

Fake News Faux Real :

Lab 8

Login/SignUp Testing:

Code:

```

main.py x
C:\Users\Aditi> Downloads > signup_signup > new_g_forlder > python-firebase-flask-login > main.py > ...
1
2
3 import pyrebase
4 from flask import Flask, flash, redirect, render_template, request, session, abort, url_for
5
6 app = Flask(__name__) #Initialize flask constructor
7
8 #Add your own details
9 config = {
10     "apiKey": "AIzaSyDwU9M-A9KrdDNtbo6D_qi8wQdKE3RVTHg",
11     "authDomain": "mypro-ce6df.firebaseio.com",
12     "projectId": "mypro-ce6df",
13     "storageBucket": "mypro-ce6df.appspot.com",
14     "messagingSenderId": "604651206875",
15     "appId": "1:604651206875:web:ee1c4de5b8b4367fb34fe",
16     "measurementId": "G-NSHDZ1BFPH",
17     "databaseURL": ""
18 }
19
20 #initialize firebase
21 firebase = pyrebase.initialize_app(config)
22 auth = firebase.auth()
23 db = firebase.database()
24
25 #Initialize person as dictionary
26 person = {"is_logged_in": False, "name": "", "email": "", "uid": ""}
27
28 #Login
29 @app.route("/")
30 def login():
31     return render_template("login.html")
32
33 #Sign up/ Register
34 @app.route("/signup")
35 def signup():
36     return render_template("signup.html")

```

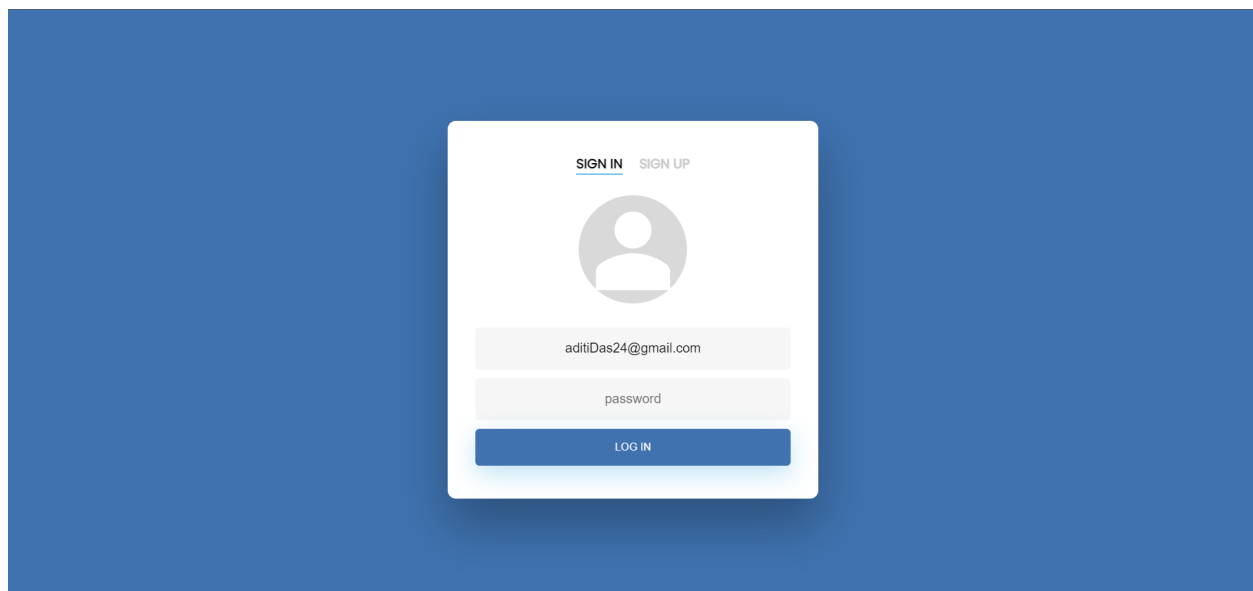
```

main.py x
C:\Users\Aditi> Downloads > signup_signup > new_g_forlder > python-firebase-flask-login > main.py > ...
38 #welcome page
39 @app.route("/welcome")
40 def welcome():
41     if person["is_logged_in"] == True:
42         return render_template("welcome.html", email = person["email"], name = person["name"])
43     else:
44         return redirect(url_for('login'))
45
46 #If someone clicks on login, they are redirected to /result
47 @app.route("/result", methods = ["POST", "GET"])
48 def result():
49     unsuccessful = 'Please check your credentials'
50     successful = 'Login successful'
51     if request.method == 'POST':
52         email = request.form['email']
53         password = request.form['pass']
54         try:
55             auth.sign_in_with_email_and_password(email, password)
56             return render_template("welcome.html", s=successful)
57         except:
58             return render_template("register.html", us=unsuccessful)
59     return render_template("login.html")
60
61 #If someone clicks on register, they are redirected to /register
62 @app.route("/register", methods = ["POST", "GET"])
63 def register():
64     if request.method == "POST": #Only listen to POST
65         result = request.form #Get the data submitted
66         email = result["email"]
67         password = result["pass"]
68         name = result["name"]
69         try:
70             #Try creating the user account using the provided data
71             auth.create_user_with_email_and_password(email, password)
72             #login the user
73             user = auth.sign_in_with_email_and_password(email, password)
74             #Add data to global person

```

```
main.py x
C:\Users\Aditi> Downloads > signup_signup > new_g_forlder > python-firebase-flask-login > main.py > ...
73 user = auth.sign_in_with_email_and_password(email, password)
74 #Add data to global person
75 global person
76 person["is_logged_in"] = True
77 person["email"] = user["email"]
78 person["uid"] = user["localId"]
79 person["name"] = name
80 #Append data to the firebase realtime database
81 data = {"name": name, "email": email}
82 db.child("users").child(person["uid"]).set(data)
83 #Go to welcome page
84 return redirect(url_for('welcome'))
85 except:
86     #if there is any error, redirect to register
87     return redirect(url_for('register'))
88
89 else:
90     if person["is_logged_in"] == True:
91         return redirect(url_for('welcome'))
92     else:
93         return redirect(url_for('register'))
94
95 if __name__ == "__main__":
96     app.run()
97
```

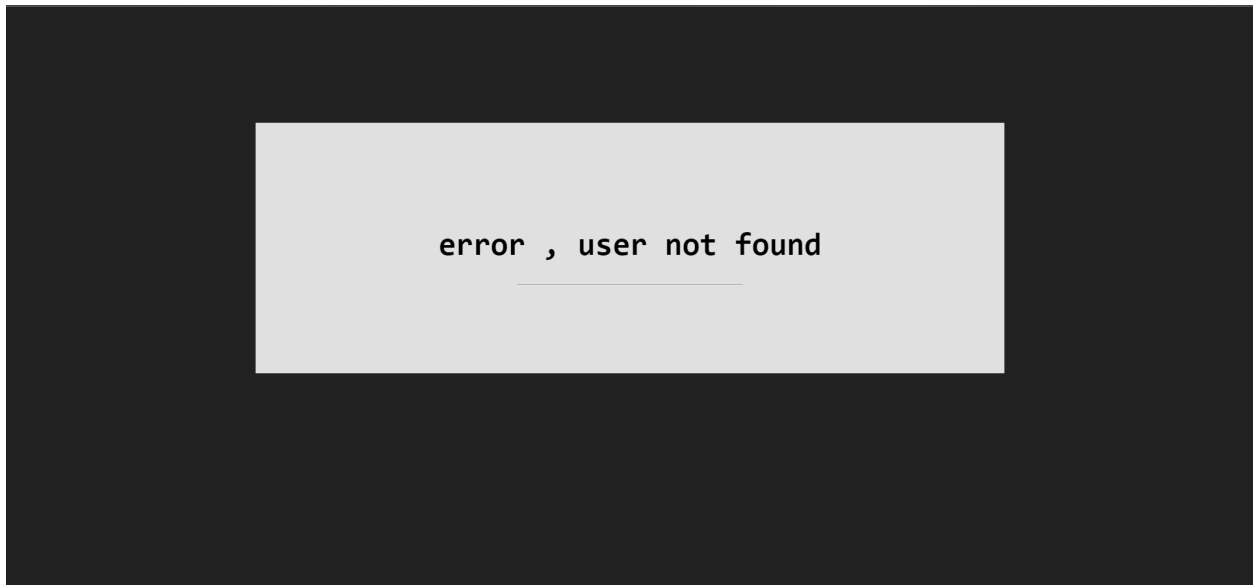
Output:



Testing:

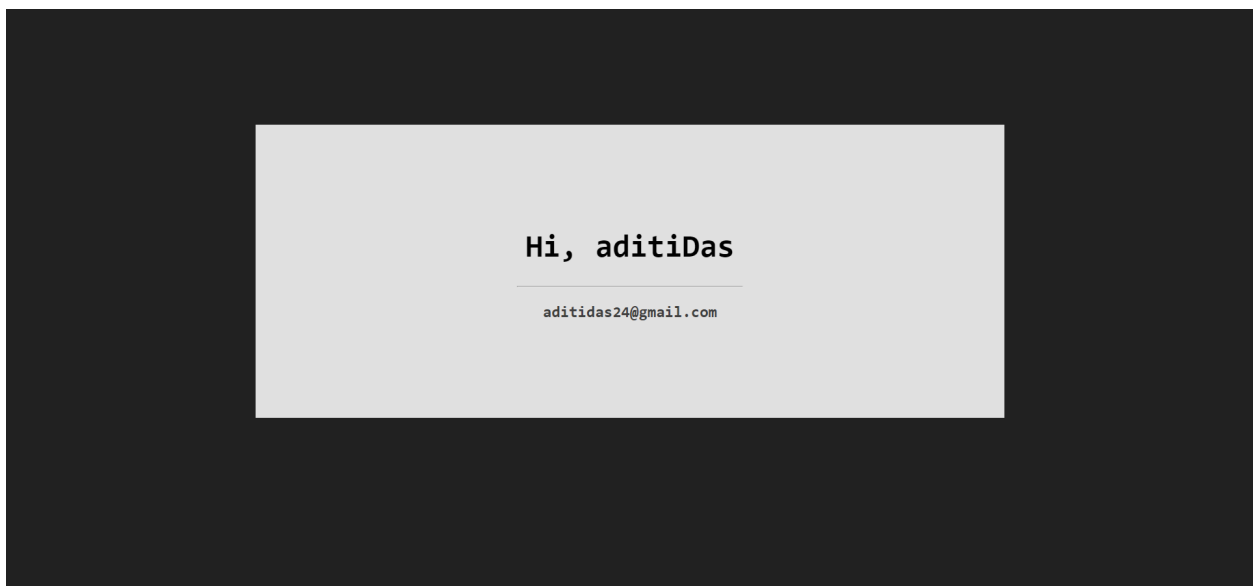
a) We first tried logging in with a ID that was not registered:

Output:



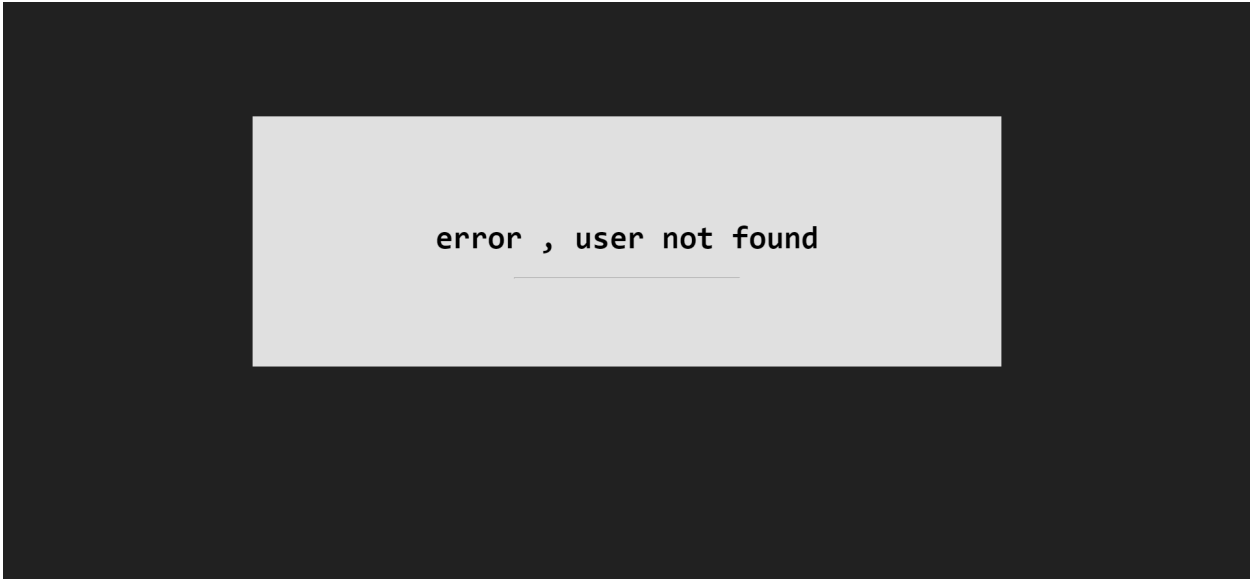
b) We then tried signing up:

Output:



- c) We then tried logging in with a registered username but wrong password

Output:

A screenshot of a terminal window with a dark background. A light gray rectangular box is centered in the terminal, containing the text "error , user not found" in a monospaced font. A horizontal line is positioned below the text.

error , user not found

- d) We finally logged in with the correct username and password

Output:

A screenshot of a terminal window with a dark background. A light gray rectangular box is centered in the terminal, containing the text "Hi," in a monospaced font. A horizontal line is positioned below the text.

Hi,

Machine Learning Testing:

We ran three different algorithms to obtain the output in increasing order of accuracy

Code:

```
def testing(model,text):
    model.fit(X_train,y_train)
    y_pred=model.predict(X_test)
    class_report=classification_report(y_test,y_pred)
    print(class_report)
    print('-----')

    print("Accuracy: " + str( accuracy_score(y_test, y_pred)*100 ) + "%")
    print('\n')

    print('')
    print('Cross-validation scores with 5 folds:')
    print('')
    print(f"ROC AUC: {round(cross_val_score(model, X_train, y_train, cv = 5, scoring = 'roc_auc').mean(), 3)}")
    print(f"precision: {round(cross_val_score(model, X_train, y_train, cv = 5, scoring = 'precision').mean(), 2)}")
    print(f"recall: {round(cross_val_score(model, X_train, y_train, cv = 5, scoring = 'recall').mean(), 2)}")
    print(f"f1: {round(cross_val_score(model, X_train, y_train, cv = 5, scoring = 'f1').mean(), 2)}")

    auc = roc_auc_score(y_test, y_pred)
    print('Area under the ROC Curve for Testing Set is: ' + str(auc))

    print('-----')

    cm = confusion_matrix(y_test, y_pred)
    predicted_probab_log = model.predict_proba(X_test)
    predicted_probab_log = predicted_probab_log[:, 1]
    fpr, tpr, _ = roc_curve(y_test, predicted_probab_log)
    y_pred = model.predict(X)

    plot_confusion_matrix(y, y_pred)

    print('-----')

    from matplotlib import pyplot
    pyplot.plot(fpr, tpr, marker='.', label=text)
    pyplot.xlabel('False Positive Rate')
    pyplot.ylabel('True Positive Rate')
    pyplot.legend()
    pyplot.grid()
    pyplot.show()
```

Results:

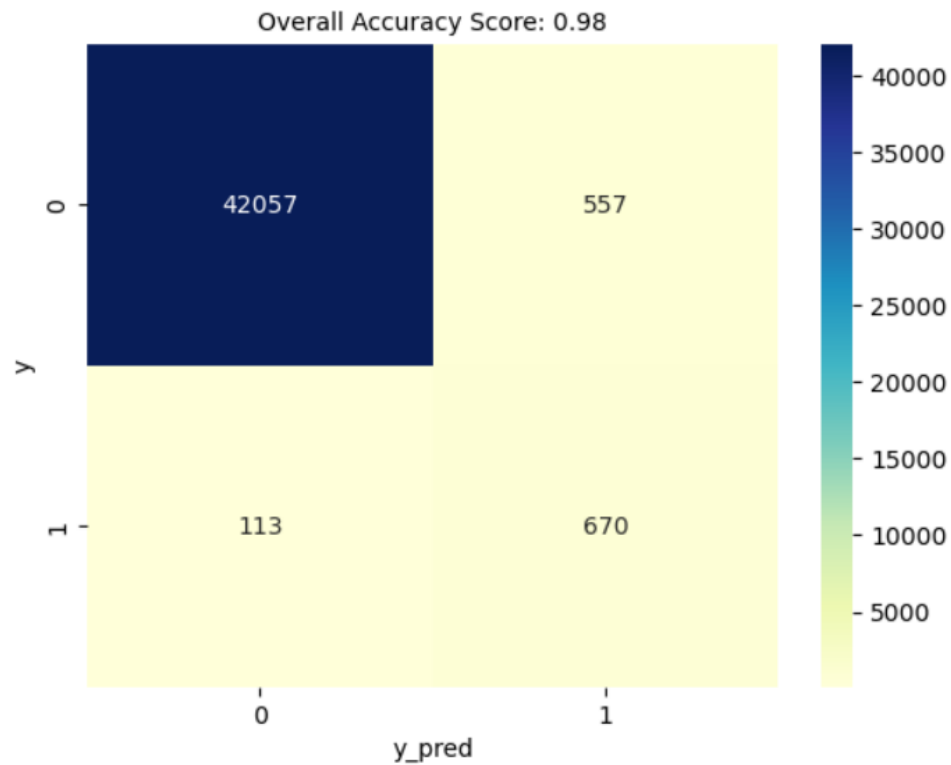
XGBoostClassifier

	precision	recall	f1-score	support
0	0.98	0.98	0.98	4896
1	0.98	0.98	0.98	4949
accuracy			0.98	9845
macro avg	0.98	0.98	0.98	9845
weighted avg	0.98	0.98	0.98	9845

Accuracy: 98.02945657694261%

Area under the ROC Curve is for Validation Set: 0.8982229870196555

Area under the ROC Curve for Testing Set is: 0.9802721212247275



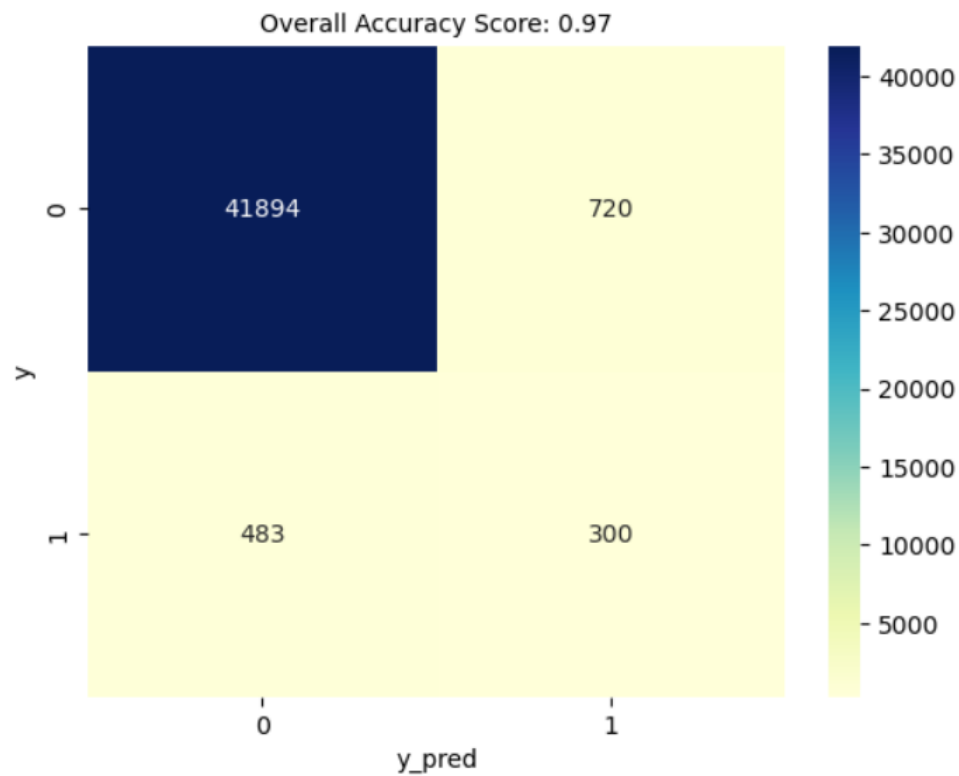
RandomForestClassifier

	precision	recall	f1-score	support
0	0.99	0.96	0.98	4896
1	0.96	0.99	0.98	4949
accuracy			0.98	9845
macro avg	0.98	0.98	0.98	9845
weighted avg	0.98	0.98	0.98	9845

Accuracy: 97.55205688166582%

Area under the ROC Curve is for Validation Set: 0.8982229870196555

Area under the ROC Curve for Testing Set is: 0.9754569732183304



Support Vector Machine

	precision	recall	f1-score	support
0	0.94	0.88	0.91	4896
1	0.89	0.94	0.92	4949
accuracy			0.91	9845
macro avg	0.91	0.91	0.91	9845
weighted avg	0.91	0.91	0.91	9845

Accuracy: 91.2544438801422%

Cross-validation scores with 5 folds:

ROC AUC: 0.971
precision: 0.88
recall: 0.94
f1: 0.91
Area under the ROC Curve for Testing Set is: 0.9123849622357192

