National College of Ireland

Configuration Manual

MSc Research Project
MSc Cybersecurity

Rona Shaji
Student ID: 23292709

School of Computing
National College of Ireland

Supervisor: Vikas Sahni

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Rona Shaji
Student ID:	23292709
Programme:	MSc Cybersecurity **Year:** 2024-2025
Module:	MSc Practicum
Lecturer:	Vikas Sahni
Submission Due Date:	15th September 2025
Project Title:	Improving API Security in the Cloud with AI/ML
Word Count:	862 **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: Rona Shaji

Date: 15/09/25

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

Rona Shaji
Student ID: 23292709

Project Title: Improving API Security in the Cloud with AI/ML
Environment: AWS EC2 (Ubuntu 24.04 LTS, t3.micro)
Main Components: Flask APIs, Random Forest, Autoencoder, JWT + Rate Limit + Regex Signatures

Tools and Technologies Used

Component	Tool/Service	Version
Programming Language	Python	3.12
Framework	Flask	3.03
ML Libraries	Scikit-learn , Keras , TensorFlow	1.5.2, 3.3.3, 2.17.0
Data Handling	Pandas , NumPy	2.2.2. 1.26.4
API Testing	curl , Postman , OWASP ZAP	8.9.1, v11.10, 2.15.0
Deployment	AWS EC2 (Ubuntu 24.04 LTS, t3.micro instance)	
Monitoring	htop , logging, manual timing	3.3.0
Security Add-ons	Flask-Limiter , PyJWT , regex-based pattern match (re module Python)	3.8.0, 2.9.0

EC2 INSTANCE SETUP

Launch EC2 Instance

- **AMI**: Ubuntu 24.04 LTS (64-bit)
- **Instance type**: t3.micro
- **Security Group Rules**:
  - Allow **SSH** (Port 22)
  - Allow **HTTP** (Port 80)
  - Allow **Custom TCP Rule** (Port 5000 or 8000 for API)

Connect to Instance via SSH

ssh -i "your_key.pem" ubuntu@your-ec2-public-ip

Python Environment Setup

```
sudo apt update && sudo apt install python3-venv python3-pip -y
python3 -m venv venv
source venv/bin/activate
```

Install Required Libraries

```
pip install flask scikit-learn pandas numpy keras tensorflow flask-limiter pyjwt gunicorn
```

PROJECT STRUCTURE

```
api_security_ml_project/
│
├── app/                    # API logic
│   ├── api.py              # Flask app
│   ├── utils.py   # Signature + JWT + Rate Limiter
│   ├── rf_predictor.py       # Random Forest logic
│   └── autoencoder_predictor.py  # Autoencoder logic
│
├── models/
│   ├── rf_model.joblib
│   └── autoencoder_model.h5
│
├── data/
│   ├── api_logs.csv
│   └── labeled_api_logs.csv
│
├── test_payloads/
│   ├── normal_login.json
│   ├── malicious_delete_user.json
│   ├── large_payload_ddos.json
│   └── xss_payload.json
│
├── evaluation_summary.txt
├── detected_anomalies.json
└── requirements.txt
```

MODEL TRAINING (Optional if models already saved)

Random Forest

```
from sklearn.ensemble import RandomForestClassifier
from sklearn.preprocessing import LabelEncoder
import pandas as pd, joblib

df = pd.read_csv("data/labeled_api_logs.csv")
X = df[['endpoint', 'method', 'payload_size', 'status_code']]
y = df['label']
```

```
# Encode categorical
for col in ['endpoint', 'method']:
    X[col] = LabelEncoder().fit_transform(X[col])

rf = RandomForestClassifier(n_estimators=100)
rf.fit(X, y)

joblib.dump(rf, 'models/rf_model.joblib')
```

Autoencoder (Keras)

```
from keras.models import Sequential
from keras.layers import Dense
import numpy as np

X_train = np.loadtxt('data/normalized_cicids.csv', delimiter=',')

model = Sequential([
    Dense(64, activation='relu', input_shape=(X_train.shape[1],)),
    Dense(32, activation='relu'),
    Dense(64, activation='relu'),
    Dense(X_train.shape[1])
])
model.compile(optimizer='adam', loss='mse')
model.fit(X_train, X_train, epochs=20, batch_size=32)

model.save('models/autoencoder_model.h5')
```

RUNNING THE APPLICATION

Start the Flask API Server

```
gunicorn -b 127.0.0.1:8000 app.api:app
```

If using Nginx as a reverse proxy, map port 80 to 8000 internally.

TESTING VIA CURL

Traditional Detection

Valid Request:

```
curl -X POST http:// your-ec2-ip /traditional_check \
-H "Content-Type: application/json" \
-d @test_payloads/normal_login.json
```

Missing/Invalid Token:

```
curl -X POST http:// your-ec2-ip /traditional_check \ -H "Content-Type: application/json" \ -d
'{"user_id": "123", "endpoint": "/data", "method": "GET"}'
```

XSS Attack:

```
curl -X POST http:// your-ec2-ip /traditional_check \
 -H "Content-Type: application/json" \
 -d '{"payload": "<script>alert(\"XSS\")</script>", "token_used":
"valid_token_123"}'
```

Random Forest Endpoint

Normal:

```
curl -X POST http:// your-ec2-ip /predict \ -H "Content-Type: application/json" \ -d
@test_payloads/normal_login.json
```

Malicious Delete User Request:

```
curl -X POST http://your-ec2-ip/predict \
-H "Content-Type: application/json" \
-d @test_payloads/malicious_delete_user.json
```

Large Payload DDoS Simulation :

```
curl -X POST http:// your-ec2-ip /predict \ -H "Content-Type: application/json" \ -d
@test_payloads/large_payload_ddos.json
```

Autoencoder Detection

```
curl -X POST http://your-ec2-ip/detect_autoencoder \
-H "Content-Type: application/json" \
-d @test_payloads/large_payload_ddos.json
```

MONITORING & LOGGING

View Logs

```
tail -f data/api_logs.csv
```

System Monitoring

```
htop   # Monitor CPU/RAM
```

COMMON ISSUES & FIXES

Problem	Fix
Autoencoder crash on EC2	Reduce dataset size or use t3.medium instance
RF model misclassifying new endpoints	Retrain with more diverse data
No response from API	Check gunicorn port binding or Nginx proxy config
Memory overflow	Use batch processing or split data during inference

PACKAGING FOR DEPLOYMENT

- Zip the following folders and send to client/supervisor:
  - /models/
  - /test_payloads/
  - /app/
  - run_server.sh (startup script)

Example run_server.sh:

```
#!/bin/bash
source venv/bin/activate
gunicorn -b 127.0.0.1:8000 app.api:app
```