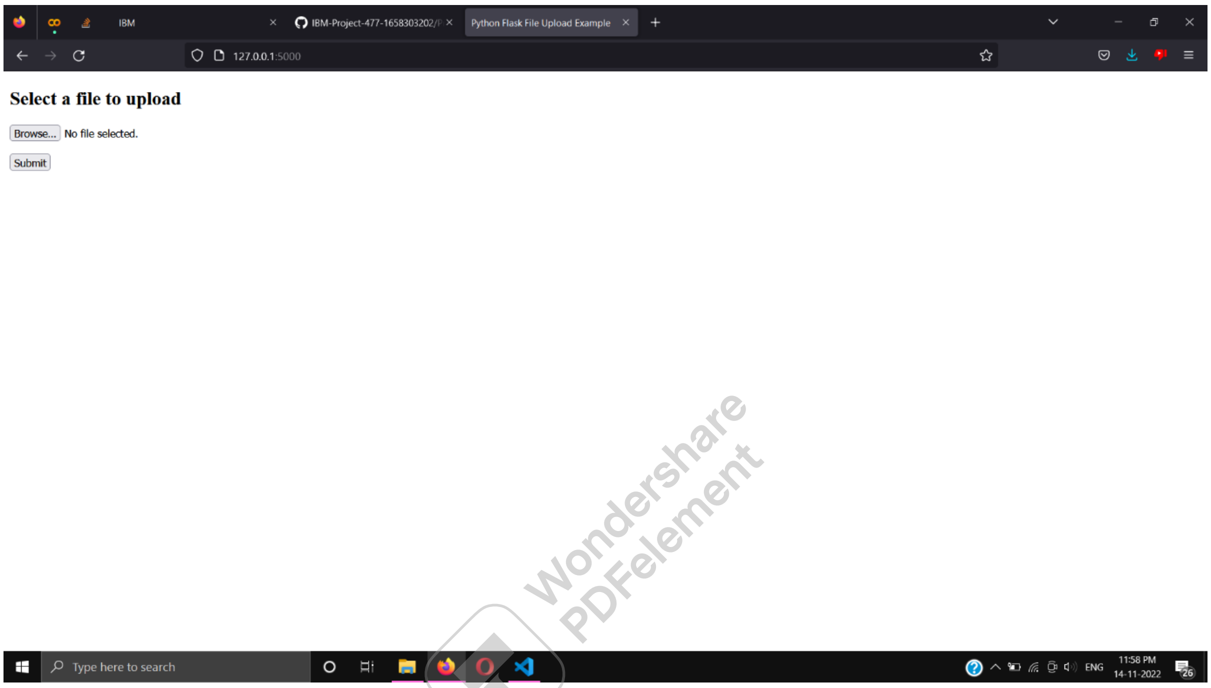
Date	12 November 2022
Team ID	PNT2022TMID52124
Project Name	Digital Naturalist - AI Enabled Tool For Biodiversity Researchers
Maximum Marks	4 Marks

Front-End:

1. Home page for uploading input

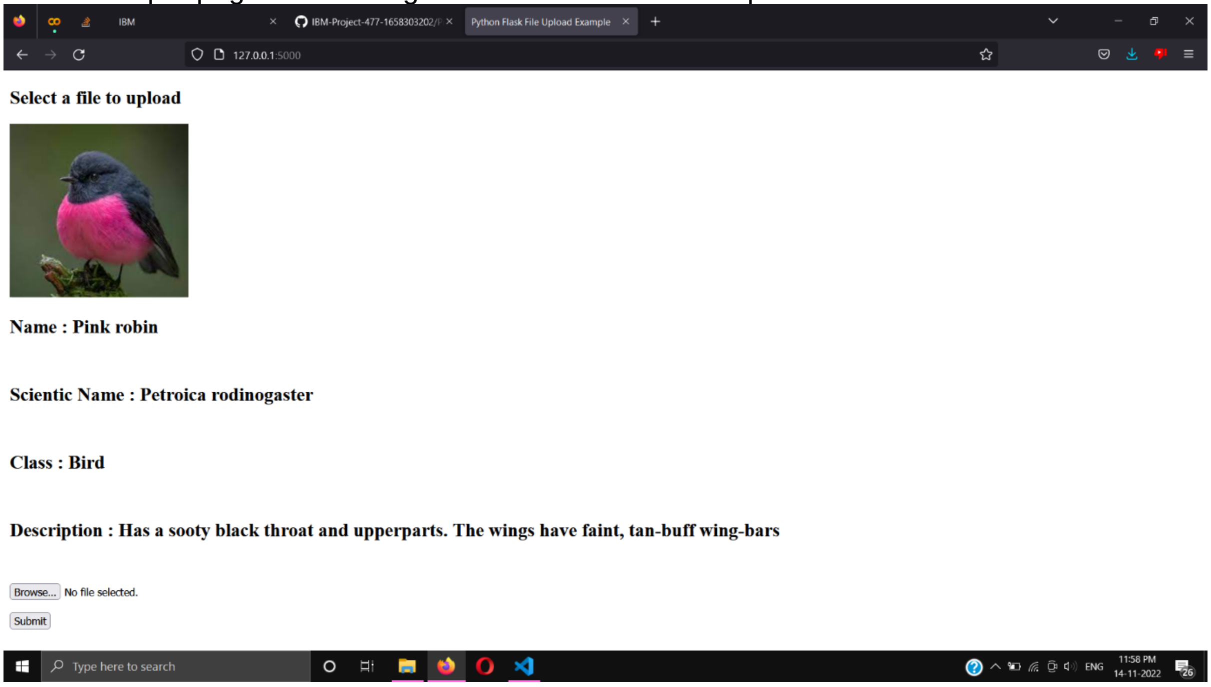


Select a file to upload

Browse... No file selected.

Submit

2. Output page for showing the classification of the species



Select a file to upload

Name : Pink robin

Scientic Name : Petroica rodinogaster

Class : Bird

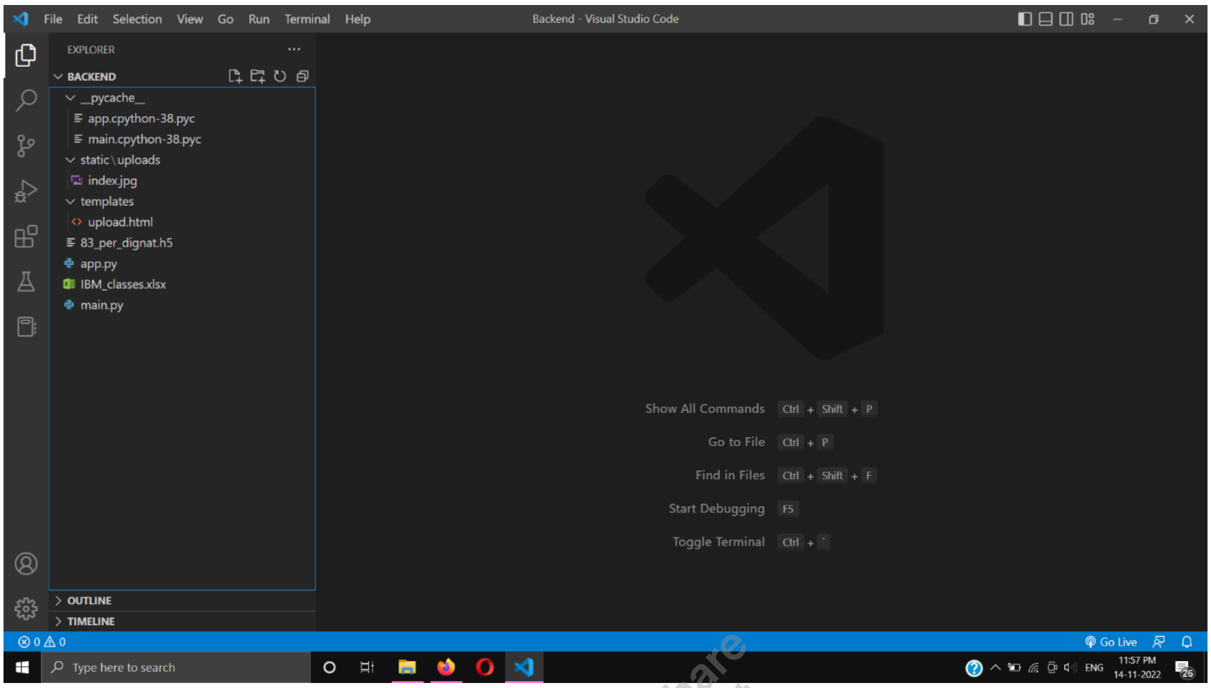
Description : Has a sooty black throat and upperparts. The wings have faint, tan-buff wing-bars

Browse... No file selected.

Submit

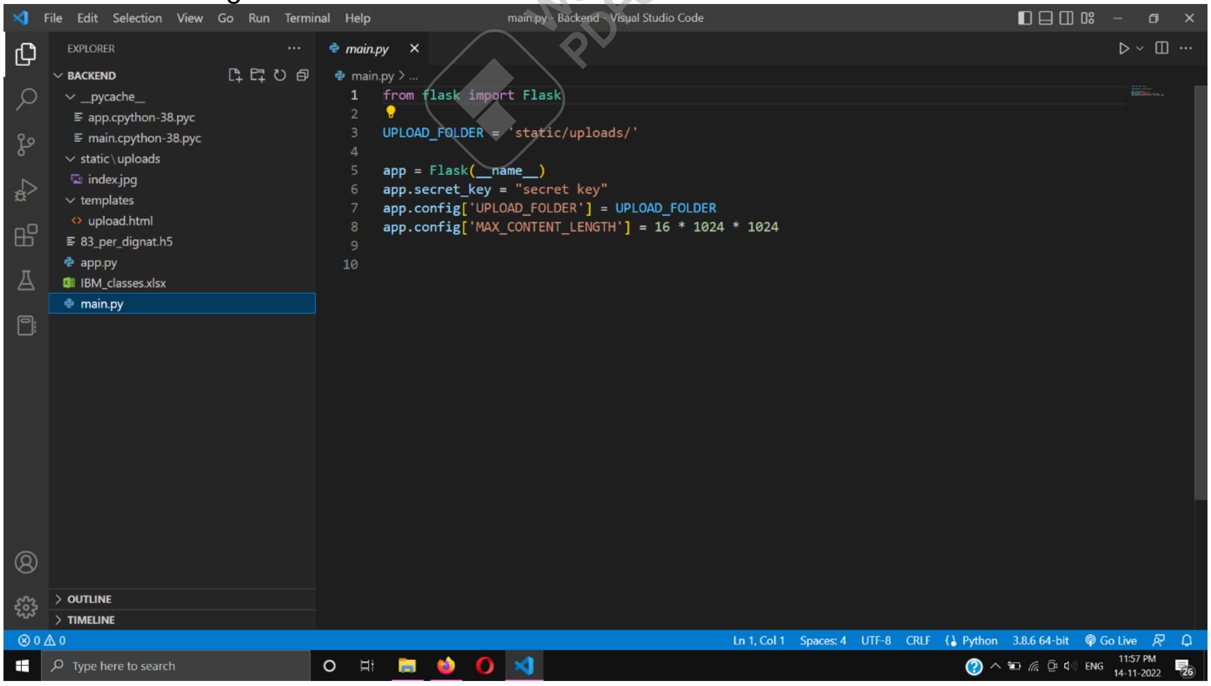
Backend using Flask

1. Application directory


File Edit Selection View Go Run Terminal Help
Backend - Visual Studio Code
EXPLORER
Show All Commands Ctrl + Shift + P
Go to File Ctrl + P
Find in Files Ctrl + Shift + F
Start Debugging F5
Toggle Terminal Ctrl + `
OUTLINE
TIMELINE
Go Live
Type here to search
11:57 PM
14-11-2022

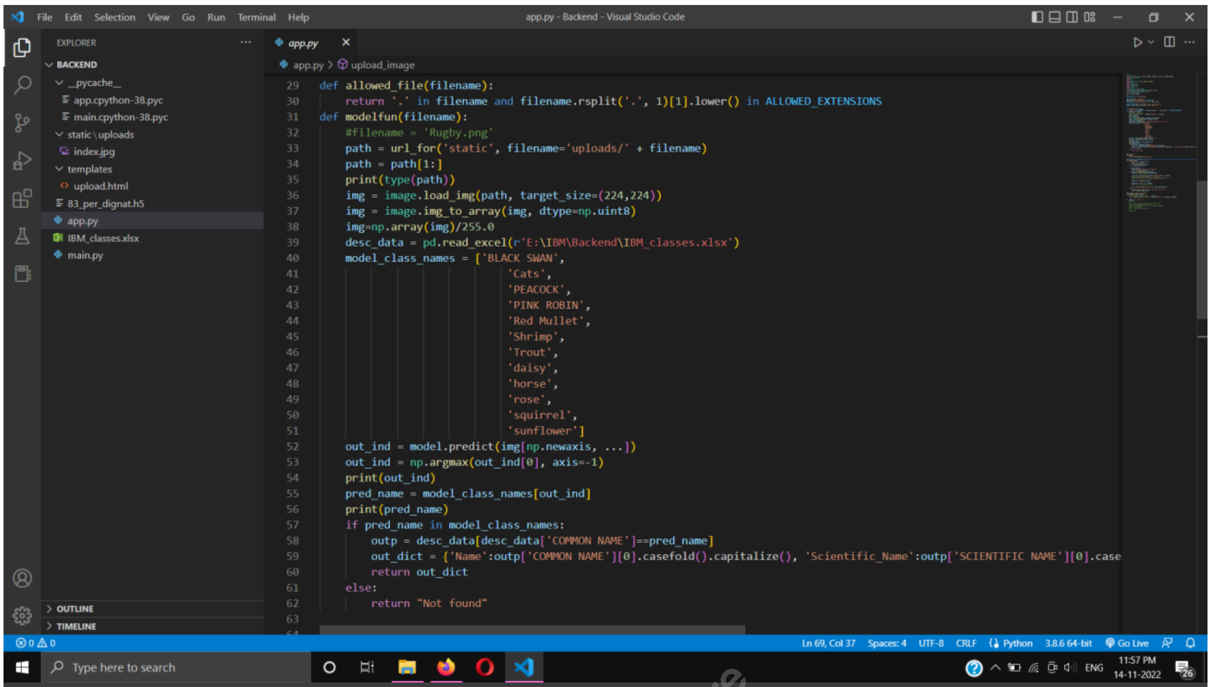
```
BACKEND
  __pycache__
    app.cpython-38.pyc
    main.cpython-38.pyc
  static\uploads
    index.jpg
  templates
    upload.html
  83_per_dignat.h5
  app.py
  IBM_classes.xlsx
  main.py
```

2. Making a Flask Instance

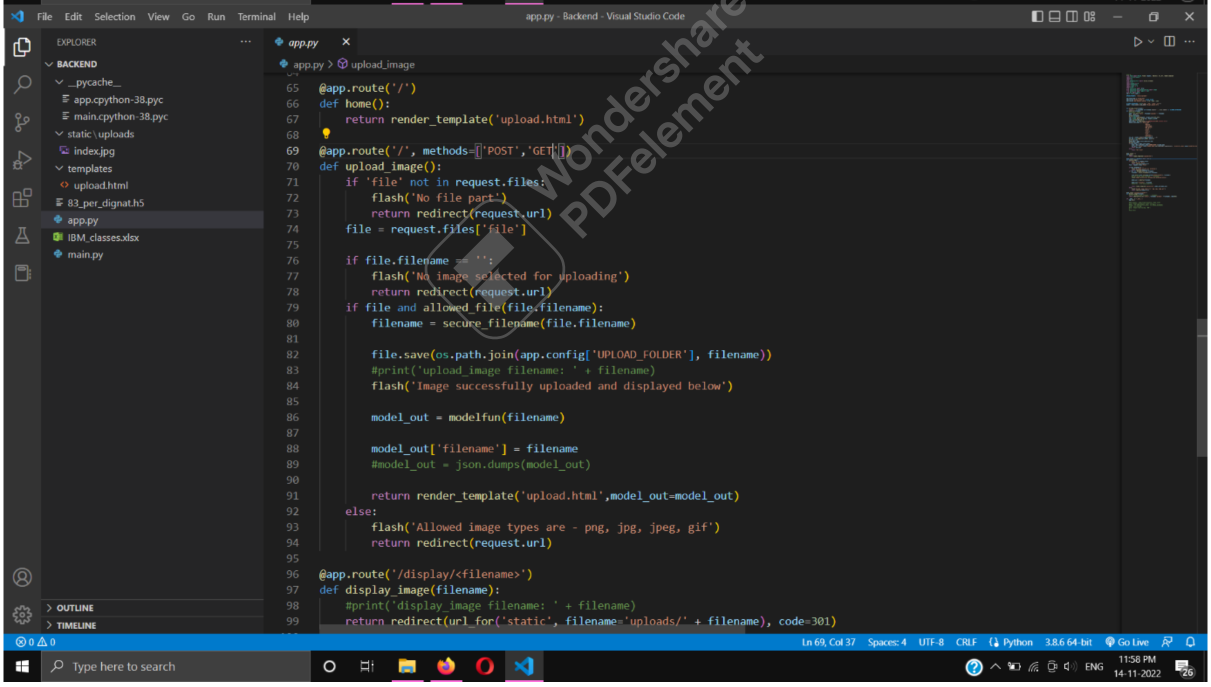

File Edit Selection View Go Run Terminal Help
main.py - Backend - Visual Studio Code
EXPLORER
main.py
main.py > ...
OUTLINE
TIMELINE
Ln 1, Col 1 Spaces: 4 UTF-8 CRLF Python 3.8.6 64-bit Go Live
Type here to search
11:57 PM
14-11-2022

```python
from flask import Flask

UPLOAD_FOLDER = 'static/uploads/'

app = Flask(__name__)
app.secret_key = "secret key"
app.config['UPLOAD_FOLDER'] = UPLOAD_FOLDER
app.config['MAX_CONTENT_LENGTH'] = 16 * 1024 * 1024
```

3. App.py for handling the requests and response. This contains the route to the HTML files

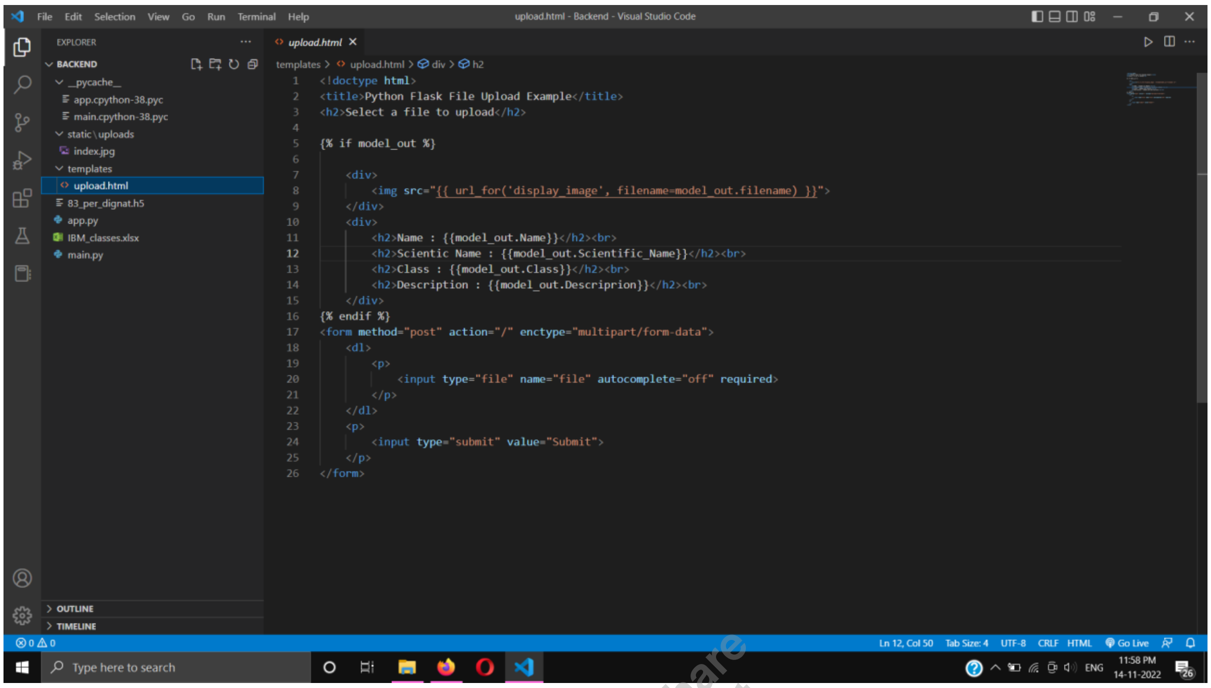


```
def allowed_file(filename):
    return '.' in filename and filename.rsplit('.', 1)[1].lower() in ALLOWED_EXTENSIONS
def modelfun(filename):
    #filename = 'Rugby.png'
    path = url_for('static', filename='uploads/' + filename)
    path = path[1:]
    print(type(path))
    img = image.load_img(path, target_size=(224,224))
    img = image.img_to_array(img, dtype=np.uint8)
    img=np.array(img)/255.0
    desc_data = pd.read_excel(r'E:\IBM\Backend\IBM_classes.xlsx')
    model_class_names = ['BLACK SWAN',
                         'Cats',
                         'PEACOCK',
                         'PINK ROBIN',
                         'Red Mullet',
                         'Shrimp',
                         'Trout',
                         'daisy',
                         'horse',
                         'rose',
                         'squirrel',
                         'sunflower']
    out_ind = model.predict(img[np.newaxis, ...])
    out_ind = np.argmax(out_ind[0], axis=-1)
    print(out_ind)
    pred_name = model_class_names[out_ind]
    print(pred_name)
    if pred_name in model_class_names:
        outp = desc_data[desc_data['COMMON NAME']==pred_name]
        out_dict = {'Name':outp['COMMON NAME'][0].casefold().capitalize(), 'Scientific_Name':outp['SCIENTIFIC NAME'][0].case
        return out_dict
    else:
        return "Not found"
```



```
@app.route('/')
def home():
    return render_template('upload.html')

@app.route('/', methods=['POST','GET'])
def upload_image():
    if 'file' not in request.files:
        flash('No file part')
        return redirect(request.url)
    file = request.files['file']

    if file.filename == '':
        flash('No image selected for uploading')
        return redirect(request.url)
    if file and allowed_file(file.filename):
        filename = secure_filename(file.filename)

        file.save(os.path.join(app.config['UPLOAD_FOLDER'], filename))
        #print('upload_image filename: ' + filename)
        flash('Image successfully uploaded and displayed below')

        model_out = modelfun(filename)

        model_out['filename'] = filename
        #model_out = json.dumps(model_out)

        return render_template('upload.html',model_out=model_out)
    else:
        flash('Allowed image types are - png, jpg, jpeg, gif')
        return redirect(request.url)

@app.route('/display/<filename>')
def display_image(filename):
    #print('display_image filename: ' + filename)
    return redirect(url_for('static', filename='uploads/' + filename), code=301)
```

4. Added Flask Jinja 2 tags in HTML Pages to get and display the data from backend



```
<!doctype html>
<title>Python Flask File Upload Example</title>
<h2>Select a file to upload</h2>

{% if model_out %}

    <div>
        <img src="{{ url_for('display_image', filename=model_out.filename) }}">
    </div>
    <div>
        <h2>Name : {{model_out.Name}}</h2><br>
        <h2>Scientic Name : {{model_out.Scientific_Name}}</h2><br>
        <h2>Class : {{model_out.Class}}</h2><br>
        <h2>Description : {{model_out.Descriprion}}</h2><br>
    </div>
{% endif %}
<form method="post" action="/" enctype="multipart/form-data">
    <dl>
        <p>
            <input type="file" name="file" autocomplete="off" required>
        </p>
    </dl>
    <p>
        <input type="submit" value="Submit">
    </p>
</form>
```