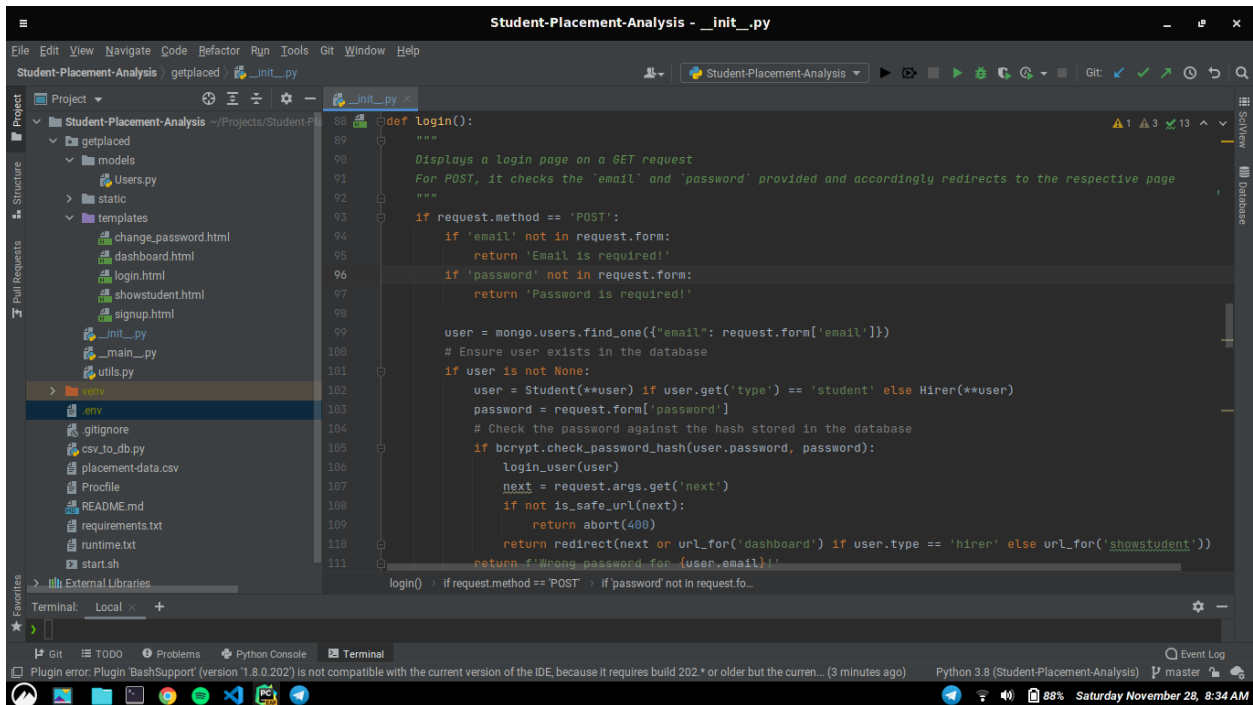


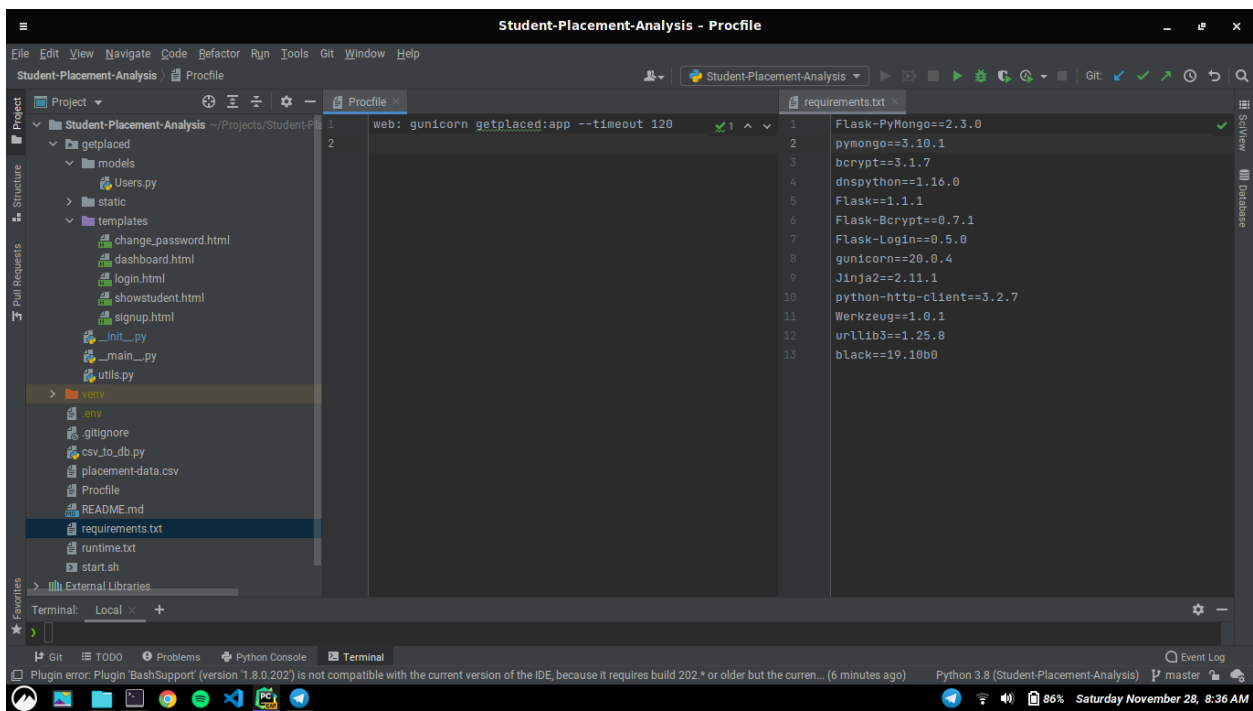
Created web project



The screenshot shows the PyCharm IDE with the file `_init_.py` open. The code defines a `login()` function that handles GET and POST requests. For POST requests, it checks if email and password are provided, hashes the password, and checks it against the database. If successful, it redirects to the dashboard or shows the student profile based on the user type.

```
def login():  
    """  
    Displays a login page on a GET request  
    For POST, it checks the 'email' and 'password' provided and accordingly redirects to the respective page  
    """  
    if request.method == 'POST':  
        if 'email' not in request.form:  
            return 'Email is required!'  
        if 'password' not in request.form:  
            return 'Password is required!'  
  
        user = mongo.users.find_one({"email": request.form['email']})  
        # Ensure user exists in the database  
        if user is not None:  
            user = Student(**user) if user.get('type') == 'student' else Hirer(**user)  
            password = request.form['password']  
            # Check the password against the hash stored in the database  
            if bcrypt.check_password_hash(user.password, password):  
                login_user(user)  
                next = request.args.get('next')  
                if not is_safe_url(next):  
                    return abort(400)  
                return redirect(next or url_for('dashboard')) if user.type == 'hirer' else url_for('showstudent')  
            return f'Wrong password for {user.email}'  
    return 'Wrong password for {user.email}'  
login() if request.method == 'POST' else if 'password' not in request.form...
```

Making sure the requirements are specified and the necessary Procfile is ready



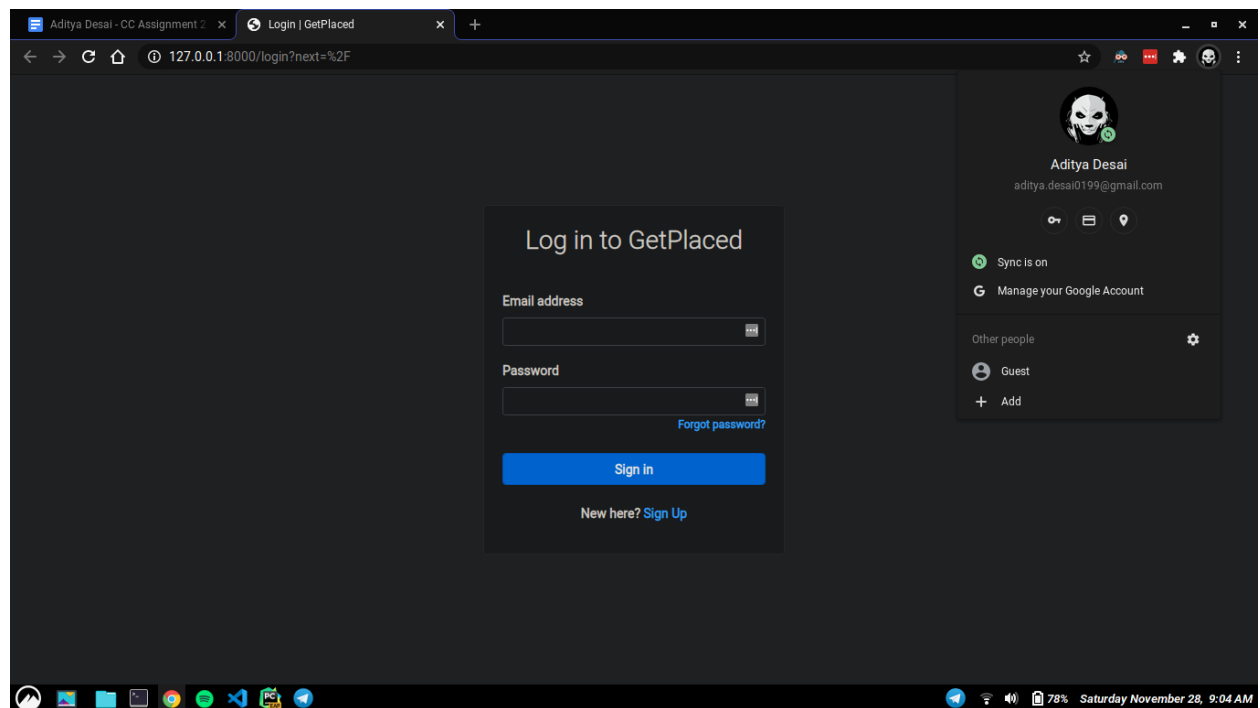
The screenshot shows the PyCharm IDE with the `Procfile` and `requirements.txt` files open. The `Procfile` defines the web application using `gunicorn` and `getplaced:app`. The `requirements.txt` lists the necessary Python dependencies.

```
web: gunicorn getplaced:app --timeout 120
```

```
Flask-PyMongo==2.3.0  
pymongo==3.10.1  
bcrypt==3.1.7  
dnspython==1.16.0  
Flask==1.1.1  
Flask-Bcrypt==0.7.1  
Flask-Login==0.5.0  
gunicorn==20.0.4  
Jinja2==2.11.1  
python-http-client==3.2.7  
Werkzeug==1.0.1  
urllib3==1.25.8  
black==19.10b0
```

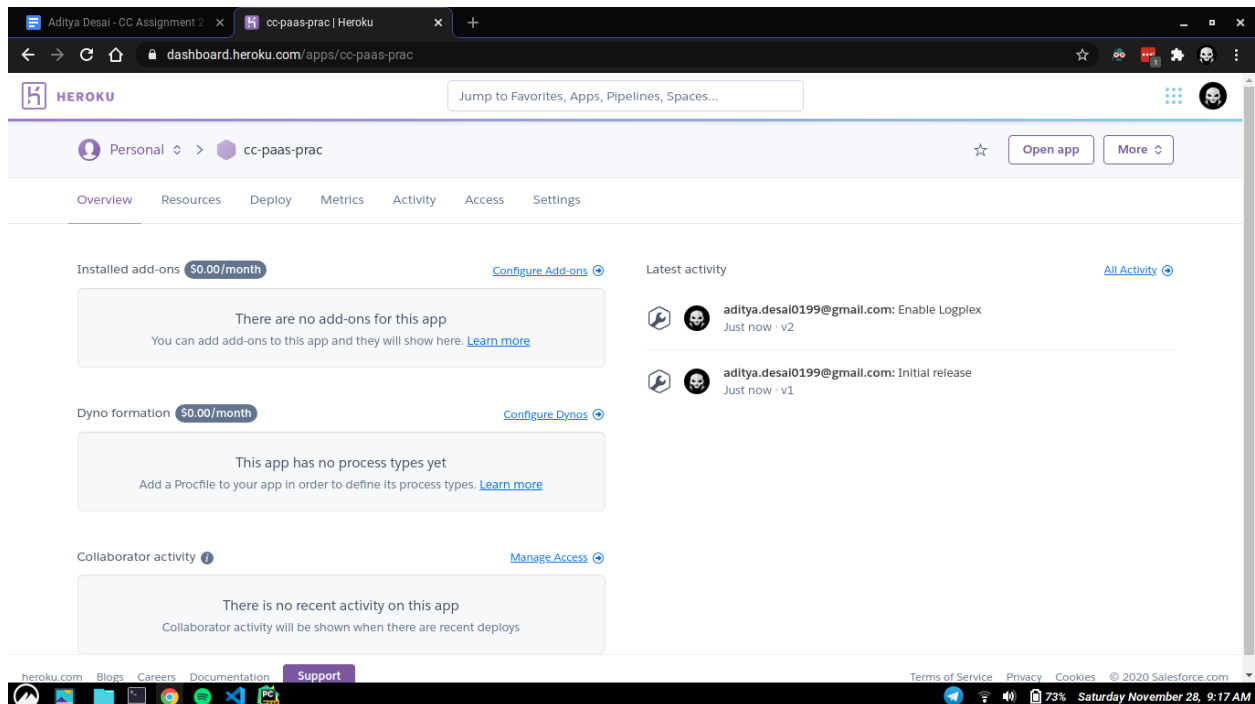
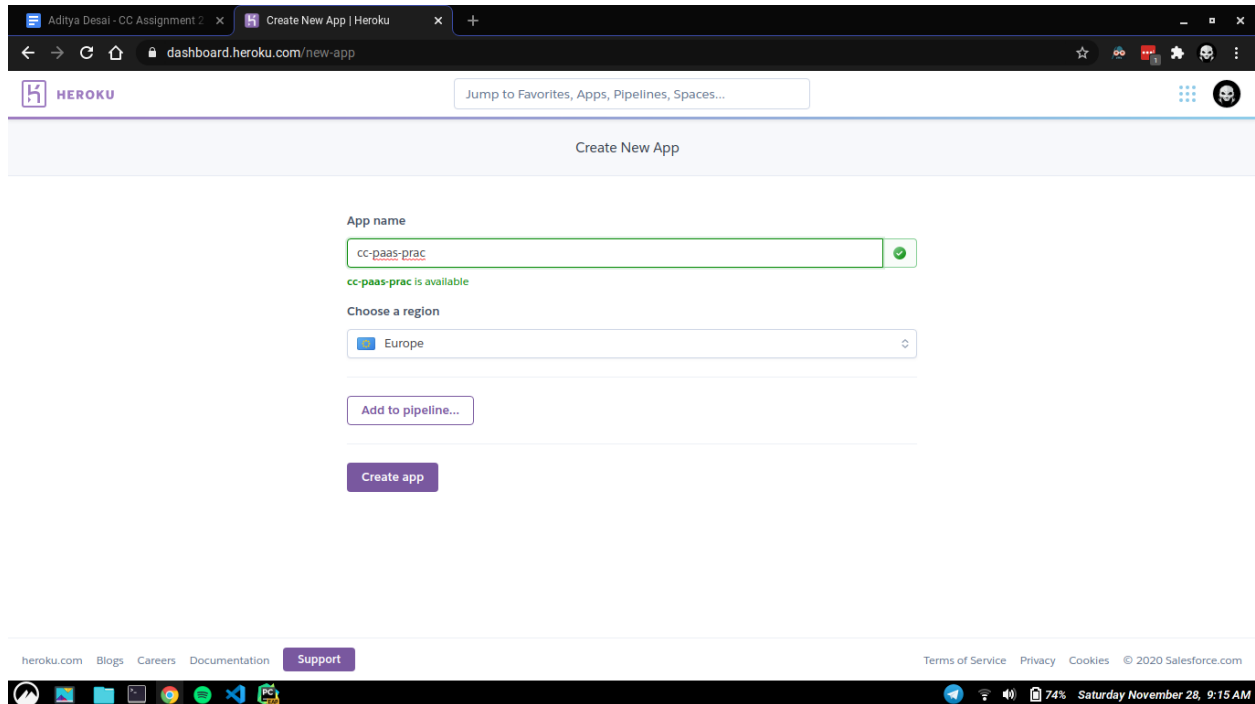
Running the project locally

```
> gunicorn getplaced:app --timeout 120
[2020-11-28 09:03:34 +0530] [7309] [INFO] Starting gunicorn 20.0.4
[2020-11-28 09:03:34 +0530] [7309] [INFO] Listening at: http://127.0.0.1:8000 (7309)
[2020-11-28 09:03:34 +0530] [7309] [INFO] Using worker: sync
[2020-11-28 09:03:34 +0530] [7311] [INFO] Booting worker with pid: 7311
```



TIME TO DEPLOY THIS TO HEROKU

Create new app on Heroku Dashboard



Login to Heroku CLI

```
> heroku login
heroku: Press any key to open up the browser to login or q to exit:
Opening browser to https://cli-auth.heroku.com/auth/cli/browser/1c2b9871-86f0-43051ZG4GAMa17wx2AQ.KEdpgQ_nmi06anRooEj6chDXZrcP45aU5zQDjpkMPyU
Logging in... done
Logged in as aditya.desai0199@gmail.com
```

Make sure you are in your Project directory

```
^ ~ ~/Projects/Student-Placement-Analysis master !2 .....
> _
```

Set remote for the app to push your project to, same as git remote

```
> heroku git:remote -a cc-paas-prac
Warning: heroku update available from 7.42.2 to 7.47.3.
set git remote heroku to https://git.heroku.com/cc-paas-prac.git
```

Push your project onto this remote to upload it to your app on heroku cloud

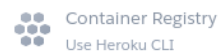
```
adityadesai@arch: ~/Projects/Student-Placement-Analysis
File Edit View Search Terminal Help
> git push heroku master
Enumerating objects: 236, done.
Counting objects: 100% (236/236), done.
Delta compression using up to 4 threads
Compressing objects: 100% (197/197), done.
Writing objects: 100% (236/236), 2.84 MiB | 417.00 KiB/s, done.
Total 236 (delta 117), reused 70 (delta 17), pack-reused 0
remote: Compressing source files... done.
remote: Building source:
remote:
remote: ----> Python app detected
remote: ! Python has released a security update! Please consider upgrading to python-3.8.6
remote: Learn More: https://devcenter.heroku.com/articles/python-runtimes
remote: ----> Installing python-3.8.2
remote: ----> Installing pip 20.1.1, setuptools 47.1.1 and wheel 0.34.2
remote: ----> Installing SQLite3
remote: ----> Installing requirements with pip
remote: Collecting Flask-PyMongo==2.3.0
remote: Downloading Flask_PyMongo-2.3.0-py2.py3-none-any.whl (12 kB)
remote: Collecting pymongo==3.10.1
remote: Downloading pymongo-3.10.1-cp38-cp38-manylinux2014_x86_64.whl (480 kB)
remote: Collecting bcrypt==3.1.7
remote: Downloading bcrypt-3.1.7-cp34-abi3-manylinux1_x86_64.whl (56 kB)
remote: Collecting dnspython==1.16.0
remote: Downloading dnspython-1.16.0-py2.py3-none-any.whl (188 kB)
remote: Collecting Flask==1.1.1
remote: Downloading Flask-1.1.1-py2.py3-none-any.whl (94 kB)
remote: Collecting Flask-Bcrypt==0.7.1
remote: Downloading Flask-Bcrypt-0.7.1.tar.gz (5.1 kB)
remote: Collecting Flask-Login==0.5.0
remote: Downloading Flask_Login-0.5.0-py2.py3-none-any.whl (16 kB)
remote: Collecting gunicorn==20.0.4
remote: Downloading gunicorn-20.0.4-py2.py3-none-any.whl (77 kB)
remote: Collecting Jinja2==2.11.1
remote: Downloading Jinja2-2.11.1-py2.py3-none-any.whl (126 kB)
remote: Collecting python-http-client==3.2.7
remote: Downloading python_http_client-3.2.7.tar.gz (7.0 kB)
```

Verify that your app has been deployed

```
remote: ----> Launching...
remote:      Released v3
remote:      https://cc-paas-prac.herokuapp.com/ deployed to Heroku
remote:
remote: Verifying deploy... done.
To https://git.heroku.com/cc-paas-prac.git
* [new branch]      master -> master
```

Alternatively you can choose to deploy your app from github itself by connecting your respective repository

Deployment method



Launch your app in the browser with <https://app-name.herokuapp.com>
Here <https://cc-paas-prac.herokuapp.com>

