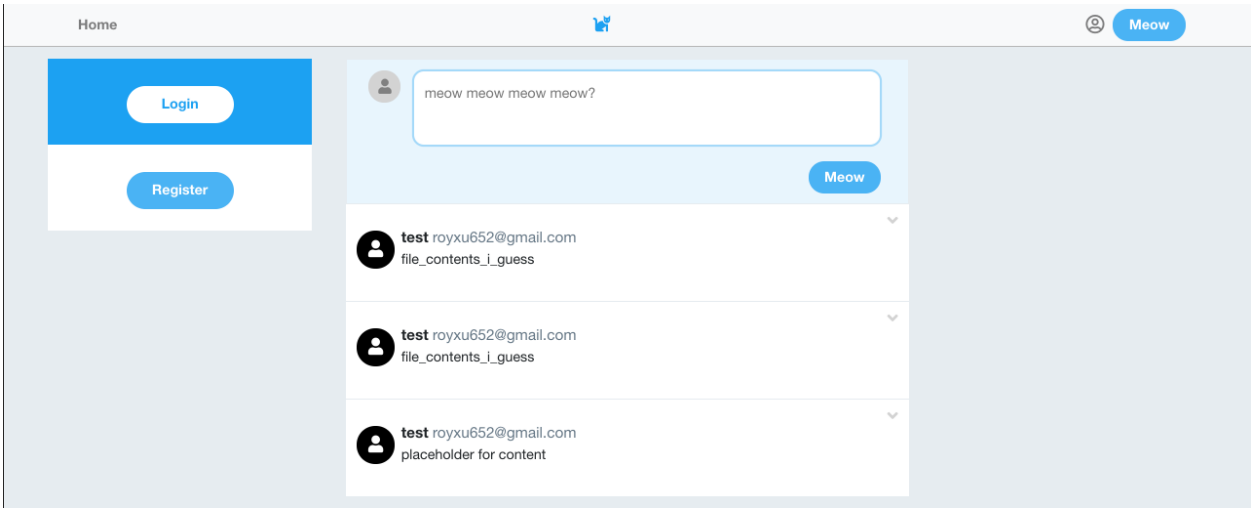
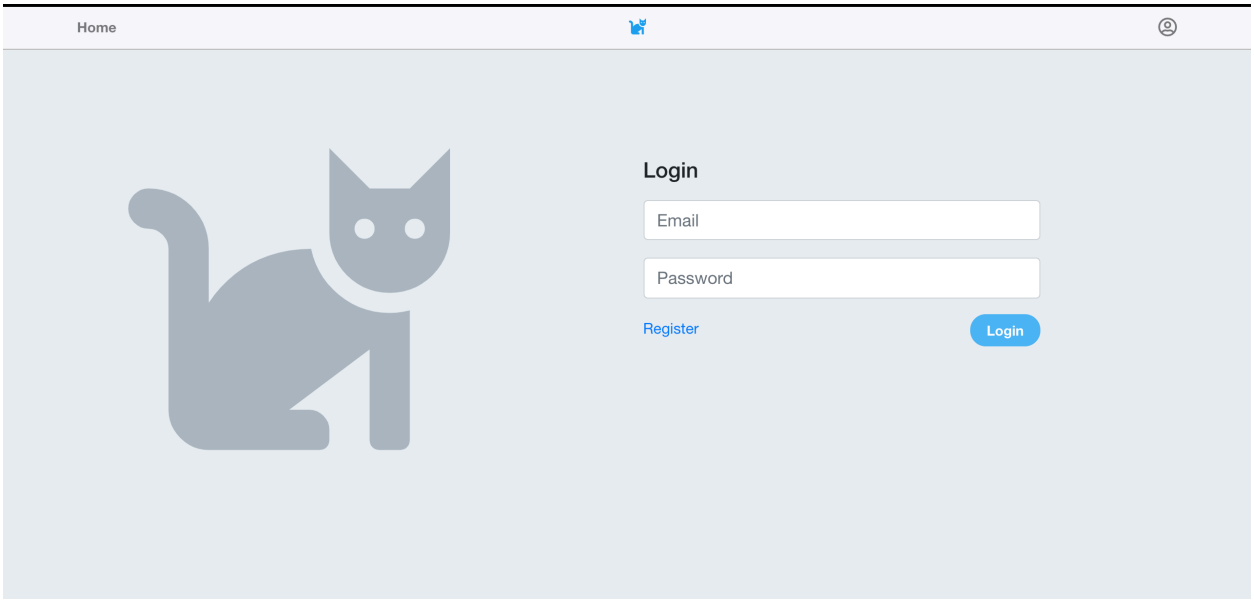
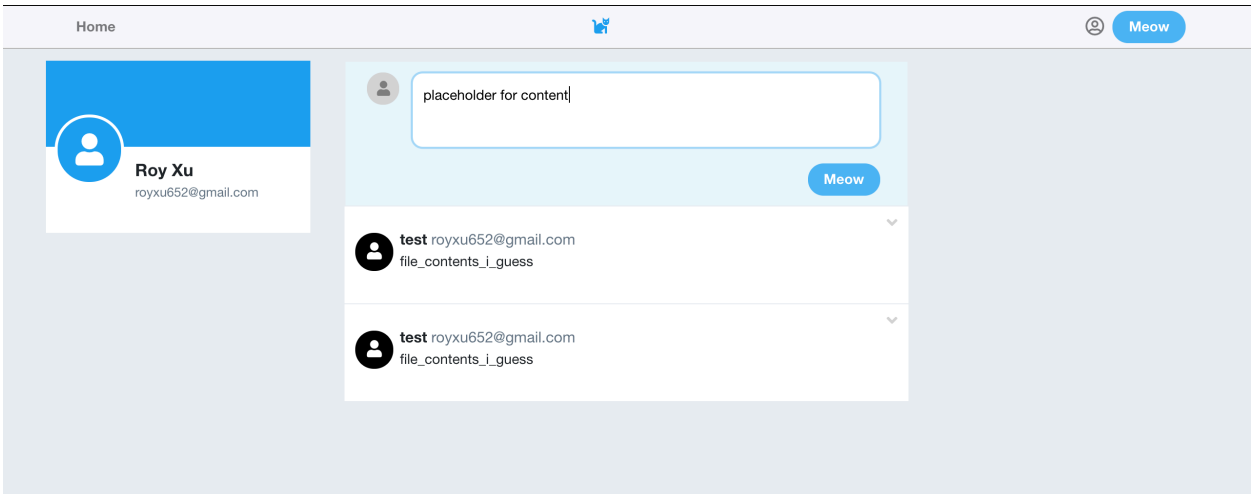
Github Repo – https://github.com/svv232/DB_Project
Credits to Bootstrap 4 and FontAwesome



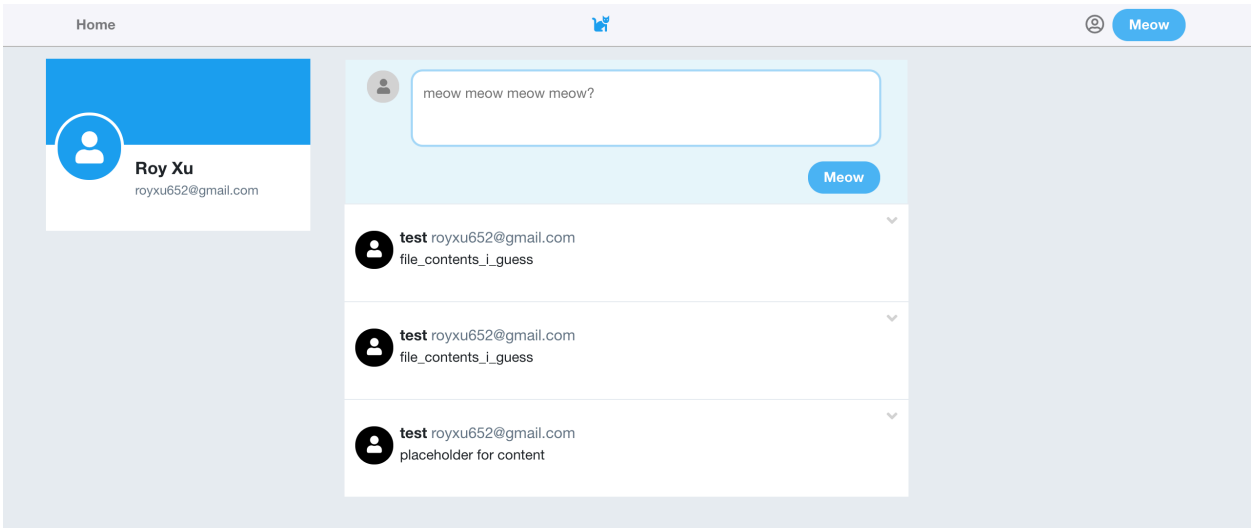
Homepage without logging in



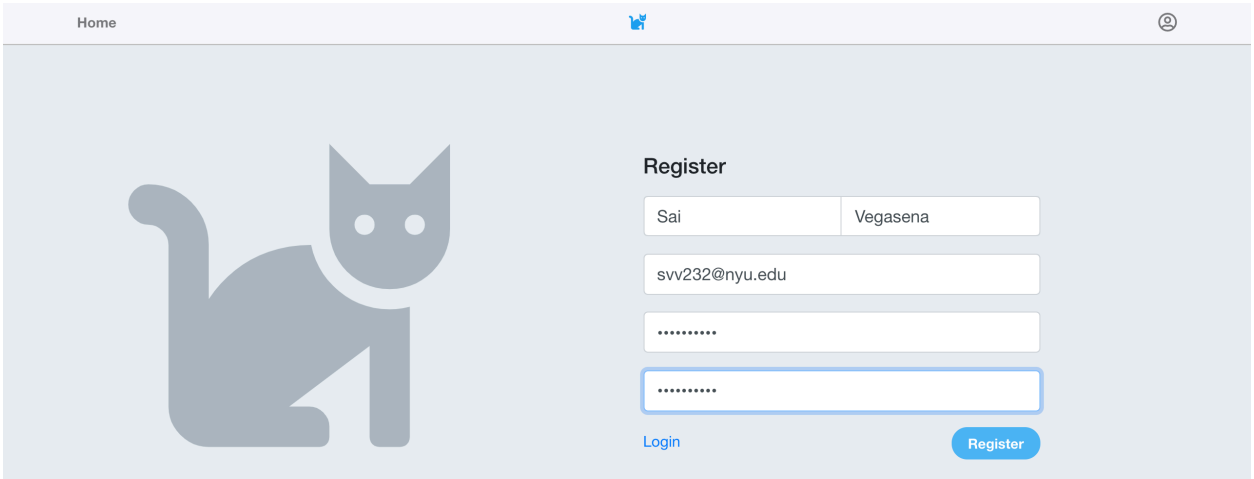
Login Page



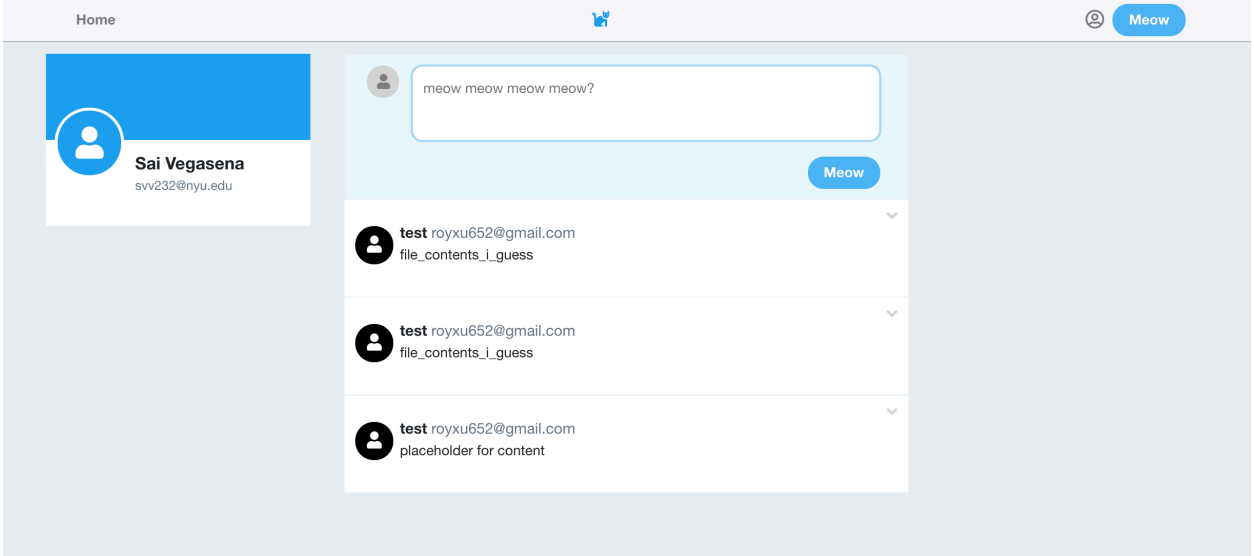
Posting placeholder content



Placeholder content example visible



Registration Page



Public posts viewable as another user

```python
@app.route('/')
def index():
    _, posts = get_public_content()

    if 'email' not in session:
        return render_template('index.html', posts=posts)

    return render_template('index.html', email=session['email'],
                           fname=session['fname'], lname=session['lname'],
                           posts=posts)


@app.route('/login', methods=['GET', 'POST'])
def login():
    if request.method == 'GET':
        return render_template('login.html')

    email = request.form['email']
    password = request.form['password']

    success, data = login_user(email, password)

    if not success:
        return render_template('login.html', error=data)

    session['email'] = email
    session['fname'] = data['fname']
    session['lname'] = data['lname']
    return redirect('/')
```

Code for Index and Login Page

```python
@app.route('/register', methods=['GET', 'POST'])
def register():
    if request.method == 'GET':
        return render_template('register.html')

    email = request.form['email']
    password = request.form['password']
    fname = request.form['fname']
    lname = request.form['lname']
    confirm = request.form['confirm']

    if password != confirm:
        error = 'Password does not match'
        return render_template('register.html', error=error)

    success, message = register_user(email, password, fname, lname)
    if not success:
        return render_template('register.html', error=message)

    session['email'] = email
    session['fname'] = fname
    session['lname'] = lname
    return redirect('/')


@app.route('/logout')
def logout():
    session.pop('email')
    return redirect('/')


@app.route('/post', methods=['POST'])
@login_required
def post():
    email = session['email']
    post = request.form.get('submission-text')
    post_content(email, 'test', post, True)
    return redirect('/')


@app.errorhandler(404)
def page_not_found(e):
    return render_template('404.html'), 404
```

Register, Logout, Creating Post, and 404

```python
import pymysql.cursors
from hashlib import sha256

conn = pymysql.connect(host='localhost',
                       port=3306,
                       user='PriCoSha',
                       password='PriCoSha',
                       db='PriCoSha',
                       charset='utf8mb4',
                       cursorclass=pymysql.cursors.DictCursor)


def login_user(email, password):
    cursor = conn.cursor()
    query = 'SELECT * FROM Person WHERE email = %s and password = %s'
    password = sha256(password.encode('utf-8')).hexdigest()

    cursor.execute(query, (email, password))
    user = cursor.fetchone()
    cursor.close()
    res = user is not None
    message = user if res else "Invalid Username or Password"
    return res, message


def register_user(email, password, fname, lname):
    cursor = conn.cursor()
    query = 'SELECT * FROM Person WHERE email = %s'
    cursor.execute(query, (email))
    data = cursor.fetchone()

    if data:
        return False, "User already exists"

    insert = 'INSERT INTO Person VALUES(%s, %s, %s, %s)'
    password = sha256(password.encode('utf-8')).hexdigest()
    cursor.execute(insert, (email, password, fname, lname))
    conn.commit()
    cursor.close()

    return True, "User successfully registered"
```

Database operations – Register and Login

```python
def get_public_content():
    cursor = conn.cursor()
    query = ('SELECT item_id, email_post, post_time, file_path, item_name ' +
             'FROM ContentItem WHERE is_pub = TRUE AND post_time <= now() + ' +
             'INTERVAL 1 DAY')
    cursor.execute(query)
    posts = cursor.fetchall()
    cursor.close()

    return True, posts


def post_content(email_post, item_name, file_path, is_pub):
    cursor = conn.cursor()
    query = ('INSERT INTO ContentItem (email_post, post_time, file_path, ' +
             'item_name, is_pub) values (%s, NOW(), %s, %s, %s)')
    cursor.execute(query, (email_post, file_path, item_name, is_pub))
    conn.commit()
    cursor.close()

    return True, "Item Sucessfully Posted"
```

Database operations – Get Content and Post Content

Feature Proposals

1. Content Specific metadata (9 in optional features)
2. Add comments
3. Defriend
4. Tagging a group
5. Change profile info (email, name, etc.)
6. Leave friend group
7. Best friend list
8. Filter by best friends / friend group
9. Filter by timestamps