

Application Building

Team ID	PNT2022TMID13268
Project Name	Smart Lender Applicant Credibility Prediction for Loan Approval

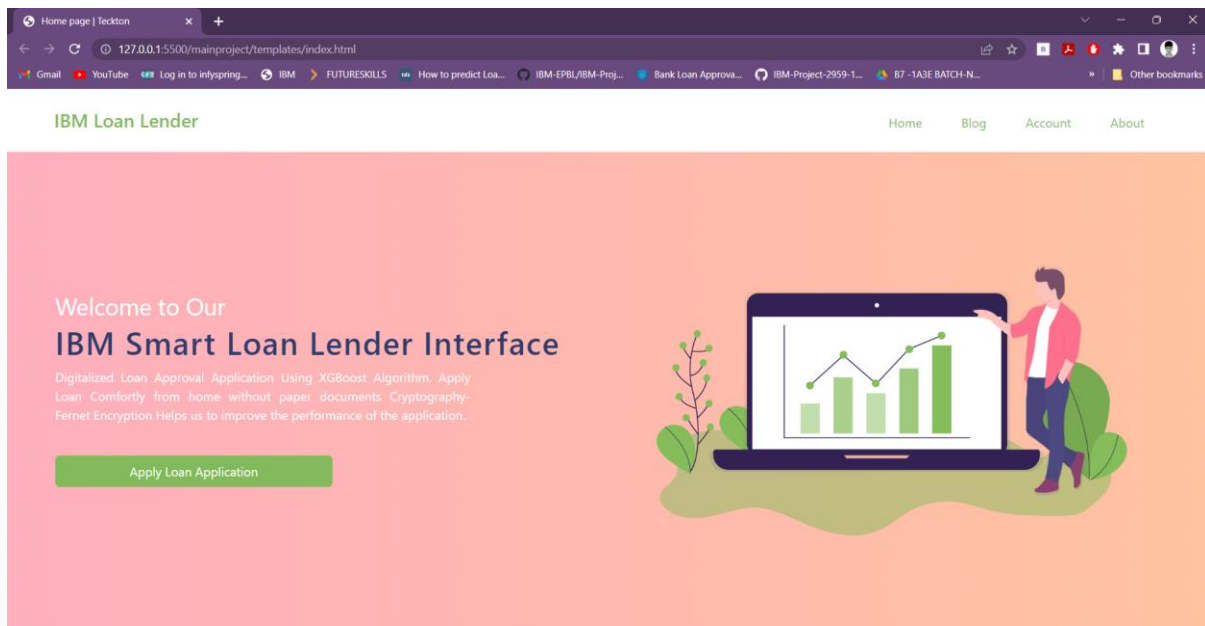
Application Building:

This section has the following tasks

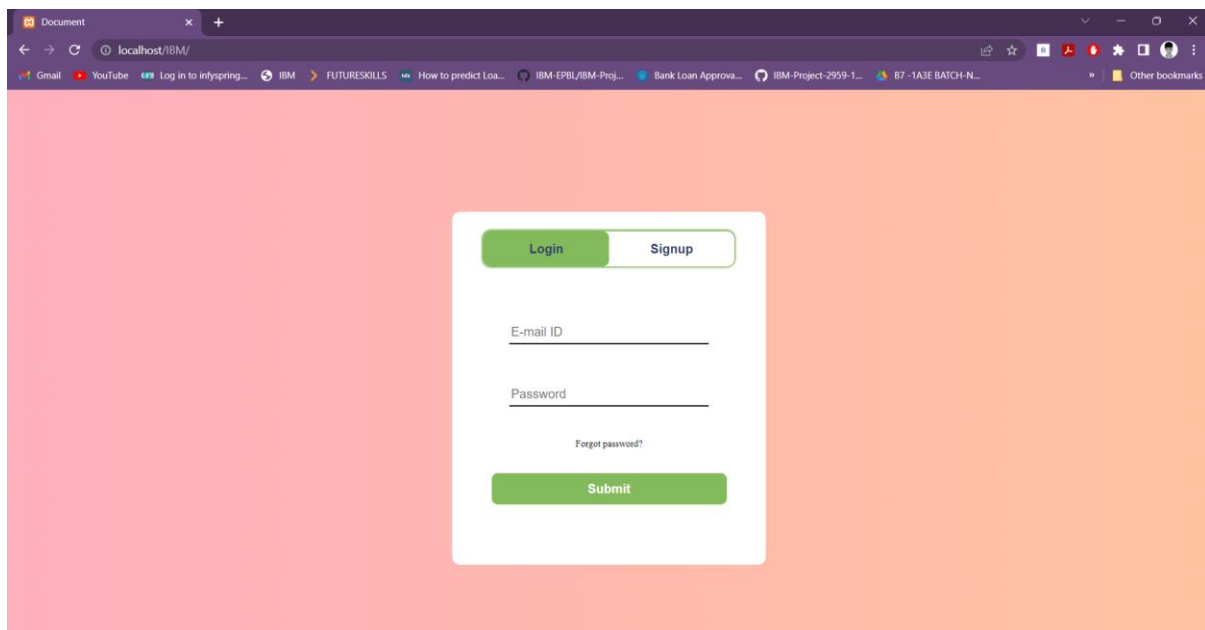
- Building HTML Pages
- Building serverside script

Building HTML Pages

index.html



Login/Register Page:



application.html

Loan Approval Prediction

127.0.0.1:5500/mainproject/templates/application.html

Fill the Loan Application

Enter Valid Details

Your Name

dd-mm-yyyy

Your Gender

Your Education

Are You Employed?

Your Marital Status

Number of Dependents

Your Property Area

Applicant Income per Month (\$/₹)

Co-Applicant Income per Month (\$/₹)


Loan and Credit Description

Your Loan Amount (\$/₹)

Your Loan Term (days)

Your Credit History

☐ I agree all statements in [Terms of service](#)




prediction.html

Result of Application

localhost / 127.0.0.1 / data_protect / 127.0.0.1:5000/prediction

Application Status



Our Loan approval system will give you the most effective classification of loan approval status according to the Debtor's Details. We used **XGBoost**. The results are displaying below.

Dear Mr/Mrs/Ms Jagan, your loan is approved!

[Back to Home](#)

Build Python Code

```
JS sever1.js Machine Learning Model Dev.py nb mainserver.py X
mainproject > mainserver.py > ...
1 from flask import Flask, redirect, url_for, request, jsonify, render_template
2 # Data manipulation
3 import pandas as pd
4 # Matrices manipulation
5 import numpy as np
6 # Script logging
7 import logging
8 # ML model
9 import joblib
10 # JSON manipulation
11 import json
12 # Utilities
13 import sys
14 import os
15
16 import json
17 import requests
18 from cryptography.fernet import Fernet
19 import mysql.connector
20 def sample(e,r1):
21     a=["http://localhost:8084/", "http://localhost:8081/", "http://localhost:8082/", "http://localhost:8083/"]
22     def datainsert(a):
23         mydb = mysql.connector.connect(
24             host="localhost",
25             user="root",
26             password="",
27             database="data_protection"
28         )
29         mycursor = mydb.cursor()
30
31         sql = "INSERT INTO data (emailid,password1,password2,password3,password4,key1,key2,key3,key4) VALUES (%s,%s,%s,%s,%s,%s,%s,%s,%s,%s)"
32         val = (a,"","","","","","","","","")
33         mycursor.execute(sql, val)
34
35         mydb.commit()
36     def sep(s):
37
38         c=int(len(s)/4)
39         k=c
40         data=[]
41         i=0
42         for j in range(4):
43             if (j==3):
44                 data.append(s[i:])
45             else:
46                 data.append(s[i:c])
47                 i=i+k
48                 c=c+k
49                 data.append(s[i:c])
```

```
mainproject > mainserver.py > ...
49     return data
50
51 def encrypt(message):
52     data=[]
53     key = Fernet.generate_key()
54     print(key)
55     data.append(key)
56     fernet = Fernet(key)
57     encMessage = fernet.encrypt(message.encode())
58     print(encMessage)
59     data.append(encMessage)
60     return data
61 data=encrypt(r1)
62 password=sep(data[1])
63 key=sep(data[0])
64
65 def postdata(a,b,c):
66     response=requests.post(a,{
67         "email":e,
68         "password":b,
69         "key":c
70     })
71 datainsert(e)
72 for i in range(4):
73     postdata(a[i],password[i],key[i])
74 def checkpass(a1,b):
75     password=[]
76     key=[]
77     a=["http://localhost:8084/", "http://localhost:8081/", "http://localhost:8082/", "http://localhost:8083/"]
78     for i in a:
79         x = requests.get(i,data=a1)
80         r=x.text.split(" ")
81         print(x.text)
82         password.append(r[0])
83         key.append(r[1])
84     password="".join(password).encode()
85     key="".join(key).encode()
86     f=Fernet(key)
87     rp=f.decrypt(password).decode()
88     if (rp==b):
89         return 1
90     else:
91         return 0
92 def checkemail(a):
93     mydb = mysql.connector.connect(
94         host="localhost",
95         user="root",
96         password="",
```

mainproject > mainserver.py > ...

```
98     )
99
100     mycursor = mydb.cursor()
101
102     mycursor.execute("SELECT * FROM data WHERE emailid='"+a+"'")
103
104     myresult = mycursor.fetchall()
105     if(len(myresult)!=0):
106         return 1
107     else:
108         return 0
109 app = Flask(__name__)
110 a=[]
111 @app.route('/',methods=["POST","GET"])
112 def hello_world():
113     if request.method=="POST":
114         user=request.form['emails']
115         user1=request.form['passwords']
116         if(checkemail(user)==0):
117             sample(user,user1)
118             return render_template("index.html")
119         elif(checkemail(user)==1):
120             return redirect("http://localhost/IBM/index.php?signid=alr")
121     else:
122         return "success"
123 @app.route('/login',methods=["POST","GET"])
124 def hello_world1():
125     if request.method=="POST":
126         user=request.form['email']
127         user1=request.form['password']
128         if(checkemail(user)==0):
129             return redirect("http://localhost/IBM/index.php?id=not")
130         elif(checkpass(user,user1)==0):
131             return redirect("http://localhost/IBM/index.php?pass=not")
132         else:
133
134             return render_template("index.html")
135     else:
136         return "success"
137
138 # Current directory
139 current_dir = os.path.dirname(__file__)
140
141 # Function
142 def ValuePredictor(data = pd.DataFrame):
143     # Model name
144     model_name = 'bin/xgboostModel.pkl'
145     # Directory where the model is stored
```

mainproject > mainserver.py > ...

```
144     model_name = 'bin/xgboostModel.pkl'
145     # Directory where the model is stored
146     model_dir = os.path.join(current_dir, model_name)
147     # Load the model
148     loaded_model = joblib.load(open(model_dir, 'rb'))
149     # Predict the data
150     result = loaded_model.predict(data)
151     return result[0]
152
153 # Application page
154 @app.route('/application')
155 def home():
156     return render_template('application.html')
157
158 # Prediction page
159 @app.route('/prediction', methods = ['POST'])
160 def predict():
161     if request.method == 'POST':
162         # Get the data from form
163         name = request.form['name']
164         gender = request.form['gender']
165         education = request.form['education']
166         self_employed = request.form['self_employed']
167         marital_status = request.form['marital_status']
168         dependents = request.form['dependents']
169         applicant_income = request.form['applicant_income']
170         coapplicant_income = request.form['coapplicant_income']
171         loan_amount = request.form['loan_amount']
172         loan_term = request.form['loan_term']
173         credit_history = request.form['credit_history']
174         property_area = request.form['property_area']
175
176         # Load template of JSON file containing columns name
177         # Schema name
178         schema_name = 'data/columns_set.json'
179         # Directory where the schema is stored
180         schema_dir = os.path.join(current_dir, schema_name)
181         with open(schema_dir, 'r') as f:
182             cols = json.loads(f.read())
183             schema_cols = cols['data_columns']
184
185         # Parse the categorical columns
186         # Column of dependents
187         try:
188             col = ('Dependents_' + str(dependents))
189             if col in schema_cols.keys():
190                 schema_cols[col] = 1
191             else:
```

```

mainproject > mainserver.py > ...
191         else:
192             pass
193     except:
194         pass
195     # Column of property area
196     try:
197         col = ('Property_Area_' + str(property_area))
198         if col in schema_cols.keys():
199             schema_cols[col] = 1
200         else:
201             pass
202     except:
203         pass
204
205     # Parse the numerical columns
206     schema_cols['ApplicantIncome'] = applicant_income
207     schema_cols['CoapplicantIncome'] = coapplicant_income
208     schema_cols['LoanAmount'] = loan_amount
209     schema_cols['Loan_Amount_Term'] = loan_term
210     schema_cols['Gender_Male'] = gender
211     schema_cols['Married_Yes'] = marital_status
212     schema_cols['Education_Not Graduate'] = education
213     schema_cols['Self_Employed_Yes'] = self_employed
214     schema_cols['Credit_History_1.0'] = credit_history
215
216     # Convert the JSON into data frame
217     df = pd.DataFrame(
218         data = {k: [v] for k, v in schema_cols.items()},
219         dtype = float
220     )
221
222     # Create a prediction
223     print(df.dtypes)
224     result = ValuePredictor(data = df)
225
226     # Determine the output
227     if int(result) == 1:
228         prediction = 'Dear Mr/Mrs/Ms {name}, your loan is approved!'.format(name = name)
229     else:
230         prediction = 'Sorry Mr/Mrs/Ms {name}, your loan is rejected!'.format(name = name)
231
232     # Return the prediction
233     return render_template('prediction.html', prediction = prediction)
234
235     # Something error
236     else:
237         # Return error
238         return render_template('error.html', prediction = prediction)

```

```

226     # Determine the output
227     if int(result) == 1:
228         prediction = 'Dear Mr/Mrs/Ms {name}, your loan is approved!'.format(name = name)
229     else:
230         prediction = 'Sorry Mr/Mrs/Ms {name}, your loan is
231                                     (variable) prediction: str =)
232     # Return the prediction
233     return render_template('prediction.html', prediction = prediction)
234                                     (variable) prediction: str
235
236     # Something error
237     else:
238         # Return error
239         return render_template('error.html', prediction = prediction)
240
241 if __name__ == '__main__':
242     app.run(debug = True)
243
244

```

Run The Application:

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER COMMENTS
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS V:\XAMPP\htdocs\IBM> & C:/Users/jegan/AppData/Local/Programs/Python/Python310/python.exe v:/XAMPP/htdocs/IBM/mainproject/mainserver.py
* Serving Flask app 'mainserver'
* 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
* Restarting with watchdog (windowsapi)
* Debugger is active!
* Debugger PIN: 307-777-028
```

