National College of Ireland

Configuration Manual

MSc Research Project
Cyber Security

Siddharth Sankar
Student ID: 23418541

School of Computing
National College of Ireland

Supervisor: Niall Heffernan

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name: Siddharth Sankar

Student ID: 23418541

Programme: MSc Cyber Security **Year:** 2025

Module: MSc Research Project

Supervisor: Niall Heffernan

Submission Due Date: 28/01/2026

Project Title: Continuous User Authentication on Mobile and Desktop Devices using Behavioural Biometrics – a Multimodal Approach

Word Count: **198** **Page Count 8**

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: Siddharth Sankar

Date: 27/01/2026

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

Siddharth Sankar
Student ID: 23418541

1 System Requirements

The project was intentionally kept lightweight so that it could be easily replicated. Here is the ideal setup to replicate the implementation:

1.1 Hardware Used

A Dell Inspiron laptop with the following specifications:

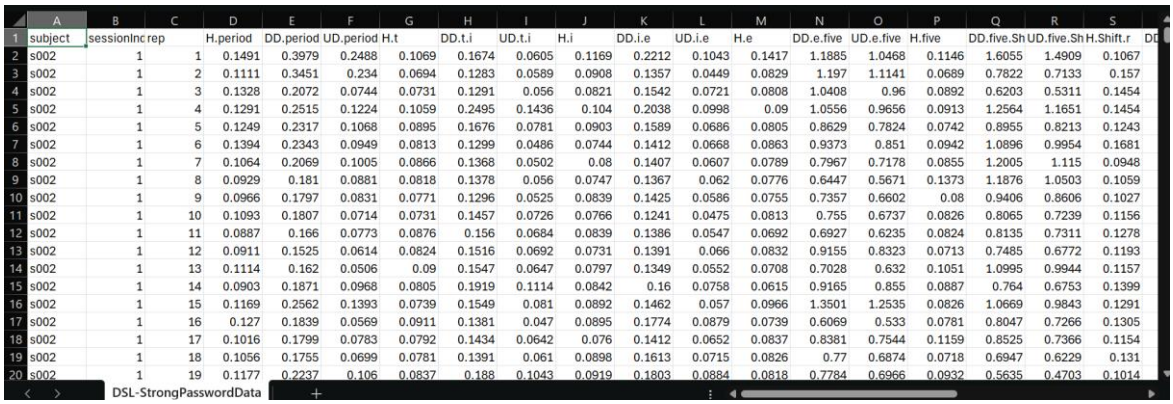
- 16GB RAM
- 11th Gen Intel (R) Core™ i7-11370H @ 3.30 GHz Processor
- Storage 256GB
- Windows 11 Operating System

1.2 Software Used

- Environment: Google Colab (can be implemented locally using Jupiter Notebook as Well)
- Language: Python 3.9
- Libraries: Pandas, Numpy, Scikit-Learn, Matplotlib, Seaborn and ScipPy were used.
- Browser: Brave

2 Data Handling

2.1 Desktop Dataset

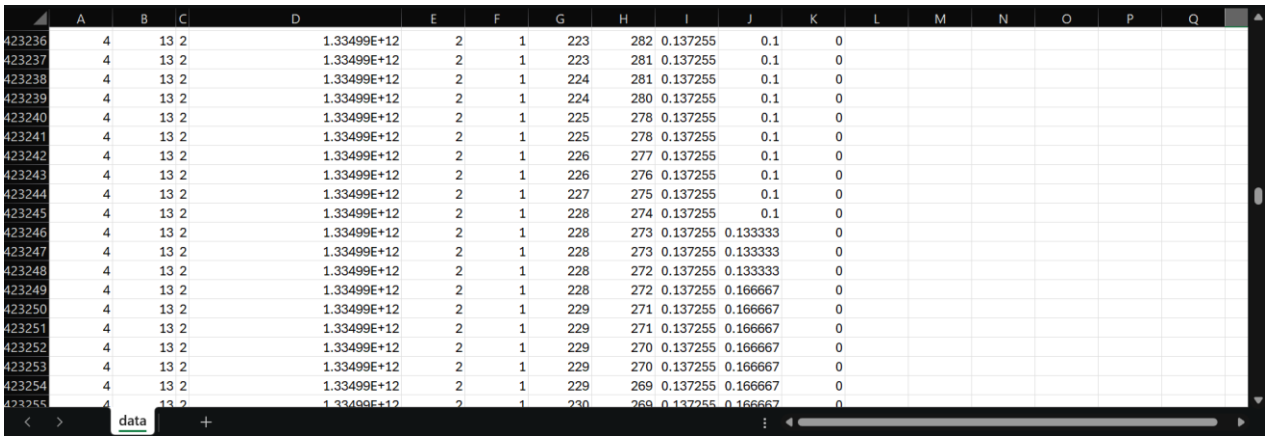


	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
1	subject	sessionInc	rep	H.period	DD.period	UD.period	H.t	DD.t.i	UD.t.i	H.i	DD.i.e	UD.i.e	H.e	DD.e.five	UD.e.five	H.five	DD.five.Sh	UD.five.Sh	H.Shift.r
2	s002	1	1	0.1491	0.3979	0.2488	0.1069	0.1674	0.0605	0.1169	0.2212	0.1043	0.1417	1.1885	1.0468	0.1146	1.6055	1.4909	0.1067
3	s002	1	2	0.1111	0.3451	0.234	0.0694	0.1283	0.0589	0.0908	0.1357	0.0449	0.0829	1.197	1.1141	0.0689	0.7822	0.7133	0.157
4	s002	1	3	0.1328	0.2072	0.0744	0.0731	0.1291	0.056	0.0821	0.1542	0.0721	0.0808	1.0408	0.96	0.0892	0.6203	0.5311	0.1454
5	s002	1	4	0.1291	0.2515	0.1224	0.1059	0.2495	0.1436	0.104	0.2038	0.0998	0.09	1.0556	0.9656	0.0913	1.2564	1.1651	0.1454
6	s002	1	5	0.1249	0.2317	0.1068	0.0895	0.1676	0.0781	0.0903	0.1589	0.0686	0.0805	0.8629	0.7824	0.0742	0.8955	0.8213	0.1243
7	s002	1	6	0.1394	0.2343	0.0949	0.0813	0.1299	0.0486	0.0744	0.1412	0.0668	0.0863	0.9373	0.851	0.0942	1.0896	0.9954	0.1681
8	s002	1	7	0.1064	0.2069	0.1005	0.0866	0.1368	0.0502	0.08	0.1407	0.0607	0.0789	0.7967	0.7178	0.0855	1.2005	1.115	0.0948
9	s002	1	8	0.0929	0.181	0.0881	0.0818	0.1378	0.056	0.0747	0.1367	0.062	0.0776	0.6447	0.5671	0.1373	1.1876	1.0503	0.1059
10	s002	1	9	0.0966	0.1797	0.0831	0.0771	0.1296	0.0525	0.0839	0.1425	0.0586	0.0755	0.7357	0.6602	0.08	0.9406	0.8606	0.1027
11	s002	1	10	0.1093	0.1807	0.0714	0.0731	0.1457	0.0726	0.0766	0.1241	0.0475	0.0813	0.755	0.6737	0.0826	0.8065	0.7239	0.1156
12	s002	1	11	0.0887	0.166	0.0773	0.0876	0.156	0.0684	0.0839	0.1386	0.0547	0.0692	0.6927	0.6235	0.0824	0.8135	0.7311	0.1278
13	s002	1	12	0.0911	0.1525	0.0614	0.0824	0.1516	0.0692	0.0731	0.1391	0.066	0.0832	0.9155	0.8323	0.0713	0.7485	0.6772	0.1193
14	s002	1	13	0.1114	0.162	0.0506	0.09	0.1547	0.0647	0.0797	0.1349	0.0552	0.0708	0.7028	0.632	0.1051	1.0995	0.9944	0.1157
15	s002	1	14	0.0903	0.1871	0.0968	0.0805	0.1919	0.1114	0.0842	0.16	0.0758	0.0615	0.9165	0.855	0.0887	0.764	0.6753	0.1399
16	s002	1	15	0.1169	0.2562	0.1393	0.0739	0.1549	0.081	0.0892	0.1462	0.057	0.0966	1.3501	1.2535	0.0826	1.0669	0.9843	0.1291
17	s002	1	16	0.127	0.1839	0.0569	0.0911	0.1381	0.047	0.0895	0.1774	0.0879	0.0739	0.6069	0.533	0.0781	0.8047	0.7266	0.1305
18	s002	1	17	0.1016	0.1799	0.0783	0.0792	0.1434	0.0642	0.076	0.1412	0.0652	0.0837	0.8381	0.7544	0.1159	0.8525	0.7366	0.1154
19	s002	1	18	0.1056	0.1755	0.0699	0.0781	0.1391	0.061	0.0898	0.1613	0.0715	0.0826	0.77	0.6874	0.0718	0.6947	0.6229	0.131
20	s002	1	19	0.1177	0.2237	0.106	0.0837	0.188	0.1043	0.0919	0.1803	0.0884	0.0818	0.7784	0.6966	0.0932	0.5635	0.4703	0.1014

DSL-StrongPasswordData

Source: https://www.cs.cmu.edu/~keystroke/#sec2
Configuration: Download the Csv file and upload it onto Colab File System

2.2 Mobile Dataset (TouchAnalytics)



	A	B	C	D	E	F	G	H	I	J	K
423236	4	13	2	1.33499E+12	2	1	223	282	0.137255	0.1	0
423237	4	13	2	1.33499E+12	2	1	223	281	0.137255	0.1	0
423238	4	13	2	1.33499E+12	2	1	224	281	0.137255	0.1	0
423239	4	13	2	1.33499E+12	2	1	224	280	0.137255	0.1	0
423240	4	13	2	1.33499E+12	2	1	225	278	0.137255	0.1	0
423241	4	13	2	1.33499E+12	2	1	225	278	0.137255	0.1	0
423242	4	13	2	1.33499E+12	2	1	226	277	0.137255	0.1	0
423243	4	13	2	1.33499E+12	2	1	226	276	0.137255	0.1	0
423244	4	13	2	1.33499E+12	2	1	227	275	0.137255	0.1	0
423245	4	13	2	1.33499E+12	2	1	228	274	0.137255	0.1	0
423246	4	13	2	1.33499E+12	2	1	228	273	0.137255	0.133333	0
423247	4	13	2	1.33499E+12	2	1	228	273	0.137255	0.133333	0
423248	4	13	2	1.33499E+12	2	1	228	272	0.137255	0.133333	0
423249	4	13	2	1.33499E+12	2	1	228	272	0.137255	0.166667	0
423250	4	13	2	1.33499E+12	2	1	229	271	0.137255	0.166667	0
423251	4	13	2	1.33499E+12	2	1	229	271	0.137255	0.166667	0
423252	4	13	2	1.33499E+12	2	1	229	270	0.137255	0.166667	0
423253	4	13	2	1.33499E+12	2	1	229	270	0.137255	0.166667	0
423254	4	13	2	1.33499E+12	2	1	229	269	0.137255	0.166667	0

Source: https://www.mariofrank.net/touchalytics/
Configuration: Download the Csv file and upload it onto Colab File System

2.3 Data Loading and Preprocessing

2.3.1 Loading Mobile Data and assigning Indices to features

```
print("--- STEP 1: LOADING & PAIRING DATA ---")
try:
    df_mob = pd.read_csv('/content/data.csv', header=None, engine='python')
except FileNotFoundError:
    print("ERROR: '/content/data.csv' not found. Please upload your mobile data file.")
    exit()

#Given Indices: [1:user, 3:time, 4:action, 6:x, 7:y, 8:pres]
USER, TIME, ACTION, X, Y, PRES = 1, 3, 4, 6, 7, 8

def get_mobile_features(df):
    data = df.values
    feats, labs = [], []
    stroke = []
    for row in data:
        try:
            if row[ACTION] == 0: stroke = [row]
            elif row[ACTION] == 2 and stroke: stroke.append(row)
            elif row[ACTION] == 1 and len(stroke)>1:
                stroke.append(row)
                s = np.array(stroke)
                if s[0,USER] == s[-1,USER]:
                    dur = s[-1,TIME]-s[0,TIME]
                    dist = np.sum(np.sqrt(np.diff(s[:,X].astype(float))**2 + np.diff(s[:,Y].astype(float))**2))
                    pres = np.mean(s[:,PRES].astype(float))
                    if dur>50 and dist>10:
                        feats.append([dur, dist, pres, dist/dur])
                        labs.append(s[0,USER])
                stroke=[]
        except: continue
    return np.array(feats), np.array(labs)

X_m_all, y_m_all = get_mobile_features(df_mob)
print(f"Mobile Loaded: {len(X_m_all)} strokes from {len(np.unique(y_m_all))} users.")
```

2.3.2 Loading Mobile Data and Keeping the first 41 users to match Mobile Data

```
desktop = "/content/DSL-StrongPasswordData (1).csv"
try:
    df_desk = pd.read_csv(desktop)
except:
    print("Error loading CMU data. Check Path.")
    exit()

# Keeping first 41 users to match mobile
users_cmu = df_desk['subject'].unique()[:41]
df_desk = df_desk[df_desk['subject'].isin(users_cmu)]
X_d_all = df_desk.iloc[:, 3:].values
y_d_all = df_desk['subject'].values
print(f"Desktop Loaded: {len(X_d_all)} samples from {len(np.unique(y_d_all))} users.")
```

2.3.3 Pairing users from each dataset and Normalising the Data

```
# 3. PAIRING to stimulate one user using both devices
print("Pairing Users")
final_Xd, final_Xm, final_y = [], [], []

unique_mob = np.unique(y_m_all)
unique_desk = np.unique(y_d_all)
min_users = min(len(unique_mob), len(unique_desk))

for i in range(min_users):
    dm = X_m_all[y_m_all == unique_mob[i]]
    dd = X_d_all[y_d_all == unique_desk[i]]
    n = min(len(dm), len(dd))
    if n > 20:
        final_Xm.append(dm[:n])
        final_Xd.append(dd[:n])
        final_y.extend([unique_desk[i]] * n)

X_D = np.vstack(final_Xd)
X_M = np.vstack(final_Xm)
Y = np.array(final_y)

# Normalize
X_D = StandardScaler().fit_transform(X_D)
X_M = MinMaxScaler().fit_transform(X_M)

print(f"FINAL FUSION DATASET: {len(Y)} samples.")
```

3 Machine Learning Implementation

3.1 Training the Random Forest Classifier

```
# --- STEP 2: TRAINING & FUSION ---
print("\n--- STEP 2: RUNNING FUSION ---")

target = unique_desk[0]
y_bin = np.where(Y == target, 1, 0) # 1 = Target, 0 = Attacker

# Split
Xd_train, Xd_test, Xm_train, Xm_test, y_train, y_test = train_test_split(
    X_D, X_M, y_bin, test_size=0.2, stratify=y_bin, random_state=42
)

# Train RFs
rf_d = RandomForestClassifier(n_estimators=100).fit(Xd_train, y_train)
rf_m = RandomForestClassifier(n_estimators=100).fit(Xm_train, y_train)
```

3.2 Applying Fusion to the Individual Modalities

```
# Predict Probabilities
p_d = rf_d.predict_proba(Xd_test)[:, 1]
p_m = rf_m.predict_proba(Xm_test)[:, 1]
p_fused = (0.5 * p_d) + (0.5 * p_m)#applying the simple weighted sum rule

y_pred_d = rf_d.predict(Xd_test)
y_pred_m = rf_m.predict(Xm_test)
y_pred_f = np.where(p_fused > 0.5, 1, 0)

# Calculate EER
def get_eer(y_true, y_score):
    fpr, tpr, _ = roc_curve(y_true, y_score)
    eer = brentq(lambda x : 1. - x - interp1d(fpr, tpr)(x), 0., 1.)
    return eer, fpr, tpr

eer_d, fpr_d, tpr_d = get_eer(y_test, p_d)
eer_m, fpr_m, tpr_m = get_eer(y_test, p_m)
eer_f, fpr_f, tpr_f = get_eer(y_test, p_fused)

print(f"\nResults for Target {target}:")
print(f"Desktop EER: {eer_d:.4f}")
print(f"Mobile EER:  {eer_m:.4f}")
print(f"FUSED EER:   {eer_f:.4f} << IMPROVEMENT")
```

4 Visualisation

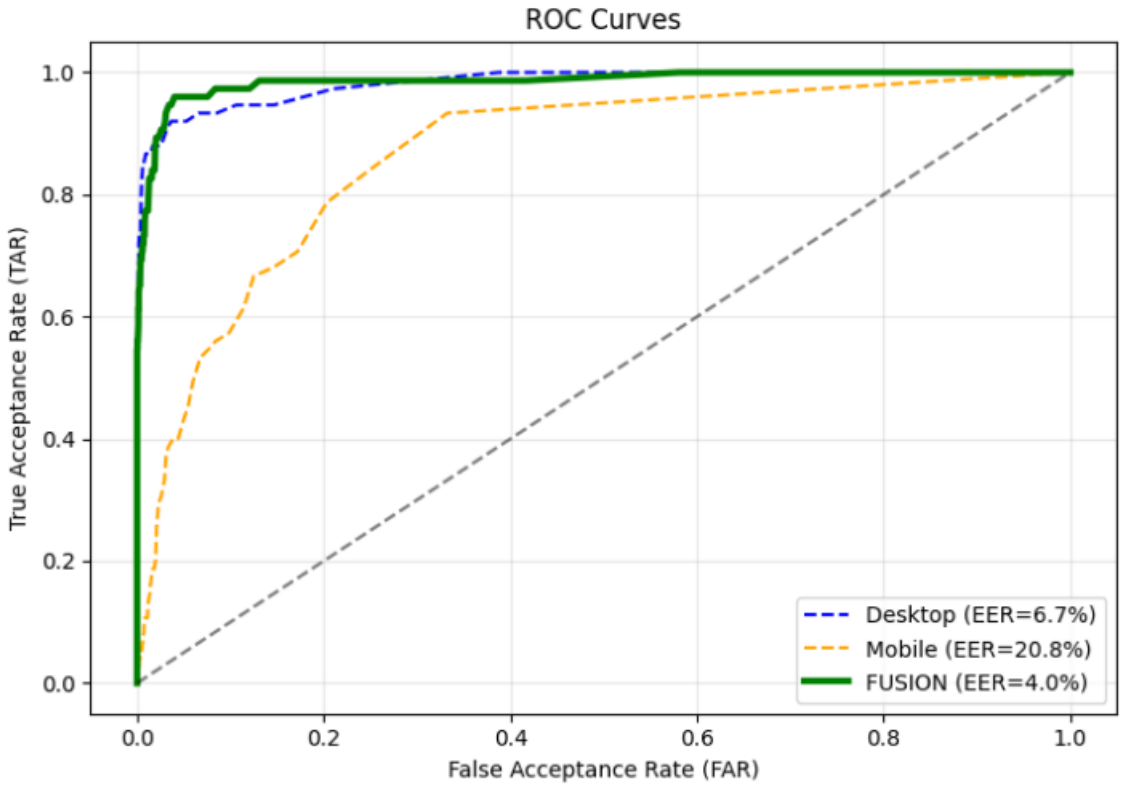
4.1 Plotting ROC and Confusion Matrixes for Each Modality

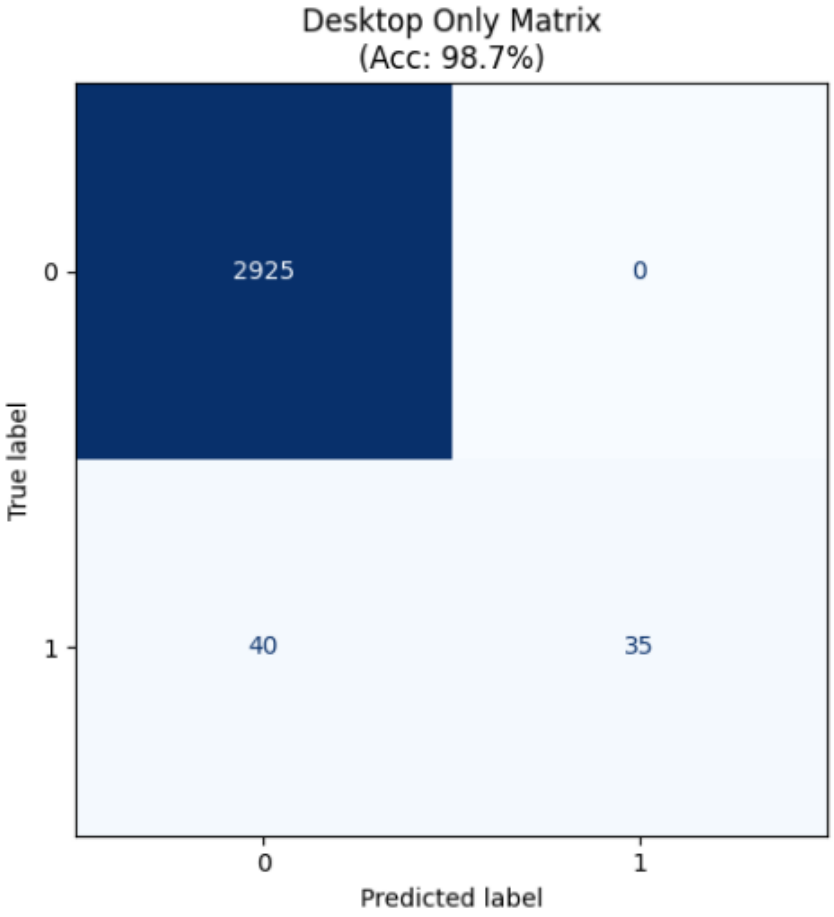
```
fig, axes = plt.subplots(2, 2, figsize=(14, 12))
plt.suptitle(f'Multimodal Authentication Results (Target: {target})', fontsize=16)


axes[0, 0].plot(fpr_d, tpr_d, linestyle='--', color='blue', label=f'Desktop (EER={eer_d:.1%})')
axes[0, 0].plot(fpr_m, tpr_m, linestyle='--', color='orange', label=f'Mobile (EER={eer_m:.1%})')
axes[0, 0].plot(fpr_f, tpr_f, color='green', linewidth=3, label=f'FUSION (EER={eer_f:.1%})')
axes[0, 0].plot([0, 1], [0, 1], 'k--', alpha=0.5)
axes[0, 0].set_title('ROC Curves')
axes[0, 0].set_xlabel('False Acceptance Rate (FAR)')
axes[0, 0].set_ylabel('True Acceptance Rate (TAR)')
axes[0, 0].legend()
axes[0, 0].grid(True, alpha=0.3)

# Plot 2: Desktop Confusion Matrix (Top Right)
ConfusionMatrixDisplay.from_predictions(y_test, y_pred_d, ax=axes[0, 1], cmap='Blues', colorbar=False)
axes[0, 1].set_title(f'Desktop Only Matrix\n(Acc: {np.mean(y_test==y_pred_d):.1%})')

# Plot 3: Mobile Confusion Matrix (Bottom Left)
ConfusionMatrixDisplay.from_predictions(y_test, y_pred_m, ax=axes[1, 0], cmap='Oranges', colorbar=False)
axes[1, 0].set_title(f'Mobile Only Matrix\n(Acc: {np.mean(y_test==y_pred_m):.1%})')

# Plot 4: Fusion Confusion Matrix (Bottom Right)
ConfusionMatrixDisplay.from_predictions(y_test, y_pred_f, ax=axes[1, 1], cmap='Greens', colorbar=False)
axes[1, 1].set_title(f'FUSION Matrix\n(Acc: {np.mean(y_test==y_pred_f):.1%})')

plt.tight_layout(rect=[0, 0.03, 1, 0.95])
plt.savefig("Thesis_Final_Dashboard.png")
print("Dashboard saved as 'Thesis_Final_Dashboard.png'")
plt.show()
```

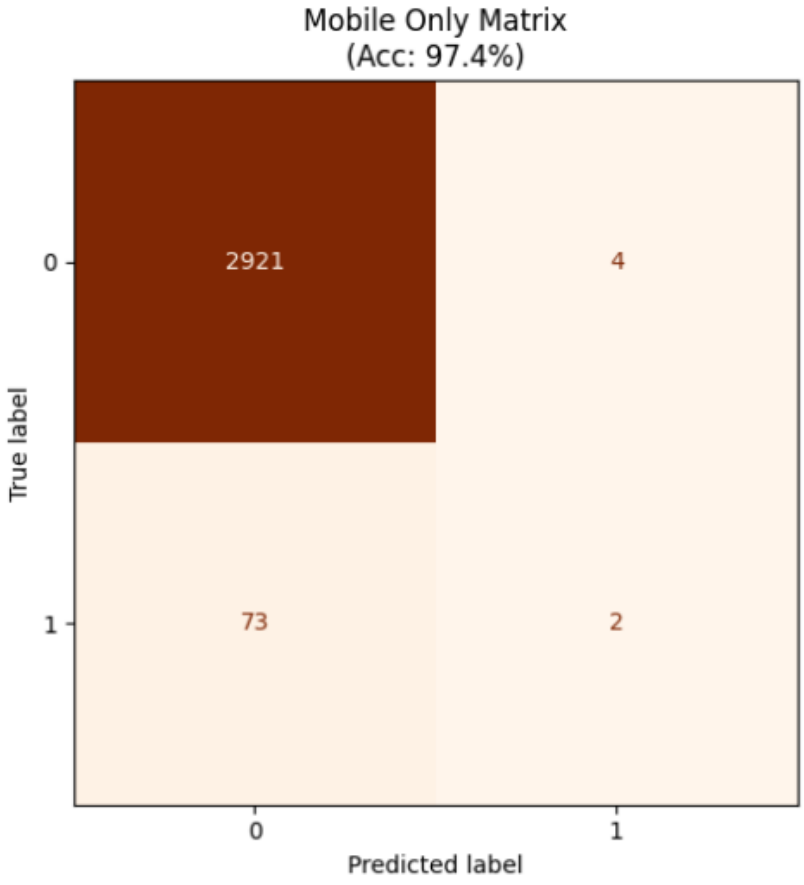
4.2 Outputs Achieved

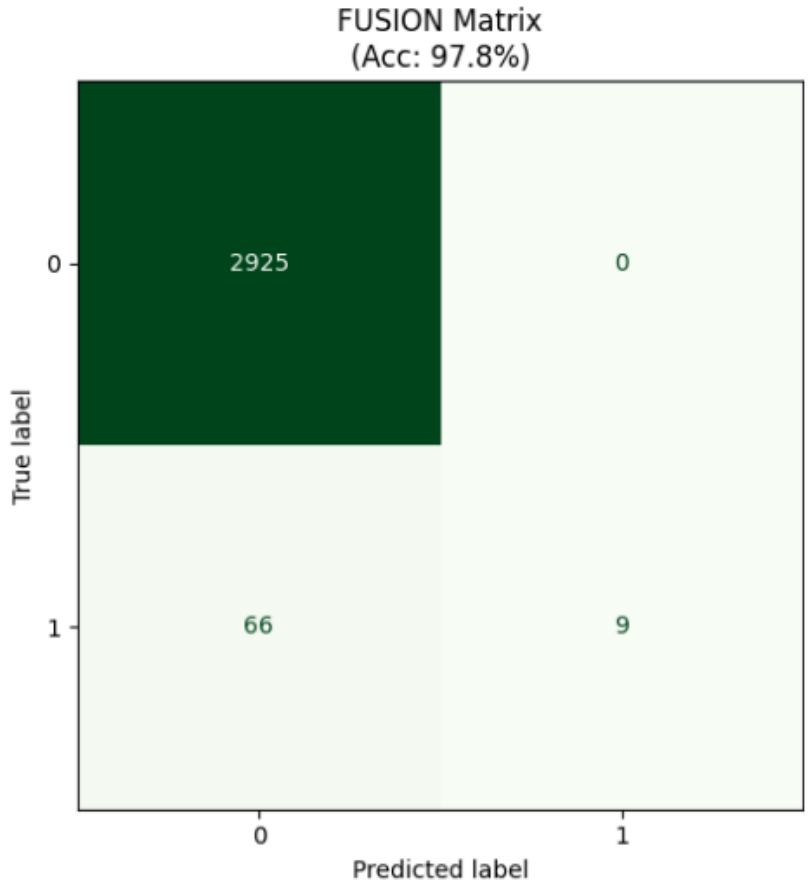
```
--- STEP 1: LOADING & PAIRING DATA ---
Mobile Loaded: 20682 strokes from 41 users.
Desktop Loaded: 16400 samples from 41 users.
Pairing Users...
FINAL FUSION DATASET: 14997 samples.
```

```
--- STEP 2: RUNNING FUSION ---

Results for Target s002:
Desktop EER: 0.0667
Mobile EER:  0.2078
FUSED EER:   0.0400 << IMPROVEMENT
```




Desktop Only Matrix
(Acc: 98.7%)
2925
0
40
35
0
1
True label
0
1
Predicted label


Mobile Only Matrix
(Acc: 97.4%)
2921
4
73
2
0
1
True label
0
1
Predicted label


FUSION Matrix
(Acc: 97.8%)
2925
0
66
9
0
1
True label
0
1
Predicted label