

Week 4 Assignment - Week4: Deployment on Flask

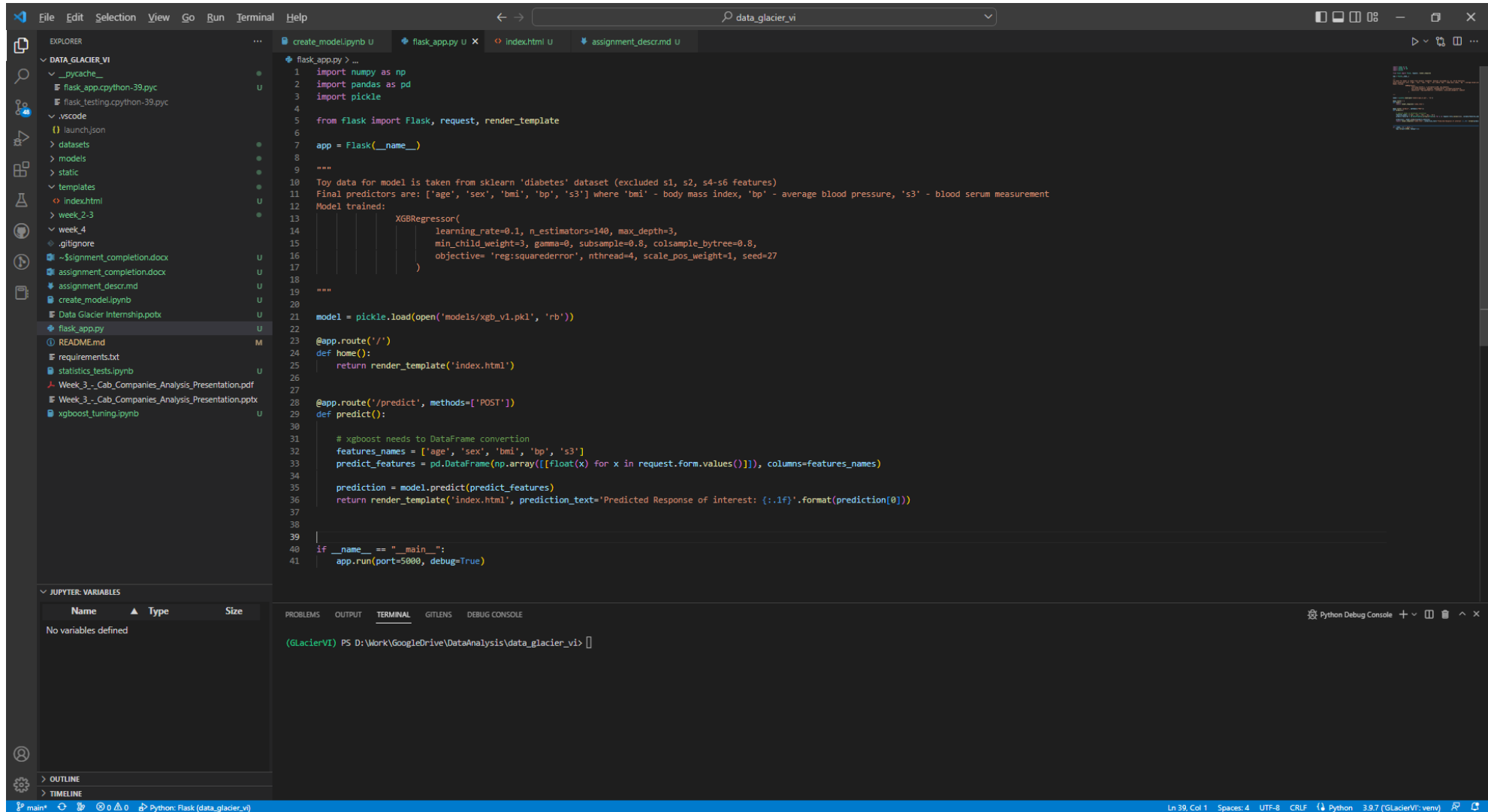
Name: Dmitry Sharukhin

Batch code: LISUM15

Submission date: Nov 23, 2022

Submitted to: Data Glacier

1. Code of Flask app



```
File Edit Selection View Go Run Terminal Help
data_glacier_vi

EXPLORER
DATA_GLACIER_VI
  __pycache__
  flask_app.cpython-39.pyc
  flask_testing.cpython-39.pyc
  .vscode
    launch.json
  datasets
  models
  static
  templates
    index.html
  week_2-3
  week_4
  .gitignore
  ~$assignment_completion.docx
  assignment_completion.docx
  assignment_descr.md
  create_model.ipynb
  Data Glacier Internship.pptx
  flask_app.py
  README.md
  requirements.txt
  statistics_tests.ipynb
  Week_3_-_Cab_Companies_Analysis_Presentation.pdf
  Week_3_-_Cab_Companies_Analysis_Presentation.pptx
  xgboost_tuning.ipynb

JUPYTER: VARIABLES
Name Type Size
No variables defined

TERMINAL
(GlacierVI) PS D:\Work\GoogleDrive\DataAnalysis\data_glacier_vi>
```

```
flask_app.py > ...
1 import numpy as np
2 import pandas as pd
3 import pickle
4
5 from flask import Flask, request, render_template
6
7 app = Flask(__name__)
8
9 """
10 Toy data for model is taken from sklearn 'diabetes' dataset (excluded s1, s2, s4-s6 features)
11 Final predictors are: ['age', 'sex', 'bmi', 'bp', 's3'] where 'bmi' - body mass index, 'bp' - average blood pressure, 's3' - blood serum measurement
12 Model trained:
13     XGBRegressor(
14         learning_rate=0.1, n_estimators=140, max_depth=3,
15         min_child_weight=3, gamma=0, subsample=0.8, colsample_bytree=0.8,
16         objective='reg:squarederror', nthread=4, scale_pos_weight=1, seed=27
17     )
18 """
19
20 model = pickle.load(open('models/xgb_v1.pkl', 'rb'))
21
22 @app.route('/')
23 def home():
24     return render_template('index.html')
25
26 @app.route('/predict', methods=['POST'])
27 def predict():
28     # xgboost needs to DataFrame conversion
29     features_names = ['age', 'sex', 'bmi', 'bp', 's3']
30     predict_features = pd.DataFrame([float(x) for x in request.form.values()], columns=features_names)
31
32     prediction = model.predict(predict_features)
33     return render_template('index.html', prediction_text='Predicted Response of interest: {:.1f}'.format(prediction[0]))
34
35 if __name__ == "__main__":
36     app.run(port=5000, debug=True)
```

2. Changed index.html (input fields edited)

The screenshot shows the VS Code editor interface with the following components:

- Explorer:** Displays the project structure. The `templates` directory is expanded, showing `index.html` as the active file.
- Editor:** Displays the content of `index.html`. The code includes a DOCTYPE declaration, a head section with meta tags and font links, and a body section containing a login form.
- Jupyter Variables:** A panel at the bottom left showing no variables defined.
- Terminal:** A panel at the bottom right showing the command prompt output: `(GlacierVI) PS D:\Work\GoogleDrive\DataAnalysis\data_glacier_vi>`

The HTML code in `index.html` is as follows:

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>ML API</title>
6   <link href="https://fonts.googleapis.com/css?family=Pacifico" rel="stylesheet" type="text/css">
7   <link href="https://fonts.googleapis.com/css?family=Arimo" rel="stylesheet" type="text/css">
8   <link href="https://fonts.googleapis.com/css?family=Hind:300" rel="stylesheet" type="text/css">
9   <link href="https://fonts.googleapis.com/css?family=Open+Sans+Condensed:300" rel="stylesheet" type="text/css">
10  <link rel="stylesheet" href="{{ url_for('static', filename='css/style.css') }}">
11 </head>
12
13 <body>
14   <div class="login">
15     <h1>Predicting Response of interest</h1>
16     <!-- Main Input For Receiving Query to our ML -->
17     <!-- predictors are: ['age', 'sex', 'bmi', 'bp', 's3'] -->
18     <form action="{{ url_for('predict') }}" method="post">
19       <input type="text" name="age" placeholder="Age" required="required" />
20       <input type="text" name="sex" placeholder="Sex (1 - male, 2 - female)" required="required" />
21       <input type="text" name="bmi" placeholder="Body Mass Index (BMI)" required="required" />
22       <input type="text" name="bp" placeholder="Average Blood Pressure" required="required" />
23       <input type="text" name="s3" placeholder="Blood Serum Measurement" required="required" />
24       <button type="submit" class="btn btn-primary btn-block btn-large">Predict</button>
25     </form>
26     <br>
27     <br>
28     {{ prediction_text }}
29   </div>
30   
31 </body>
32 </html>
```

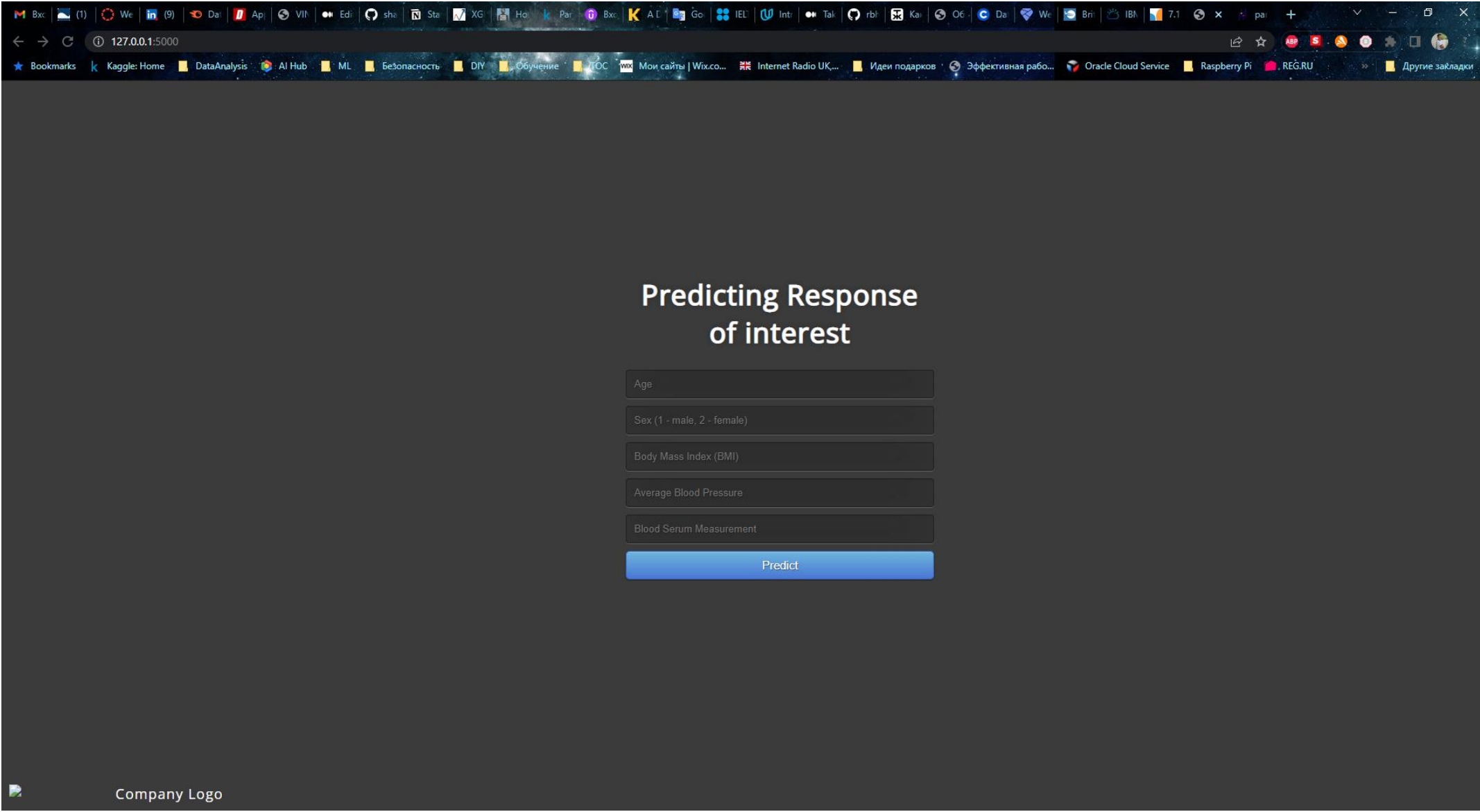
3. Running Flask in debug mode in VisualStudioCode

```
1 import numpy as np
2 import pandas as pd
3 import pickle
4
5 from flask import Flask, request, render_template
6
7 app = Flask(__name__)
8
9 """
10 Toy data for model is taken from sklearn 'diabetes' dataset (excluded s1, s2, s4-s6 features)
11 Final predictors are: ['age', 'sex', 'bmi', 'bp', 's3'] where 'bmi' - body mass index, 'bp' - average blood pressure, 's3' - blood serum measurement
12 Model trained:
13
14         XGBRegressor(
15             learning_rate=0.1, n_estimators=140, max_depth=3,
16             min_child_weight=3, gamma=0, subsample=0.8, colsample_bytree=0.8,
17             objective= 'reg:squarederror', nthread=4, scale_pos_weight=1, seed=27
18         )
19 """
20
21 model = pickle.load(open('models/xgb_v1.pkl', 'rb'))
22
23 @app.route('/')
24 def home():
25     return render_template('index.html')
26
27
28 @app.route('/predict', methods=['POST'])
29 def predict():
30
31     # xgboost needs to DataFrame convention
32     features_names = ['age', 'sex', 'bmi', 'bp', 's3']
33     predict_features = pd.DataFrame(np.array([[float(x) for x in request.form.values()]]), columns=features_names)
34
35     prediction = model.predict(predict_features)
36     return render_template('index.html', prediction_text='Predicted Response of interest: {:.1f}'.format(prediction[0]))
37
38
39
40 if __name__ == "__main__":
41     app.run(port=5000, debug=True)
```

PROBLEMS OUTPUT TERMINAL GITLENS DEBUG CONSOLE

```
(GlacierVI) PS D:\Work\GoogleDrive\DataAnalysis\data_glacier_vi> d; cd 'd:\Work\GoogleDrive\DataAnalysis\data_glacier_vi'; & 'd:\Work\venv\GLacierVI\Scripts\python.exe' 'c:\Users\Дмитрий\.vscode\extensions\ms-python.python-2022.18.2\pythonFiles\lib\pytho
n\debugpy\adapter\..\..\debugpy\launcher' '56341' '-.' '-m' 'flask' 'run' '--no-debugger' '--no-reload'
* Serving Flask app 'flask_app.py'
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:5000
Press CTRL+C to quit
[]
```

4. Running app homepage



5. Filling test data

Predicting Response of interest

35

1

25

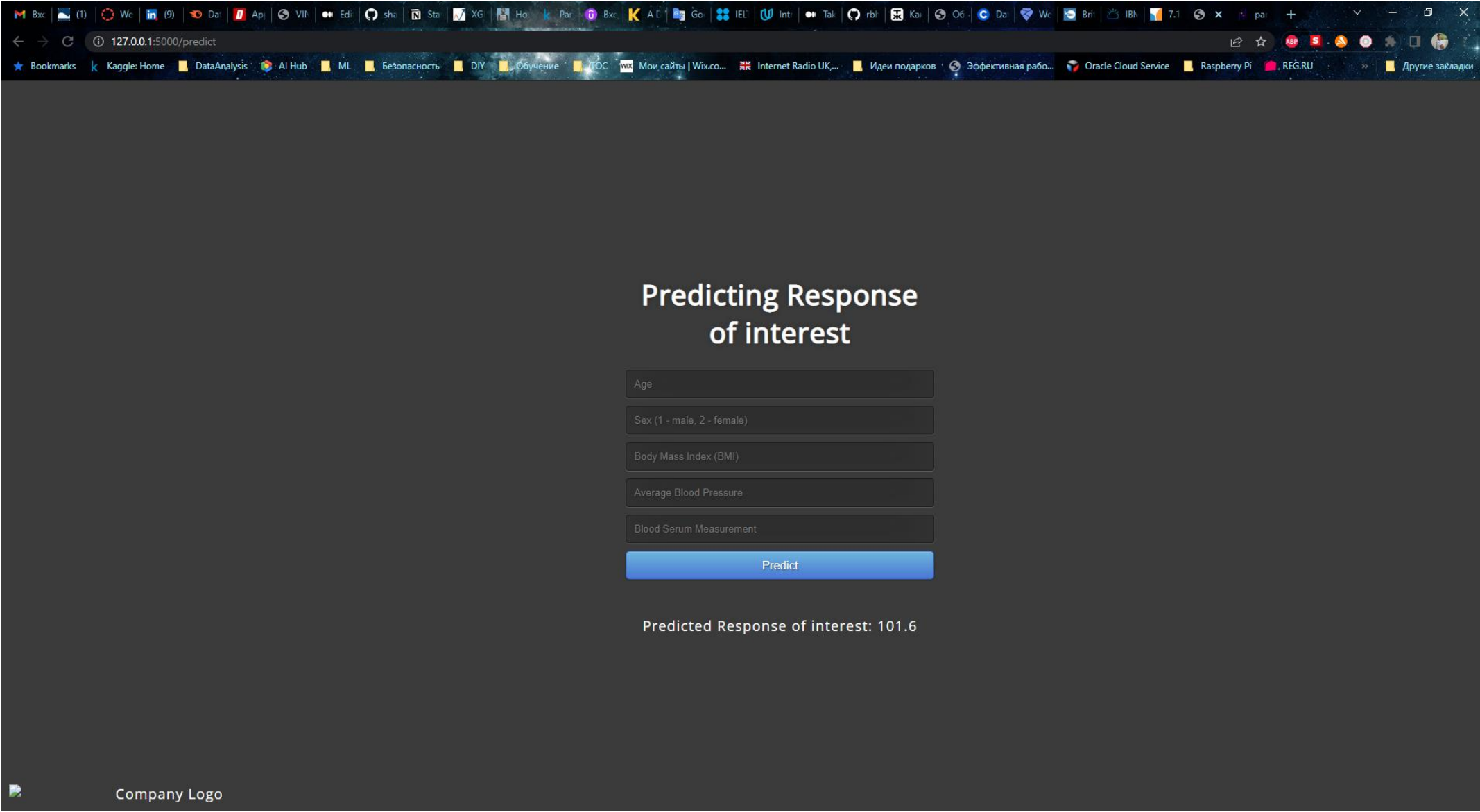
100

95

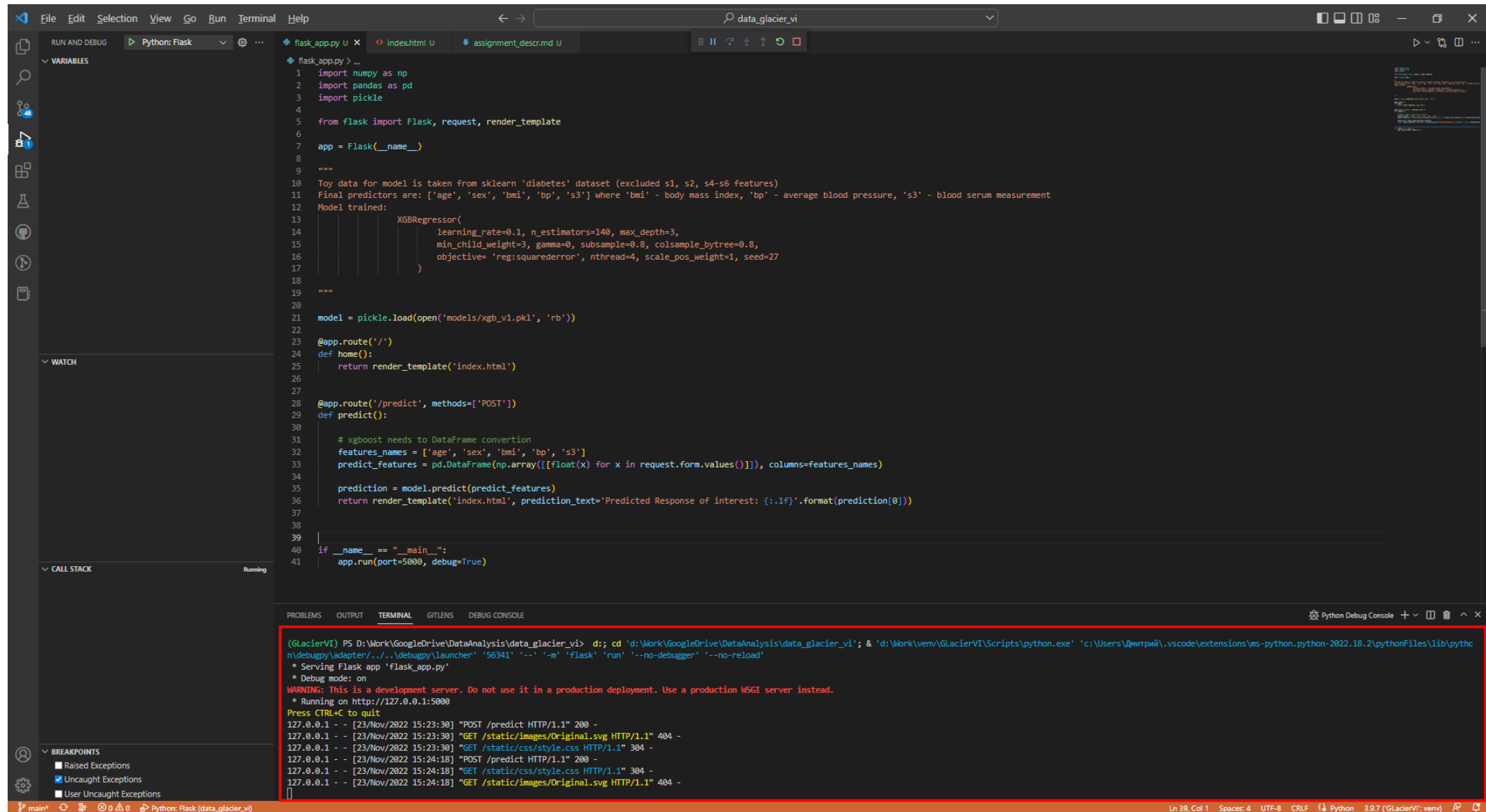
Predict

Company Logo

6. Prediction result shown on web page



7. Flask output in terminal window



The image shows a Visual Studio Code editor window with a Python Flask application open. The application code is in `flask_app.py` and includes imports for `numpy`, `pandas`, and `pickle`, along with `Flask` and `request` from the `flask` module. It defines a `home` route and a `predict` route that uses an `XGBRegressor` model to predict based on input features. The terminal window at the bottom shows the command to run the application and its output, including a warning about using a development server and a list of HTTP requests and responses.

```
1 import numpy as np
2 import pandas as pd
3 import pickle
4
5 from flask import Flask, request, render_template
6
7 app = Flask(__name__)
8
9 """
10 Toy data for model is taken from sklearn 'diabetes' dataset (excluded s1, s2, s4-s6 features)
11 Final predictors are: ['age', 'sex', 'bmi', 'bp', 's3'] where 'bmi' - body mass index, 'bp' - average blood pressure, 's3' - blood serum measurement
12 Model trained:
13
14         XGBRegressor(
15             learning_rate=0.1, n_estimators=140, max_depth=3,
16             min_child_weight=3, gamma=0, subsample=0.8, colsample_bytree=0.8,
17             objective='reg:squarederror', nthread=4, scale_pos_weight=1, seed=27
18         )
19
20 """
21 model = pickle.load(open('models/xgb_v1.pkl', 'rb'))
22
23 @app.route('/')
24 def home():
25     return render_template('index.html')
26
27
28 @app.route('/predict', methods=['POST'])
29 def predict():
30
31     # xgboost needs to DataFrame conversion
32     features_names = ['age', 'sex', 'bmi', 'bp', 's3']
33     predict_features = pd.DataFrame(np.array([[float(x) for x in request.form.values()]]), columns=features_names)
34
35     prediction = model.predict(predict_features)
36     return render_template('index.html', prediction_text='Predicted Response of interest: {:.1f}'.format(prediction[0]))
37
38
39
40 if __name__ == "__main__":
41     app.run(port=5000, debug=True)
```

Terminal Output:

```
(GlacierVI) PS D:\Work\GoogleDrive\DataAnalysis\data_glacier_vi> d:; cd 'd:\Work\GoogleDrive\DataAnalysis\data_glacier_vi'; & 'd:\Work\venv\GlacierVI\Scripts\python.exe' 'c:\Users\Дмитрий\.vscode\extensions\ms-python.python-2022.18.2\pythonFiles\lib\pytho
n\debugpy\adapter\..\..\debugpy\launcher' '56341' '--' '-m' 'flask' 'run' '--no-debugger' '--no-reload'
* Serving Flask app 'flask_app.py'
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:5000
Press CTRL+C to quit
127.0.0.1 - - [23/Nov/2022 15:23:30] "POST /predict HTTP/1.1" 200 -
127.0.0.1 - - [23/Nov/2022 15:23:30] "GET /static/images/Original.svg HTTP/1.1" 404 -
127.0.0.1 - - [23/Nov/2022 15:23:30] "GET /static/css/style.css HTTP/1.1" 304 -
127.0.0.1 - - [23/Nov/2022 15:24:18] "POST /predict HTTP/1.1" 200 -
127.0.0.1 - - [23/Nov/2022 15:24:18] "GET /static/css/style.css HTTP/1.1" 304 -
127.0.0.1 - - [23/Nov/2022 15:24:18] "GET /static/images/Original.svg HTTP/1.1" 404 -
```