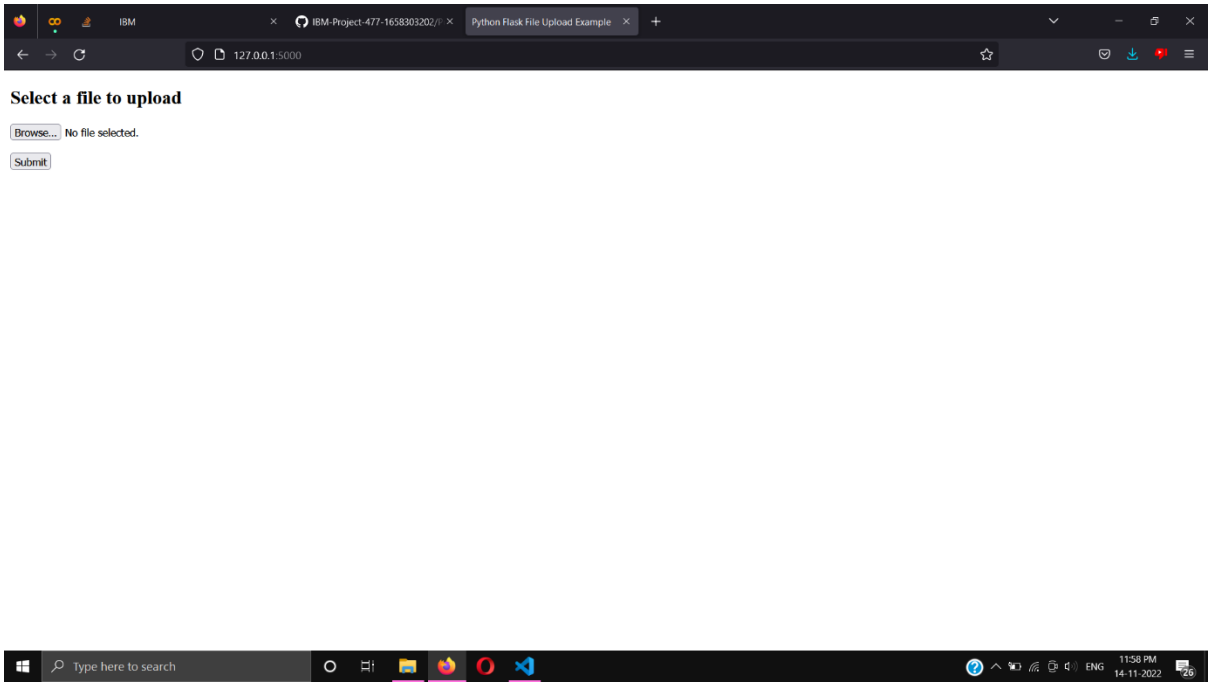
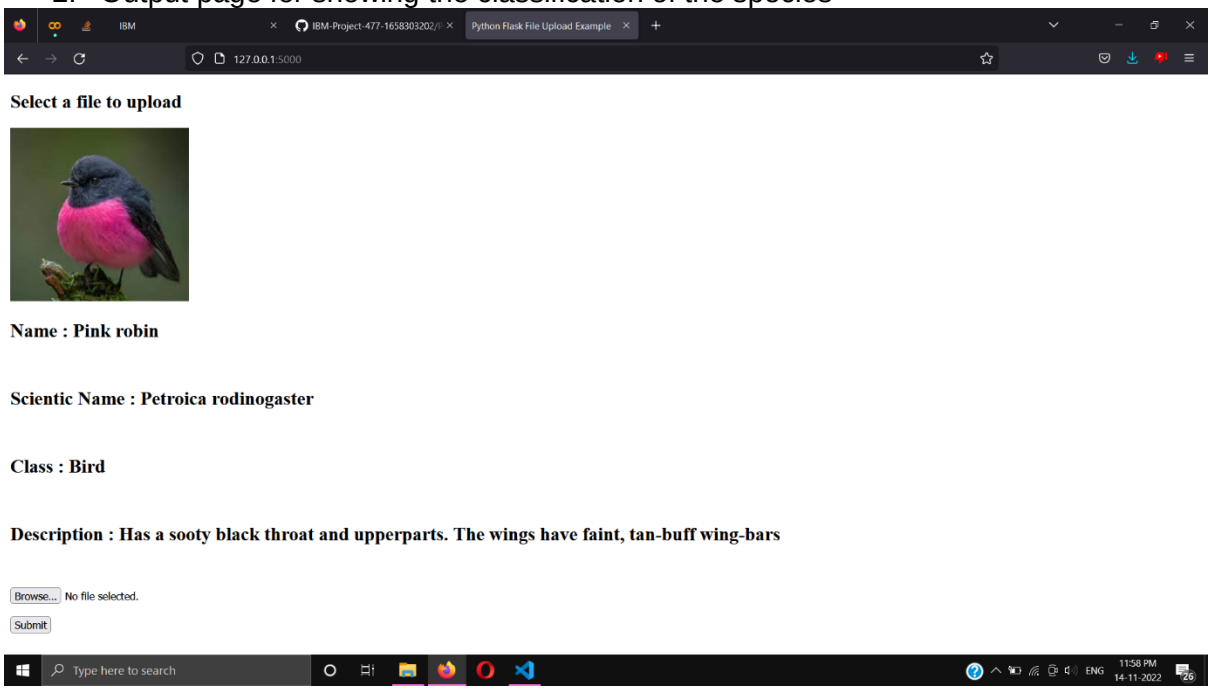
Date	12 November 2022
Team ID	PNT2022TMID35899
Project Name	Digital Naturalist - AI Enabled Tool For Biodiversity Researchers
Maximum Marks	4 Marks

Front-End:

1. Home page for uploading input

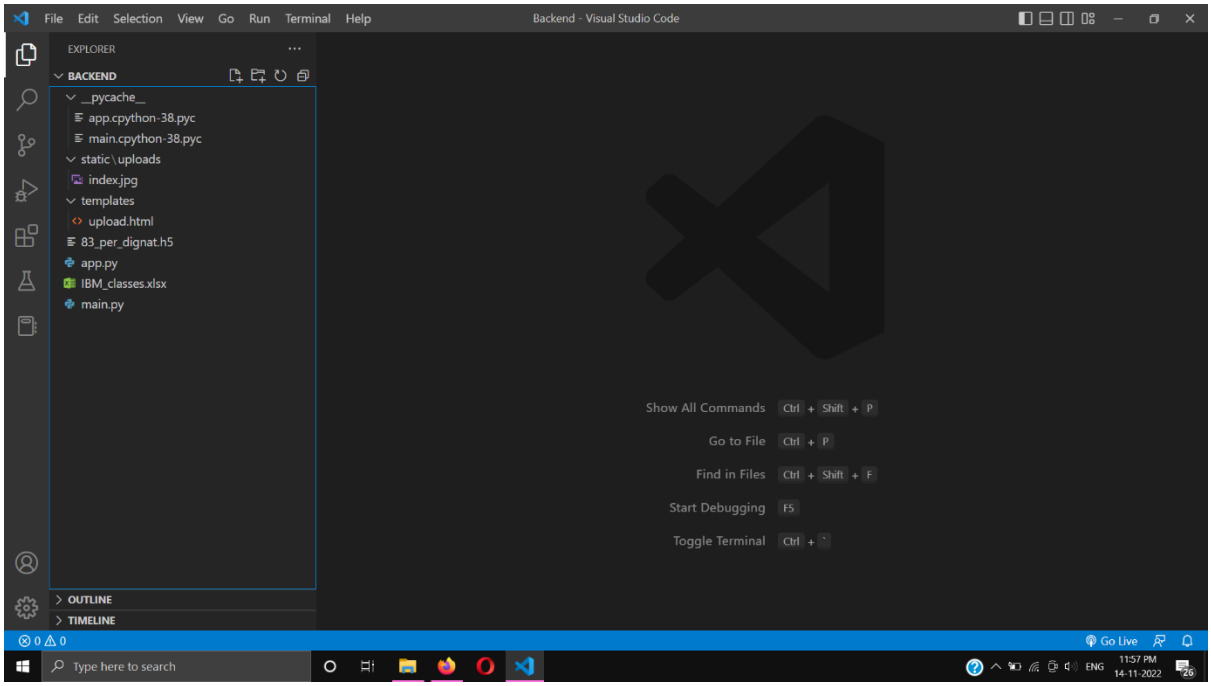


2. Output page for showing the classification of the species

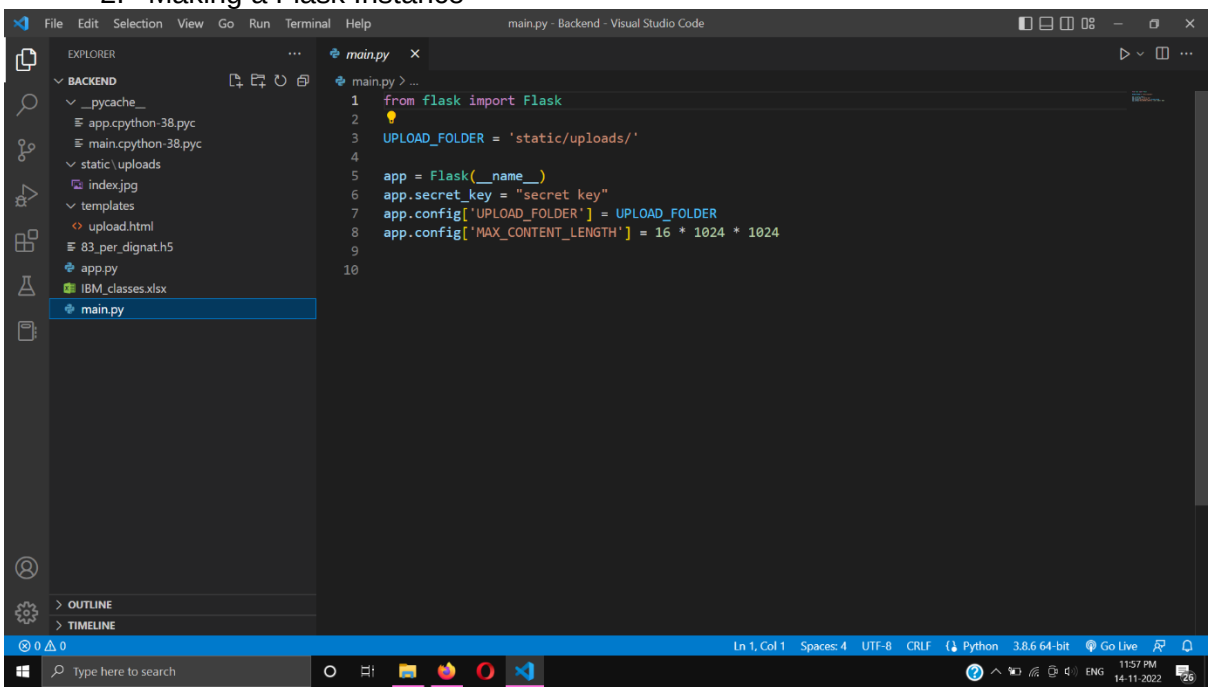


Backend using Flask

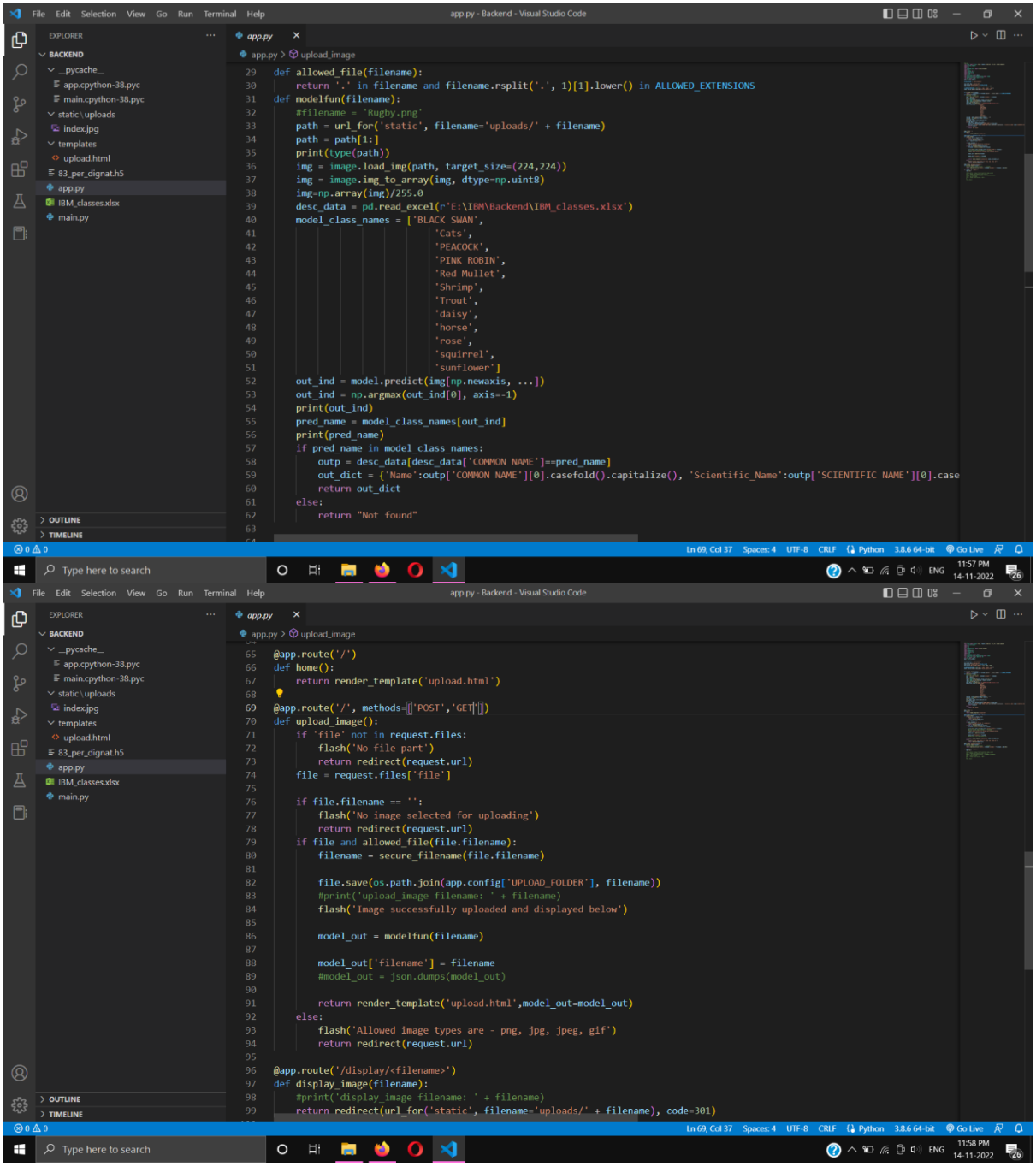
1. Application directory



2. Making a Flask Instance



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