

Kelompok 4

ASSIGNMENT MODEL DEPLOYMENT

2702352616 - Jovita Putri Aulia

2702363254 - Keyla Faritsha Rindani

2702322720 - Nabila Azwad Ambbiya

2702244284 - Renata Aqila Ridha Putri

2702264576 - Stephanie Nadya

Model
Deployment

Extrovert vs. Introvert Behavior Data

	Time_spent_Alone	Stage_fear	Social_event_attendance	Going_outside	Drained_after_socializing	Friends_circle_size	Post_frequency	Personality
0	4.0	No	4.0	6.0	No	13.0	5.0	Extrovert
1	9.0	Yes	0.0	0.0	Yes	0.0	3.0	Introvert
2	9.0	Yes	1.0	2.0	Yes	5.0	2.0	Introvert
3	0.0	No	6.0	7.0	No	14.0	8.0	Extrovert
4	3.0	No	9.0	4.0	No	8.0	5.0	Extrovert

```
df.shape
```

```
(2900, 8)
```

Dataset ini berisi 2.900 data dengan 8 fitur yang berkaitan dengan perilaku sosial dan ciri kepribadian, yang dirancang untuk mengeksplorasi dan mengklasifikasikan individu sebagai ekstrovert atau introvert.

FEATURES

No	Feature	Description
1	Time_spent_Alone	Hours spent alone daily (0–11).
2	Stage_fear	Presence of stage fright (Yes/No).
3	Social_event_attendance	Frequency of social events (0–10).
4	Going_outside	The occupation or profession of the person.

FEATURES

No	Feature	Description
5	Drained_after_socializing	Feeling drained after socializing (Yes/No).
6	Friends_circle_size	Number of close friends (0–15).
7	Post_frequency	Social media post frequency (0–10).
8	Personality	Extrovert or Introvert (target variable).

KONDISI DATA

df.info()

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2900 entries, 0 to 2899
Data columns (total 8 columns):
#   Column                                Non-Null Count  Dtype
---  -
0   Time_spent_Alone                      2837 non-null   float64
1   Stage_fear                           2827 non-null   object
2   Social_event_attendance              2838 non-null   float64
3   Going_outside                       2834 non-null   float64
4   Drained_after_socializing            2848 non-null   object
5   Friends_circle_size                 2823 non-null   float64
6   Post_frequency                      2835 non-null   float64
7   Personality                          2900 non-null   object
dtypes: float64(5), object(3)
memory usage: 181.4+ KB
```

Dari info ini bisa disimpulkan beberapa informasi:

- Data type pada tiap kolom sudah tepat.
- Ada missing values pada tiap kolom dataset kecuali kolom ‘Personality’.

Berikut rincian jumlah missing value per kolom:

Time_spent_Alone	63
Stage_fear	73
Social_event_attendance	62
Going_outside	66
Drained_after_socializing	52
Friends_circle_size	77
Post_frequency	65
Personality	0

IMPUTASI MISSING VALUES KOLOM NUMERIK

	Time_spent_Alone	Social_event_attendance	Going_outside	Friends_circle_size	Post_frequency
count	2837.000000	2838.000000	2834.000000	2823.000000	2835.000000
mean	4.505816	3.963354	3.000000	6.268863	3.564727
std	3.479192	2.903827	2.247327	4.289693	2.926582
min	0.000000	0.000000	0.000000	0.000000	0.000000
25%	2.000000	2.000000	1.000000	3.000000	1.000000
50%	4.000000	3.000000	3.000000	5.000000	3.000000
75%	8.000000	6.000000	5.000000	10.000000	6.000000
max	11.000000	10.000000	7.000000	15.000000	10.000000

1. Time_spent_Alone

Mean > Median
Imputasi dengan **median**

2. Social_event_attendance

Mean > Median
Imputasi dengan **median**

3. Going_outside

Mean = Median
Imputasi dengan **mean**

4. Friends_circle_size

Mean > Median
Imputasi dengan **median**

5. Post_frequency

Mean > Median
Imputasi dengan **median**

IMPUTASI MISSING VALUES KOLOM KATEGORIKAL

```
Stage_fear
No      1417
Yes     1410
Name: count, dtype: int64

Drained_after_socializing
No      1441
Yes     1407
Name: count, dtype: int64
```

Kolom 'Stage_fear' dan 'Drained_after_socializing' ini diimputasi dengan nilai modus masing-masing kolom.

DATA SETELAH IMPUTASI

	Time_spent_Alone	Social_event_attendance	Going_outside	Friends_circle_size	Post_frequency
count	2900.000000	2900.000000	2900.000000	2900.000000	2900.000000
mean	4.494828	3.942759	3.000000	6.235172	3.552069
std	3.441971	2.875987	2.221597	4.237255	2.894794
min	0.000000	0.000000	0.000000	0.000000	0.000000
25%	2.000000	2.000000	1.000000	3.000000	1.000000
50%	4.000000	3.000000	3.000000	5.000000	3.000000
75%	7.000000	6.000000	5.000000	10.000000	6.000000
max	11.000000	10.000000	7.000000	15.000000	10.000000

Hasil describe setelah imputasi menunjukkan:

- Semua kolom sudah tidak ada missing values (count = 2900)
- Distribusi tidak berubah ekstrem setelah imputasi
- Tidak muncul nilai outlier baru
- Mean dan median tetap konsisten dengan kondisi sebelum imputasi

ENCODING

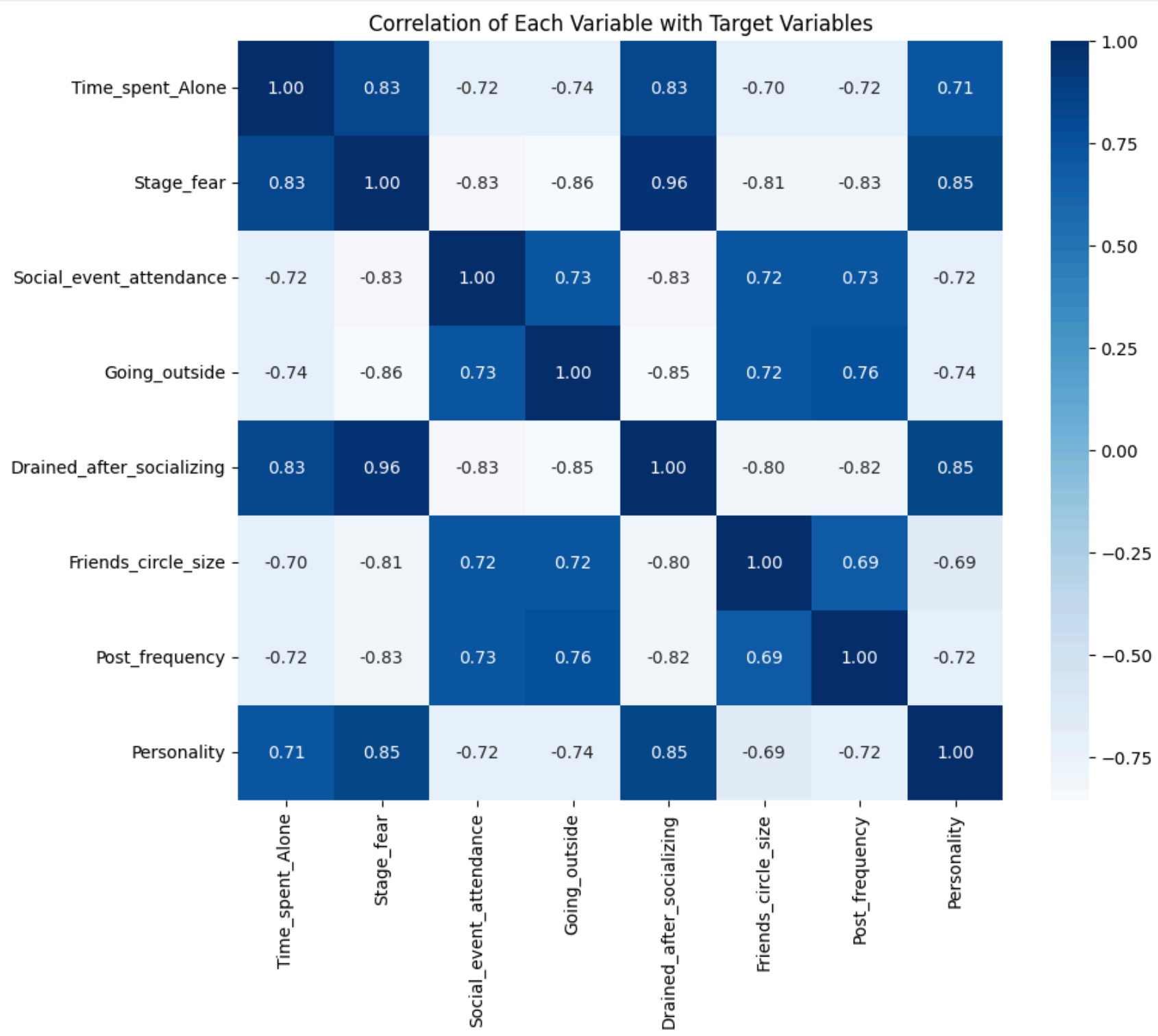
Terdapat 3 kolom yang perlu diencode dengan binary maupun label encoding, berikut rinciannya:

- Stage_fear dengan binary encoding
- Drained_after_socializing dengan binary encoding
- Personality dengan label encoding

Pemetaannya adalah No menjadi 0 dan Yes menjadi 1 untuk binary encoding. Sedangkan Extrovert menjadi 0 dan Introvert menjadi 1 untuk label encoding.

```
binary = {"Stage_fear": {"No": 0, "Yes" : 1},  
          "Drained_after_socializing": {"No" : 0, "Yes" : 1}}  
df = df.replace(binary)  
  
label = {"Personality": {"Extrovert": 0, "Introvert": 1}}  
df = df.replace(label)
```

HEATMAP



Heatmap ini menunjukkan korelasi antar variabel. Warna biru tua menandakan korelasi yang kuat, sedangkan biru muda menandakan korelasi lemah.

Beberapa poin penting:

- Time_spent_Alone berkorelasi positif dengan Stage_fear dan Drained_after_socializing.
- Stage_fear sangat berkorelasi dengan Drained_after_socializing (0.96).
- Personality berkorelasi negatif dengan variabel seperti Stage_fear dan Time_spent_Alone, menandakan kecenderungan ekstrover.

Heatmap ini membantu memahami hubungan antar fitur sebelum proses modeling

DATA SPLITTING

Pertama, kolom 'Personality' di set menjadi target variabel.

```
input_df=df.drop('Personality',axis=1)
output_df=df['Personality']
```

Lalu, data dipisah menjadi 80% training dan 20% testing sehingga pembagiannya:

```
x_train, x_test, y_train, y_test = train_test_split(input_df, output_df, test_size = 0.2, random_state = 0)
```

```
print("shape X_train: ",x_train.shape)
print("shape X_test: ",x_test.shape)
print("shape y_train: ",y_train.shape)
print("shape y_test: ",y_test.shape)
```

```
shape X_train:  (2320, 7)
shape X_test:   (580, 7)
shape y_train:  (2320,)
shape y_test:   (580,)
```

MODEL - RANDOM FOREST

Classification Report					
	precision	recall	f1-score	support	
0	0.89	0.92	0.91	270	
1	0.93	0.90	0.92	310	
accuracy			0.91	580	
macro avg	0.91	0.91	0.91	580	
weighted avg	0.91	0.91	0.91	580	

Pada dataset ini, model Random Forest mampu mengklasifikasikan individu sebagai ekstrovert atau introvert dengan akurasi sebesar 91%.

Model ini menunjukkan keseimbangan performa pada kedua kelas, dengan nilai precision, recall, dan f1-score di atas 0.90. Hal ini menunjukkan bahwa model dapat mengenali pola perilaku dengan baik, tanpa cenderung bias terhadap salah satu kelas.

RANDOM FOREST (HYPERPARAMETER TUNING)

```
param_dist = {  
    'n_estimators': [100, 200, 300, 400, 500],  
    'max_depth': [None, 5, 10, 15, 20],  
    'min_samples_split': [2, 5, 10],  
    'min_samples_leaf': [1, 2, 4],  
    'max_features': ['auto', 'sqrt', 'log2']  
}
```

Hyperparameter tuning dilakukan menggunakan RandomizedSearchCV, yaitu metode untuk mencari kombinasi parameter terbaik secara acak. Berikut arti masing-masing parameter yang diuji:

- n_estimators: Jumlah pohon dalam model.
- max_depth: Batas kedalaman pohon.
- min_samples_split: Minimum data untuk membagi node.
- min_samples_leaf: Minimum data pada daun pohon.
- max_features: Jumlah fitur yang dipakai saat split.

Setelah dilakukan hyperparameter tuning menggunakan RandomizedSearchCV model mengalami peningkatan nilai akurasi menjadi 93%

Ini menunjukkan bahwa pemilihan parameter yang tepat berdampak signifikan terhadap performa model.

	precision	recall	f1-score	support
0	0.91	0.93	0.92	270
1	0.94	0.92	0.93	310
accuracy			0.93	580
macro avg	0.93	0.93	0.93	580
weighted avg	0.93	0.93	0.93	580

MODEL - XGBOOST

Classification Report

	precision	recall	f1-score	support
0	0.90	0.92	0.91	270
1	0.93	0.91	0.92	310
accuracy			0.91	580
macro avg	0.91	0.91	0.91	580
weighted avg	0.91	0.91	0.91	580

Pada dataset ini, model XGBoost memiliki nilai akurasi sebesar 91% sama seperti Random Forest

XGBoost membangun model secara bertahap dengan memanfaatkan kesalahan dari iterasi sebelumnya, sehingga mampu menghasilkan prediksi yang lebih akurat dan stabil.

XGBOOST (HYPERPARAMETER TUNING)

```
param_grid = {
    'n_estimators': [50, 100, 200],
    'max_depth': [3, 5, 7],
    'learning_rate': [0.01, 0.1, 0.2],
    'min_child_weight': [1, 2]
}
```

Setelah dilakukan hyperparameter tuning menggunakan GridSearchCV model mengalami peningkatan nilai akurasi menjadi 93%

menunjukkan bahwa pemilihan parameter yang tepat berdampak signifikan terhadap performa model.

- `n_estimators`: jumlah pohon yang digunakan dalam boosting.
- `max_depth`: kedalaman maksimum tiap pohon (mencegah overfitting).
- `learning_rate`: seberapa besar kontribusi setiap pohon baru (semakin kecil, training lebih lambat tapi akurat).
- `min_child_weight`: jumlah minimum sampel yang dibutuhkan untuk membuat node baru (mengontrol kompleksitas model).

Classification Report				
	precision	recall	f1-score	support
0	0.91	0.93	0.92	270
1	0.94	0.92	0.93	310
accuracy			0.93	580
macro avg	0.93	0.93	0.93	580
weighted avg	0.93	0.93	0.93	580

Cara Menjalankan FastAPI

1. Buat object base model

```
PreProcessing.ipynb  personality.py ×  fast_api.py  TC New Request  streamlit_app.py

personality.py > ...
1  from pydantic import BaseModel
2
3  class PersonalityBase(BaseModel):
4      Time_spent_Alone: float
5      Stage_fear: int
6      Social_event_attendance: float
7      Going_outside: float
8      Drained_after_socializing: int
9      Friends_circle_size: float
10     Post_frequency: float
11
```


2. Buat fast_api.py untuk bagian backend

```
PreProcessing.ipynb  personality.py  fast_api.py X  TC New Request  streamlit_app.py

fast_api.py > ...
1  import uvicorn
2  from fastapi import FastAPI
3  from personality import PersonalityBase
4  import pandas as pd
5  import joblib
6
7  # Load trained pipeline model
8  model = joblib.load("trained_model.pkl")
9
10 # Define Label mapping
11 label_map = {0: "Extrovert", 1: "Introvert"}
12
13 # Initialize app
14 app = FastAPI()
15
16 @app.get("/")
17 def read_root():
18     return {"message": "Personality Prediction API is running"}
19
20 @app.post("/predict")
21 def predict(data: PersonalityBase):
22     # Convert input to DataFrame
23     input_df = pd.DataFrame([data.dict()])
24
25     # Predict
26     prediction = int(model.predict(input_df)[0])
27     label = label_map.get(prediction, "Unknown")
28
29     return {
30         "prediction": prediction,
31         "label": label
32     }
33
34 if __name__ == '__main__':
35     uvicorn.run(app, host='127.0.0.1', port=7860)
```

3. Pastikan sudah membuat virtual environment terlebih dahulu. Lalu open cmd dan activate si environmentnya

```
C:\Users\Renata Aqila>D:
```

```
D:\>cd D:\Sekolah\binus\SEM 4\Model Deployment\Group4_New
```

```
D:\Sekolah\binus\SEM 4\Model Deployment\Group4_New>.\env\Scripts\activate.bat
```

4. Lalu install beberapa library nya seperti uvicorn, fastapi, numpy, pandas, dan scikit-learn

```
C:\WINDOWS\system32\cmd. X + v

D:\Sekolah\binus\SEM 4\Model Deployment\Group4_New>.\env\Scripts\activate.bat

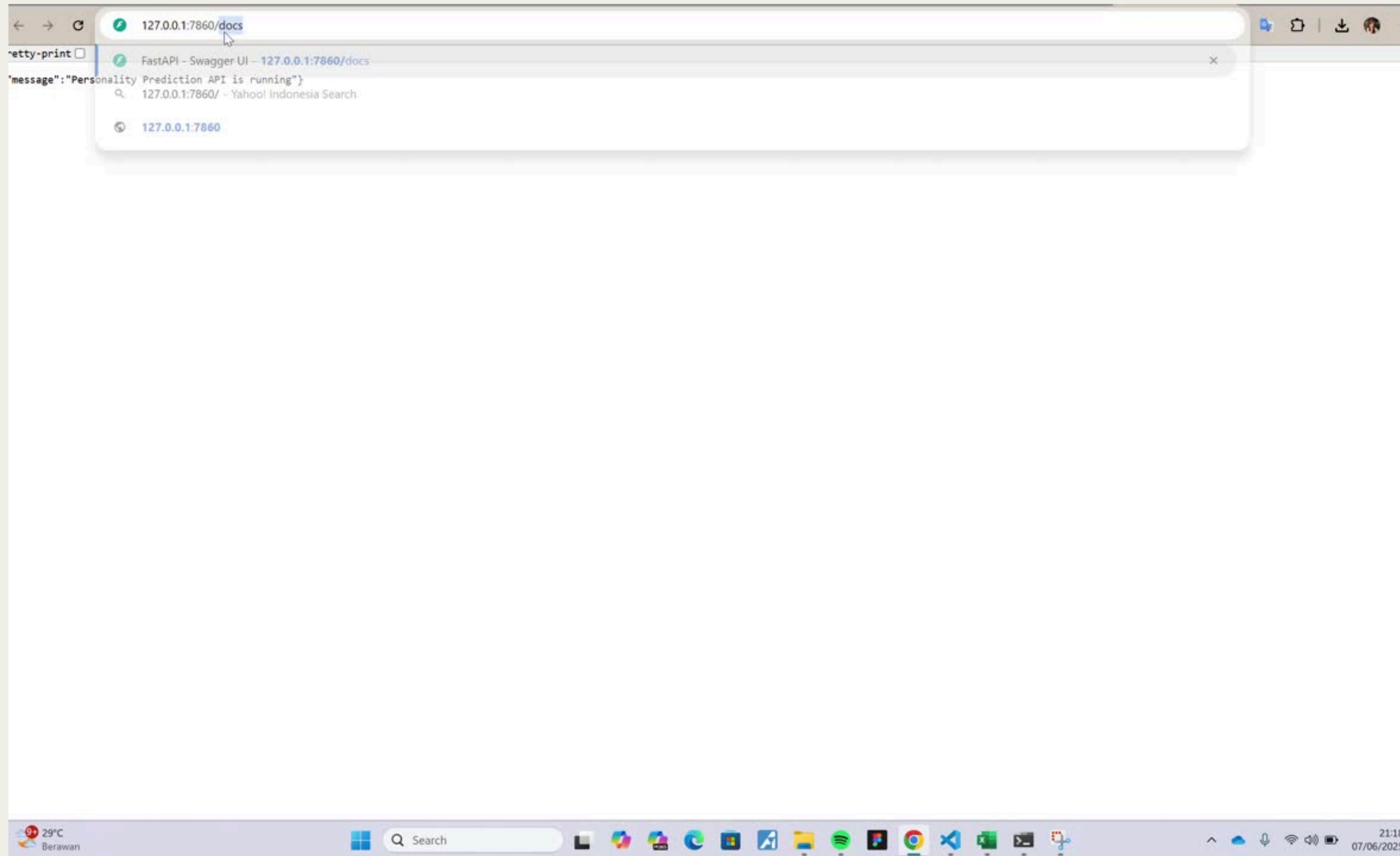
(env) D:\Sekolah\binus\SEM 4\Model Deployment\Group4_New>pip install uvicorn
Collecting uvicorn
  Using cached https://files.pythonhosted.org/packages/ad/bd/d47ee02312640fcf26c7e1c807402d5c5eab468571153a94ec8f7ada0e46/uvicorn-0.22.0-py3-none-any.whl
Collecting click>=7.0 (from uvicorn)
  Using cached https://files.pythonhosted.org/packages/7e/d4/7ebdbd03970677812aac39c869717059dbb71a4cfc033ca6e5221787892c/click-8.1.8-py3-none-any.whl
Collecting typing-extensions; python_version < "3.8" (from uvicorn)
  Using cached https://files.pythonhosted.org/packages/ec/6b/63cc3df74987c36fe26157ee12e09e8f9db4de771e0f3404263117e75b95/typing_extensions-4.7.1-py3-none-any.whl
Collecting h11>=0.8 (from uvicorn)
  Using cached https://files.pythonhosted.org/packages/95/04/ff642e65ad6b90db43e668d70fffb6736436c7ce41fcc549f4e9472234127/h11-0.14.0-py3-none-any.whl
Collecting importlib-metadata; python_version < "3.8" (from click>=7.0->uvicorn)
  Using cached https://files.pythonhosted.org/packages/ff/94/64287b38c7de4c90683630338cf28f129decbbba0a44f0c6db35a873c73c4/importlib_metadata-6.7.0-py3-none-any.whl
Collecting colorama; platform_system == "Windows" (from click>=7.0->uvicorn)
  Using cached https://files.pythonhosted.org/packages/d1/d6/3965ed04c63042e047cb6a3e6ed1a63a35087b6a609aa3a15ed8ac56c221/colorama-0.4.6-py2.py3-none-any.whl
Collecting zipp>=0.5 (from importlib-metadata; python_version < "3.8"->click>=7.0->uvicorn)
  Using cached https://files.pythonhosted.org/packages/5b/fa/c9e82bbe1af6266adf08afb563905eb87cab83fde00a0a08963510621047/zipp-3.15.0-py3-none-any.whl
Installing collected packages: typing-extensions, zipp, importlib-metadata, colorama, click, h11, uvicorn
Successfully installed click-8.1.8 colorama-0.4.6 h11-0.14.0 importlib-metadata-6.7.0 typing-extensions-4.7.1 uvicorn-0.22.0 zipp-3.15.0
WARNING: You are using pip version 19.2.3, however version 24.0 is available.
You should consider upgrading via the 'python -m pip install --upgrade pip' command.

(env) D:\Sekolah\binus\SEM 4\Model Deployment\Group4_New>pip install fastapi
```

5. Setelah itu, ketik 'uvicorn fast_api:app --reload' di cmd, dan nanti akan muncul link yang redirect ke fastAPI nya

```
(env) D:\Sekolah\binus\SEM 4\Model Deployment\Group4_New>uvicorn fast_api:app --reload --port 7860
INFO: Will watch for changes in these directories: ['D:\\Sekolah\\binus\\SEM 4\\Model Deployment\\Group4_New']
INFO: Uvicorn running on http://127.0.0.1:7860 (Press CTRL+C to quit)
INFO: Started reloader process [12656] using StatReload
INFO: Started server process [25868]
INFO: Waiting for application startup.
INFO: Application startup complete.
INFO: 127.0.0.1:54374 - "GET / HTTP/1.1" 200 OK
INFO: 127.0.0.1:54375 - "GET /docs HTTP/1.1" 200 OK
INFO: 127.0.0.1:54375 - "GET /openapi.json HTTP/1.1" 200 OK
INFO: 127.0.0.1:54378 - "POST /predict HTTP/1.1" 200 OK
INFO: 127.0.0.1:54378 - "POST /predict HTTP/1.1" 200 OK
WARNING: StatReload detected changes in 'streamlit_app.py'. Reloading...
INFO: Shutting down
INFO: Waiting for application shutdown.
INFO: Application shutdown complete.
INFO: Finished server process [25868]
INFO: Started server process [10016]
INFO: Waiting for application startup.
INFO: Application startup complete.
```


6. Tinggal di click linknya, lalu tambahkan '/docs' di website



Testing

Di ipynb, kita menggunakan rumus codingan ini untuk mengambil salah satu data dari csv kita.
Misalnya kita mengambil data di baris pertama

```
sample = df.iloc[0].to_dict()  
print(sample)
```

✓ 0.0s

```
{'Time_spent_Alone': 4.0, 'Stage_fear': 0.0, 'Social_event_attendance':  
4.0, 'Going_outside': 6.0, 'Drained_after_socializing': 0.0,  
'Friends_circle_size': 13.0, 'Post_frequency': 5.0, 'Personality': 0.0}
```

Testing

Setelah itu, kita input ke dalam thunder clientnya

JSON Content

```
1  {
2    "Time_spent_Alone": 4.0,
3    "Stage_fear": 0,
4    "Social_event_attendance": 4.0,
5    "Going_outside": 6.0,
6    "Drained_after_socializing": 0,
7    "Friends_circle_size": 13.0,
8    "Post_frequency": 5.0
9  }
10 |
```

Testing

Seperti yang di file preprocessing, kita udah encode beberapa variable seperti Stage_fear, Drained_After_socializing dan Personality, jadi ketika input di thunder client ini, sudah menggunakan angka semua

JSON Content

```
1  {
2    "Time_spent_Alone": 4.0,
3    "Stage_fear": 0,
4    "Social_event_attendance": 4.0,
5    "Going_outside": 6.0,
6    "Drained_after_socializing": 0,
7    "Friends_circle_size": 13.0,
8    "Post_frequency": 5.0
9  }
10 |
```

```
binary = {"Stage_fear": {"No": 0, "Yes" : 1},
          "Drained_after_socializing": {"No" : 0, "Yes" : 1}}
df = df.replace(binary)

label = {"Personality": {"Extrovert": 0, "Introvert": 1}}
df = df.replace(label)
```


File Edit Selection View Go Run Terminal Help

Group4_New

EXPLORER

- GROUP4_NEW
 - __pycache__
 - env
 - fast_api.py
 - personality_dataset.csv
 - personality.py
 - PreProcessing.ipynb
 - requirements.txt
 - streamlit_app.py
 - trained_model.pkl

PreProcessing.ipynb personality.py fast_api.py New Request X streamlit_app.py

POST http://127.0.0.1:7860/predict Send

Query Headers² Auth Body¹ Tests Pre Run

JSON XML Text Form Form-encode GraphQL Binary

JSON Content

```
1 {
2   "Time_spent_Alone": 4.0,
3   "Stage_fear": 0,
4   "Social_event_attendance": 4.0,
5   "Going_outside": 6.0,
6   "Drained_after_socializing": 0,
7   "Friends_circle_size": 13.0,
8   "Post_frequency": 5.0
9 }
10
```

Status: 200 OK Size: 36 Bytes Time: 32 ms

Response Headers⁵ Cookies Results Docs

```
1 {
2   "prediction": 0,
3   "label": "Extrovert"
4 }
```

Copy

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS JUPYTER

Terminal:

```
'st.cache_data' or 'st.cache_resource'.
More information [in our docs](https://docs.streamlit.io/library/advanced-features/caching).
```

OUTLINE TIMELINE

Python streamlit powershell

Reconnect to Discord

28°C Berawan

Search

Go Live

21:36 07/06/2025

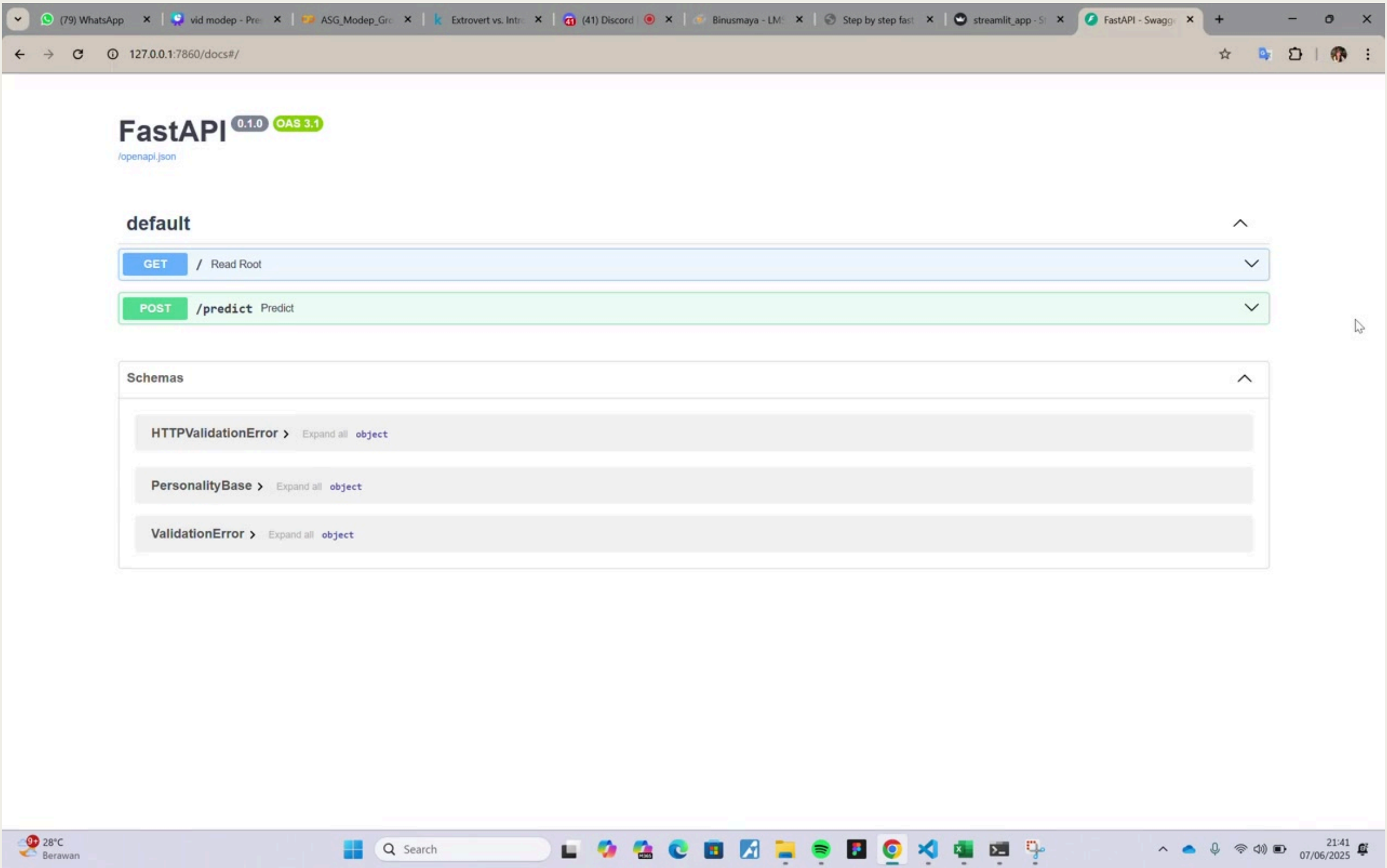
Hasil ouput nya udah sesuai dengan data yang kita punya, yaitu Extrovert

Status: 200 OK Size: 36 Bytes Time: 32 ms

Response	Headers ⁵	Cookies	Results	Docs
1	{			
2	"prediction": 0,			
3	"label": "Extrovert"			
4	}			

```
{'Time_spent_Alone': 4.0, 'Stage_fear': 0.0, 'Social_event_attendance': 4.0, 'Going_outside': 6.0, 'Drained_after_socializing': 0.0, 'Friends_circle_size': 13.0, 'Post_frequency': 5.0, 'Personality': 0.0}
```

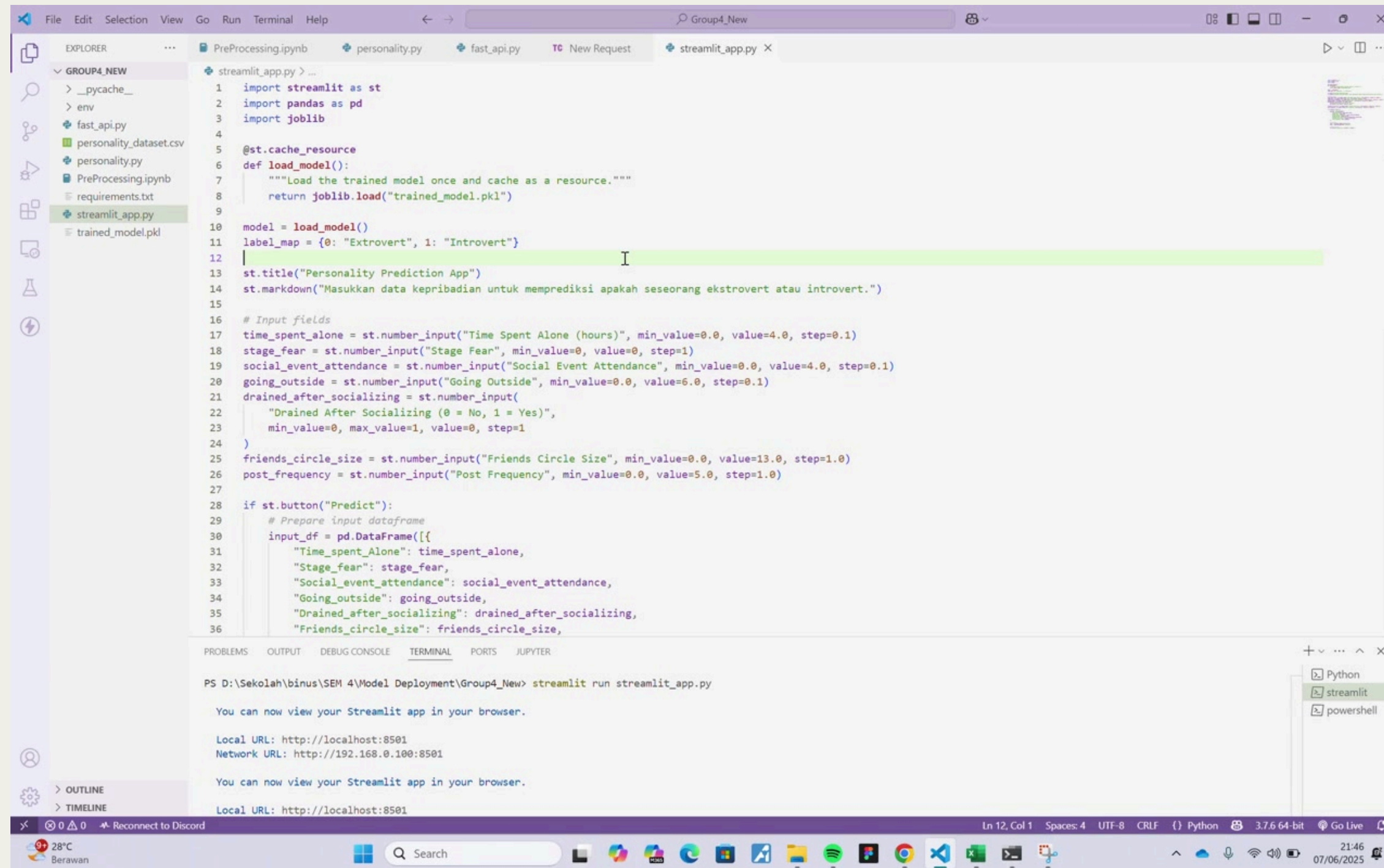
Setelah itu, kita juga coba input data di fastAPI nya, dan menunjukkan output yang sama seperti di Thunder Client



Setelah itu, kita juga coba input data di fastAPI nya, dan menunjukkan output yang sama seperti di Thunder Client

Code	Details
200	<div><div>Response body</div><div><pre>{ "prediction": 0, "label": "Extrovert" }</pre></div></div>

Selanjutnya, kita membuat codingan streamlitnya dan menjalankan 'streamlit run streamlit_app.py' di terminal vscode, lalu kita klik link yang tertera di terminal vscode



The screenshot shows the VS Code interface with a file explorer on the left, a code editor in the center, and a terminal at the bottom. The code editor displays the contents of `streamlit_app.py`, which is a Streamlit application for personality prediction. The code includes imports for `streamlit`, `pandas`, and `joblib`, a function to load a trained model, and several input fields for user data. A 'Predict' button is also present. The terminal shows the command `streamlit run streamlit_app.py` being executed, and the output displays the local and network URLs for viewing the app.

```
1 import streamlit as st
2 import pandas as pd
3 import joblib
4
5 @st.cache_resource
6 def load_model():
7     """Load the trained model once and cache as a resource."""
8     return joblib.load("trained_model.pkl")
9
10 model = load_model()
11 label_map = {0: "Extrovert", 1: "Introvert"}
12
13 st.title("Personality Prediction App")
14 st.markdown("Masukkan data kepribadian untuk memprediksi apakah seseorang ekstrovert atau introvert.")
15
16 # Input fields
17 time_spent_alone = st.number_input("Time Spent Alone (hours)", min_value=0.0, value=4.0, step=0.1)
18 stage_fear = st.number_input("Stage Fear", min_value=0, value=0, step=1)
19 social_event_attendance = st.number_input("Social Event Attendance", min_value=0.0, value=4.0, step=0.1)
20 going_outside = st.number_input("Going Outside", min_value=0.0, value=6.0, step=0.1)
21 drained_after_socializing = st.number_input(
22     "Drained After Socializing (0 = No, 1 = Yes)",
23     min_value=0, max_value=1, value=0, step=1
24 )
25 friends_circle_size = st.number_input("Friends Circle Size", min_value=0.0, value=13.0, step=1.0)
26 post_frequency = st.number_input("Post Frequency", min_value=0.0, value=5.0, step=1.0)
27
28 if st.button("Predict"):
29     # Prepare input dataframe
30     input_df = pd.DataFrame([
31         {
32             "Time_spent_Alone": time_spent_alone,
33             "Stage_fear": stage_fear,
34             "Social_event_attendance": social_event_attendance,
35             "Going_outside": going_outside,
36             "Drained_after_socializing": drained_after_socializing,
37             "Friends_circle_size": friends_circle_size,
```

PS D:\Sekolah\binus\SEM 4\Model Deployment\Group4_New> streamlit run streamlit_app.py

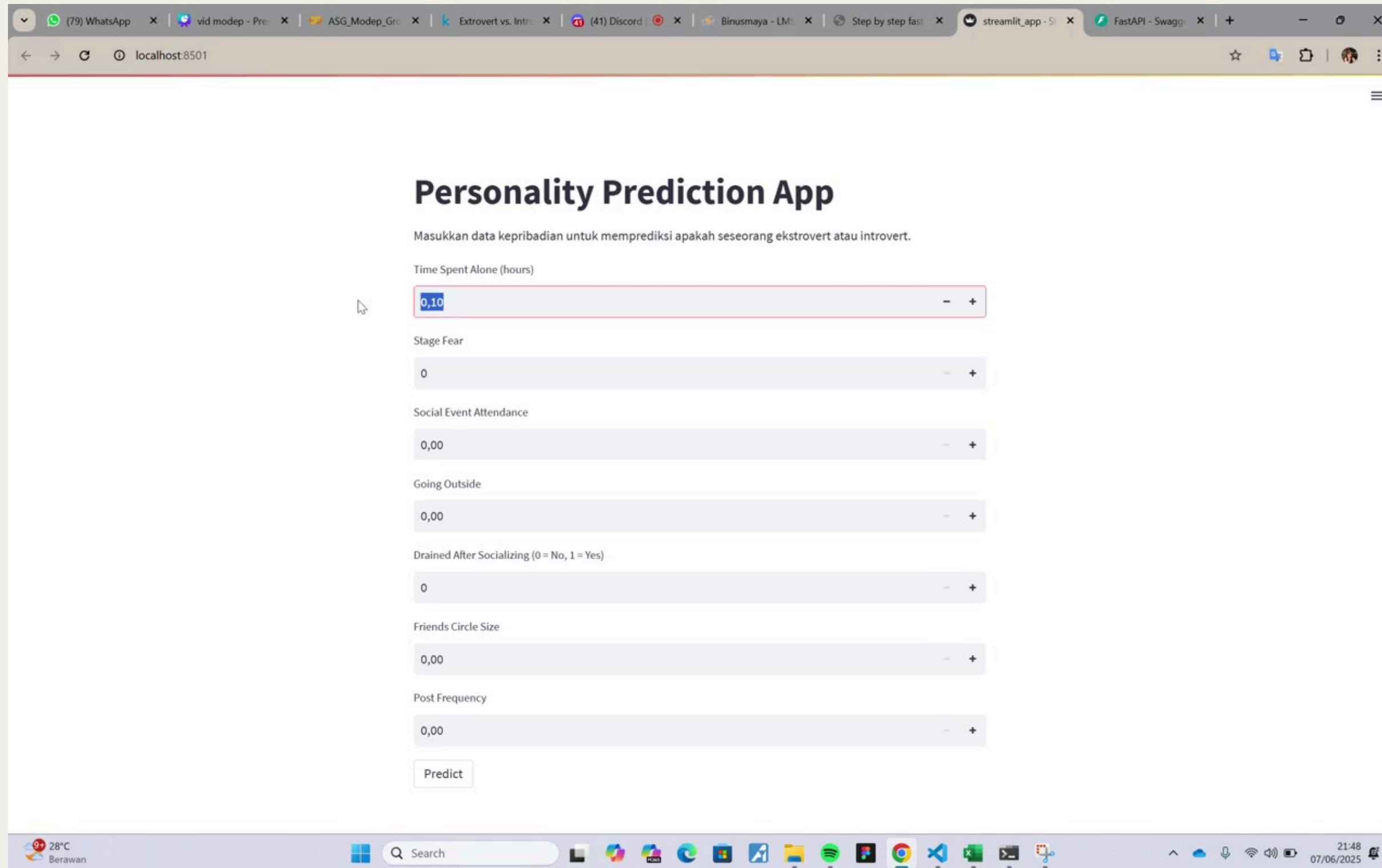
You can now view your Streamlit app in your browser.

Local URL: <http://localhost:8501>
Network URL: <http://192.168.0.100:8501>

You can now view your Streamlit app in your browser.

Local URL: <http://localhost:8501>

Redirect ke streamlit, kita langsung input sesuai dengan inputan kita di awal. Hasil output menunjukkan kalau orang adalah Extrovert



The screenshot shows a web browser window with multiple tabs open. The active tab is titled 'streamlit_app - S'. The address bar shows 'localhost:8501'. The page displays a 'Personality Prediction App' interface. The instructions state: 'Masukkan data kepribadian untuk memprediksi apakah seseorang ekstrovert atau introvert.' The form includes several input fields with sliders:

- Time Spent Alone (hours): 0,10
- Stage Fear: 0
- Social Event Attendance: 0,00
- Going Outside: 0,00
- Drained After Socializing (0 = No, 1 = Yes): 0
- Friends Circle Size: 0,00
- Post Frequency: 0,00

A 'Predict' button is located at the bottom of the form. The Windows taskbar at the bottom shows the system clock as 21:48 on 07/06/2025, and the weather as 28°C in Berawan.

Thank You!