

## Project Design Phase-|| Open Source Framework

Date	03 November 2023
Team ID	NM2023TMID02623
Project Name	Creating an Email Campaign Using Mailchimp

### Open Source Framework:

#### Flask:

Flask itself is a web framework for building web applications, but it can be used along with other Python libraries to create a web application that manages and initiates email campaigns.

#### Setup Flask:

Install Flask using `pip install Flask`. Create a Flask web application with routes and templates.

#### Integrate Email Sending:

Use a library like Flask-Mail to integrate email sending capabilities.

Configure your email server settings, such as SMTP server, username, and password.

#### Create Email Templates:

Use Flask's template engine to create HTML email templates.

Render these templates dynamically with information from your application.

#### Build Campaign Management:

Create routes and views for managing email campaigns.

Allow users to input campaign details, recipients, and email content.

#### Schedule or Trigger Email Sending:

Use a scheduler or trigger mechanism to send emails at scheduled times or based on certain events.

```
from flask import Flask, render_template, request
```

```
from flask_mail import Mail, Message
```

```
app = Flask(__name__)
```

```
app.config['MAIL_SERVER'] = 'your_smtp_server'
app.config['MAIL_PORT'] = 587
app.config['MAIL_USE_TLS'] = True
app.config['MAIL_USERNAME'] = 'your_email'
app.config['MAIL_PASSWORD'] = 'your_password'
mail = Mail(app)
@app.route('/')
def index():
    return render_template('index.html')
@app.route('/send_email', methods=['POST'])
def send_email():
    subject = request.form['subject']
    recipients = request.form['recipients'].split(',')
    body = request.form['body']
    msg = Message(subject=subject, recipients=recipients, body=body)
    mail.send(msg)
return 'Email sent successfully!'
if __name__ == '__main__':
    app.run(debug=True)
```

