```
!pip install flask flask-login flask-sqlalchemy
```

```
Requirement already satisfied: flask in c:\users\khura\anaconda3\lib\site-packages (3.1.0)
Requirement already satisfied: flask-login in c:\users\khura\anaconda3\lib\site-packages (0.6
Requirement already satisfied: flask-sqlalchemy in c:\users\khura\anaconda3\lib\site-packages
Requirement already satisfied: Werkzeug>=3.1 in c:\users\khura\anaconda3\lib\site-packages (3
Requirement already satisfied: Jinja2>=3.1.2 in c:\users\khura\anaconda3\lib\site-packages (3
Requirement already satisfied: itsdangerous>=2.2 in c:\users\khura\anaconda3\lib\site-package
Requirement already satisfied: click>=8.1.3 in c:\users\khura\anaconda3\lib\site-packages (fr
Requirement already satisfied: blinker>=1.9 in c:\users\khura\anaconda3\lib\site-packages (fr
Requirement already satisfied: sqlalchemy>=2.0.16 in c:\users\khura\anaconda3\lib\site-packag
Requirement already satisfied: colorama in c:\users\khura\anaconda3\lib\site-packages (from c
Requirement already satisfied: MarkupSafe>=2.0 in c:\users\khura\anaconda3\lib\site-packages
Requirement already satisfied: greenlet!=0.4.17 in c:\users\khura\anaconda3\lib\site-packages
Requirement already satisfied: typing-extensions>=4.6.0 in c:\users\khura\anaconda3\lib\site-


[notice] A new release of pip is available: 24.3.1 -> 25.0
[notice] To update, run: python.exe -m pip install --upgrade pip
```

```
! pip install flask_bcrypt
```

```
Collecting flask_bcrypt
  Downloading Flask_Bcrypt-1.0.1-py3-none-any.whl.metadata (2.6 kB)
Requirement already satisfied: Flask in c:\users\khura\anaconda3\lib\site-packages (from flas
Requirement already satisfied: bcrypt>=3.1.1 in c:\users\khura\anaconda3\lib\site-packages (3
Requirement already satisfied: cffi>=1.1 in c:\users\khura\anaconda3\lib\site-packages (from
Requirement already satisfied: six>=1.4.1 in c:\users\khura\anaconda3\lib\site-packages (from
Requirement already satisfied: Werkzeug>=3.1 in c:\users\khura\anaconda3\lib\site-packages (3
Requirement already satisfied: Jinja2>=3.1.2 in c:\users\khura\anaconda3\lib\site-packages (3
Requirement already satisfied: itsdangerous>=2.2 in c:\users\khura\anaconda3\lib\site-package
Requirement already satisfied: click>=8.1.3 in c:\users\khura\anaconda3\lib\site-packages (fr
Requirement already satisfied: blinker>=1.9 in c:\users\khura\anaconda3\lib\site-packages (fr
Requirement already satisfied: pycparser in c:\users\khura\anaconda3\lib\site-packages (from
Requirement already satisfied: colorama in c:\users\khura\anaconda3\lib\site-packages (from c
Requirement already satisfied: MarkupSafe>=2.0 in c:\users\khura\anaconda3\lib\site-packages
Downloading Flask_Bcrypt-1.0.1-py3-none-any.whl (6.0 kB)
Installing collected packages: flask_bcrypt
Successfully installed flask_bcrypt-1.0.1
```

```
[notice] A new release of pip is available: 24.3.1 -> 25.0
[notice] To update, run: python.exe -m pip install --upgrade pip
```

```
! pip install pymysql
```

```
Collecting pymysql
  Downloading PyMySQL-1.1.1-py3-none-any.whl.metadata (4.4 kB)
Downloading PyMySQL-1.1.1-py3-none-any.whl (44 kB)
Installing collected packages: pymysql
Successfully installed pymysql-1.1.1
```

```
[notice] A new release of pip is available: 24.3.1 -> 25.0
[notice] To update, run: python.exe -m pip install --upgrade pip
```

```
import os
os.urandom(24).hex()
```

```
'f8e323875a6103f71c9705a03f5f6fc2698a33e1894c1de6'
```

```
from flask import Flask, render_template, request, redirect, url_for, flash
from flask_sqlalchemy import SQLAlchemy
from flask_login import LoginManager, UserMixin, login_user, login_required, logout_user, cu
from flask_bcrypt import Bcrypt
import secrets

app = Flask(__name__)
app.secret_key = 'f8e323875a6103f71c9705a03f5f6fc2698a33e1894c1de6'

# Database configuration
app.config['SQLALCHEMY_DATABASE_URI'] = 'mysql+pymysql://my_user:password@localhost/my_databa
app.config['SQLALCHEMY_TRACK_MODIFICATIONS'] = False

db = SQLAlchemy(app)
bcrypt = Bcrypt(app)
login_manager = LoginManager(app)
login_manager.login_view = 'login'

# User model
```

```
class User(db.Model, UserMixin):
    id = db.Column(db.Integer, primary_key=True)
    email = db.Column(db.String(255), unique=True, nullable=False)
    password_hash = db.Column(db.String(255), nullable=False)
    email_verified = db.Column(db.Boolean, default=False)
    verification_token = db.Column(db.String(255))
    created_at = db.Column(db.TIMESTAMP, server_default=db.func.current_timestamp())
    updated_at = db.Column(db.TIMESTAMP, server_default=db.func.current_timestamp(), onupdate
    last_login = db.Column(db.TIMESTAMP)
    reset_token = db.Column(db.String(255))
    reset_token_expires = db.Column(db.TIMESTAMP)
    status = db.Column(db.String(50), default='active')

    def set_password(self, password):
        self.password_hash = bcrypt.generate_password_hash(password).decode('utf-8')

    def check_password(self, password):
        return bcrypt.check_password_hash(self.password_hash, password)

# Load user for Flask-Login
@login_manager.user_loader
def load_user(user_id):
    return User.query.get(int(user_id))

# Routes
@app.route('/')
def home():
    return render_template('home.html')

@app.route('/register', methods=['GET', 'POST'])
def register():
    if request.method == 'POST':
        email = request.form.get('email')
        password = request.form.get('password')

        # Check if user already exists
        if User.query.filter_by(email=email).first():
            flash('Email already registered!', 'danger')
            return redirect(url_for('register'))

        # Create new user
        new_user = User(email=email)
```

```
        new_user.set_password(password)
        new_user.verification_token = secrets.token_urlsafe(32)  # Generate verification toke

        db.session.add(new_user)
        db.session.commit()

        # Send verification email (pseudo-code)
        send_verification_email(new_user.email, new_user.verification_token)

        flash('Registration successful! Please check your email to verify your account.', 'su
        return redirect(url_for('login'))

    return render_template('register.html')

@app.route('/verify/<token>')
def verify_email(token):
    user = User.query.filter_by(verification_token=token).first()

    if user:
        user.email_verified = True
        user.verification_token = None  # Clear the token after verification
        db.session.commit()
        flash('Email verified successfully!', 'success')
    else:
        flash('Invalid or expired token.', 'danger')

    return redirect(url_for('login'))

@app.route('/login', methods=['GET', 'POST'])
def login():
    if request.method == 'POST':
        email = request.form.get('email')
        password = request.form.get('password')

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

        if user and user.check_password(password):
            if user.email_verified:
                login_user(user)
                user.last_login = db.func.current_timestamp()  # Update last login time
                db.session.commit()
                flash('Login successful!', 'success')
```