400">
      Don’t have an account yet?
      <a
        href="{{url_for('auth.sign_up')}}"
        class="font-medium text-primary-600 hover:underline dark:text-
primary-500"
        >Sign up</a
      >
    </p>
  </form>
</main>

{% endblock %}
```

SignUp.html

```
{% extends 'base.html' %} {% block title %}SignUp{% endblock %} {% block
content
%}

<div class="py-10">
  <form
    class="max-w-lg shadow-2xl mt-12 border rounded-md px-6 py-5 mx-auto
bg-blue-50 opacity-150"
    method="post"
  >
    <div class="py-2">
```

```
      <label
        for="email"
        class="block mb-2 text-sm font-medium text-gray-900 dark:text-
white"
        >Your email</label
      >
      <input
        type="email"
        name="email"
        id="email"
        class="bg-gray-50 border border-gray-300 text-gray-900 sm:text-sm
rounded-lg focus:ring-primary-600 focus:border-primary-600 block w-full p-
2.5 dark:bg-gray-700 dark:border-gray-600 dark:placeholder-gray-400
dark:text-white dark:focus:ring-blue-500 dark:focus:border-blue-500"
        placeholder="name@company.com"
        required
      />
    </div>
    <div class="py-2">
      <label
        for="name"
        class="block mb-2 text-sm font-medium text-gray-900 dark:text-
white"
        >Your Name</label
      >
      <input
        type="name"
        name="name"
        id="name"
        class="bg-gray-50 border border-gray-300 text-gray-900 sm:text-sm
rounded-lg focus:ring-primary-600 focus:border-primary-600 block w-full p-
```

```
2.5 dark:bg-gray-700 dark:border-gray-600 dark:placeholder-gray-400
dark:text-white dark:focus:ring-blue-500 dark:focus:border-blue-500"
        placeholder="name"
        required=""
      />
    </div>
    <div class="py-2">
      <label
        for="pass"
        class="block mb-2 text-sm font-medium text-gray-900 dark:text-
white"
        >Password</label
      >
      <input
        type="pass"
        name="pass"
        id="pass"
        placeholder="••••••••"
        class="bg-gray-50 border border-gray-300 text-gray-900 sm:text-sm
rounded-lg focus:ring-primary-600 focus:border-primary-600 block w-full p-
2.5 dark:bg-gray-700 dark:border-gray-600 dark:placeholder-gray-400
dark:text-white dark:focus:ring-blue-500 dark:focus:border-blue-500"
        required=""
      />
    </div>
    <div class="py-2">
      <label
        for="pass_conf"
        class="block mb-2 text-sm font-medium text-gray-900 dark:text-
white"
        >Confirm password</label
      >
      <input
        type="pass_conf"
        name="pass_conf"
        id="pass_conf"
        placeholder="••••••••"
        class="bg-gray-50 border border-gray-300 text-gray-900 sm:text-sm
rounded-lg focus:ring-primary-600 focus:border-primary-600 block w-full p-
2.5 dark:bg-gray-700 dark:border-gray-600 dark:placeholder-gray-400
dark:text-white dark:focus:ring-blue-500 dark:focus:border-blue-500"
        required=""
      />
    </div>

    <div class="grid place-items-center py-6">
      <button type="submit" class="w-fit bg-blue-300 px-4 py-3 rounded">
        Create an account
      </button>
    </div>

    <p class="text-sm font-light text-gray-700">
      Already have an account?
      <a
        href="{{url_for('auth.login')}}"
        class="font-medium text-primary-600 hover:underline dark:text-
primary-500"
        >Login here</a
      >
    </p>
  </form>
</div>

{% endblock %}
```

MemeInput.html

```
{% extends 'base.html' %} {% block title %}Home{% endblock %} {% block
content
%}
<section class="px-2">
  <form
    class="max-w-lg shadow-2xl mt-12 border rounded-md px-6 py-5 mx-auto
bg-blue-200 opacity-150"
    method="post"
  >
    <div class="">
      <label
        for="meme"
        class="block mb-2 text-sm font-medium text-gray-900 dark:text-
white"
        >Your Input</label
      >
      <input
        type="text"
        name="meme"
        id="meme"
        class="bg-gray-50 border border-gray-300 text-gray-900 sm:text-sm
rounded-lg focus:ring-primary-600 focus:border-primary-600 block w-full p-
2.5 dark:bg-gray-700 dark:border-gray-600 dark:placeholder-gray-400
dark:text-white dark:focus:ring-blue-500 dark:focus:border-blue-500"
        placeholder="Meme"
        required=""
      />
    </div>
    <div class="grid place-items-center py-8">
      <button type="submit" class="w-fit bg-orange-200 rounded px-4 py-2">
```

```
        Generate
      </button>
    </div>
  </form>
</section>
{% endblock %}
```

GitHub & Project Video Demo Link

Github Link: https://github.com/naanmudhalvan-SI/PBL-NT-GP-8344-1683280363

Vidoe Demo Link:
https://drive.google.com/file/d/1zfWmg0bygCa8nnENO66JkRiKyq6nuqHW/view?usp=share_link