

Individual Reflective

Secure Systems Architecture module covered a wide range of beneficial topics like:

- Introduction to Operating Systems and Secure Systems Architecture
- Modelling and Socket Programming
- Systems Engineering and Modelling
- The Scientific Method and its Role in Modelling Distributed Services and Systems
- Current and Future Challenges of Operating Systems and Distributed Systems
- Security Testing of Distributed Systems

Attached is the document "Evidence" which displays screenshots of my module contributions i.e., Initial post, PowerPoint presentation, project's python scripts, etc.

Additionally, I had the opportunity to collaborate on a group project with three of my peers. Our selected case study, (Kodali, et al., 2016), focused on a low-cost system serving as a smart home security and home automation system. Our task was to create a detailed proposal/design document (in unit 3), that would guide the development of the system.

To ensure project's success, we appointed a group leader and agreed on matters like: communication methods (Google Drive, Zoom, WhatsApp, emails, GitHub), work distribution, meeting schedules, etc.

The first part of the project, design document creation, I identified the case study risks, potential vulnerabilities, mitigations, and designed UML diagrams. Evidence doc: Figures 3-10, is a compilation of my findings from sources such as Abdullah et al.

(2019), Anand et al. (2020), etc. This increased my understanding of home security and automation systems, including IoT devices, and their potential risks, such as dependence on Wi-Fi, susceptibility to Denial-of-Sleep (DoSL) attacks, the effectiveness of bandwidth limitations, among others.

I faced a challenge running an AD tool with Java. During one of our discussions, I requested help from my team and one of them remotely guided me using TeamViewer.

The second part of the project, we created a python prototype system that demonstrated the functioning and interaction between a simulated device and a simulated controller, so as to investigate one of our hypothesis questions.

We faced a challenge due to our lack of coding experience, so we decided to first research on it and held regular meetings (every 1-3 days) to share our findings.

Initially, we opted to use Django to design an application with 2-Factor Authentication login, user registration and authorisation. After my peer completed the Django app, I created a Docker image and container, and scanned it (Evidence doc: Figures 11-15).

However, after further research on IoT devices, since our app was not simulating any of the home automation controls, I discovered that paho-mqtt open-source client libraries for Python were more suitable for machine-to-machine communications (PyPi, N.D.). I proposed this to my group and after their own research, we agreed to create a light control prototype using paho-mqtt and then dockerized it later.

I designed a motion-simulated light control and performed manual and automatic tests using pytest (Evidence doc: Figure 16), while my peers worked on other simulations, security features, testing, dockerization, and documentation.

Although we did not investigate my hypothesis question “*does the motion sensor device and light control system function collaboratively to ensure secure communication and operation of the system?*”, Figures 17-18 in the Evidence document show a successful connection return code ($rc=0$) using the MQTT protocol and a broker to ensure reliable and secure message delivery. Additionally, the use of JSON encoding ensures easy readability and understanding of the messages between the devices (Steve, 2022).

However, we chose to investigate my peer's hypothesis on authentication because it provided in-depth analysis of one of the security features.

In terms of the development cycle, we followed an Agile development methodology and I believe that we were able to work on the project in an organized, flexible way and stayed on track and met our deadlines. At the same time, we could have used our time well by not focusing too much on one task. While in testing, we were able to perform both manual and automated tests on the connectivity, subscription, security, etc, - of the simulation so as to ensure that it was functioning as expected.

Some of the biggest challenges that we faced as a team were time constraints and the lack of enough learning materials on paho-mqtt on a Windows environment, therefore making it difficult to successfully implement some of the security features including the ones we had planned to implement in the project.

Despite of these difficulties, our collaborative and hardworking group was able to implement key features, resulting in a functional project submission.

Generally, the whole experience has been very knowledgeable and eye-opening. Moreover, I recognized the importance of good communication and collaborating with team members so as to ensure that we were all working towards the same goals.

Moving forward, I will use the knowledge and experience gained from this project to continue improving my technical and communication skills. I will also apply the lessons learned from this experience to future projects, ensuring that I can contribute effectively to development teams and deliver high-quality products that meet the set requirements.

References

- Abdullah, T., Ali, W., Malebary, S. & Ahmed, A. A., 2019. A Review of Cyber Security Challenges, Attacks and Solutions for Internet of Things Based Smart Home. *International Journal of Computer Science and Network Security (IJCSNS)*, 19(9), pp. 139-146.
- Anand, P. et al., 2020. IoT Vulnerability Assessment for Sustainable Computing: Threats, Current Solutions, and Open Challenges. *IEEE Access*, Volume 8, pp. 168825-168853.
- Borgini, J., 2021. *Tackle IoT application security threats and vulnerabilities*. [Online] Available at: <https://www.techtarget.com/iotagenda/tip/Tackle-IoT-application-security-threats-and-vulnerabilities> [Accessed 2 February 2023].

Kodali, R. K., Jain, V., Bose, S. & Boppana, L., 2016. *IoT based smart security and home automation system*. Greater Nodia, IEEE.

PyPi, N.D.. *paho-mqtt 1.6.1*. [Online]

Available at: <https://pypi.org/project/paho-mqtt/#:~:text=This%20code%20provides%20a%20client,.9%2B%20or%203.6%2B>
[Accessed 26 February 2023].

Steve, 2022. *Python MQTT Client Connections– Working with Connections*. [Online]

Available at: <http://www.steves-internet-guide.com/client-connections-python-mqtt/>
[Accessed 10 February 2023].

Screenshots of my Contributions in this Module

Figure 1: PowerPoint Presentation on Symbian Operating System

Slide 1: Symbian Operating System
By Beatrice Mutegi

Slide 2: Questions to be tackled are:
With your selected/ assigned OS, answer the following questions:
■ Who created the OS? When was it created? What was its purpose (i.e., desktop, server, appliance, IoT, etc.)?
■ What was its unique feature/ characteristic/ contribution? Is it still relevant today?
■ Is the OS still in use or has it been superseded? What replaced it?
The provided link is a starting point – you should find at LEAST one more link to support your answers.
Please create a PowerPoint presentation consisting of a max of 4 slides that answer the questions above.
Be prepared to share and discuss your slides in this week's seminar.

Slide 3: Who created the OS? When was it created? What was its purpose (i.e., desktop, server, appliance, IoT, etc.)?
■ Symbian is a mobile operating system (OS) that was created by Symbian Ltd. in 1998.
■ The company was a joint venture between Ericsson, Nokia, Motorola, and Psion (Symbian, 2020).
■ The purpose of this OS was to provide a common platform for mobile devices, such as smartphones and PDAs.
■ Symbian was the most widely used mobile OS in the world until the rise of Android and iOS in the late 2000s.
■ According to a report by Gartner (2011), Symbian OS had a market share of 48.8% in 2010, which fell to 11.4% in 2011.

Slide 4: What was its unique feature/ characteristic/ contribution?
■ The unique feature of Symbian was its open-source nature. Unlike other mobile OSs of its time, such as Windows Mobile and BlackBerry OS, Symbian was an open-source and had a large developer community. This allowed for a wide range of third-party applications to be developed for the platform, which was one of the key factors in its success.
■ Symbian also had a strong focus on security and reliability, making it a popular choice for enterprise users. According to a report by Symbian Foundation (2010), Symbian had a user base of over 400 million users globally in 2010.

Slide 5: Is it still relevant today?
Is the OS still in use or has it been superseded?
What replaced it?
■ In terms of relevancy, Symbian is no longer a major player in the mobile OS market.

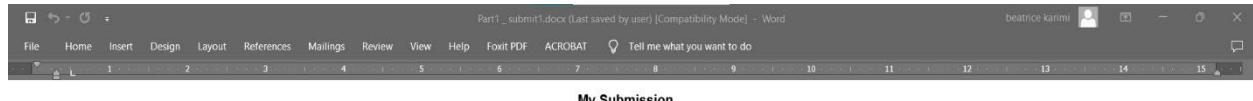
Slide 6: References
■ Gartner. (2011). Gartner Says Symbian OS will Slip to 11.4% Market Share in 2011. Retrieved from <https://www.gartner.com/en/newsroom/press-releases/2011-07-27-gartner-says-symbian-os-will-slip-to-11-4-market-share-in-2011>.

Slide 4 of 6 French (France) Accessibility: Good to go

Figure 2: Initial Post

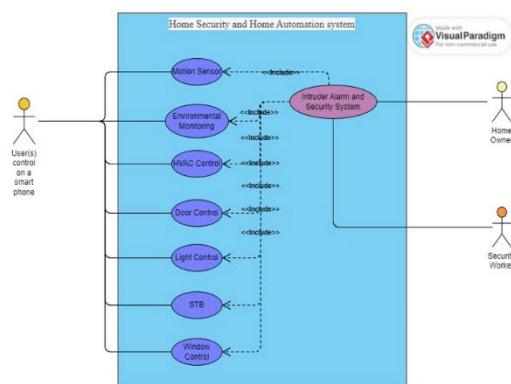
The screenshot shows a web-based forum interface for the University of Essex. At the top, there's a navigation bar with links for 'University of Essex', 'Online', 'My Modules', and user profile information for 'Beatrice Mutegi'. Below the navigation is the title 'Secure Systems Architecture January 2023'. The main content area displays a post titled 'Initial Post' by Beatrice Mutegi on Wednesday, 1 February 2023, at 3:28 AM. The post discusses SysML as a general-purpose language for modeling system architectures. It includes a bulleted list comparing SysML and UML, mentioning SysML's focus on hardware and software components. Below the post, there's a section titled 'References' with a link to a journal article by Chabibi et al. (2018) and a link to the SysML Open Source Project. At the bottom of the post area, there are buttons for 'Permalink' and 'Reply'. A small 'Knowledge Base' button is visible in the top right corner of the post area.

Figure 3: My Contribution to 1st Part of the Group Assessment (A)



Introduction

The (Kodali, et al., 2016) case study provides an overview of a low-cost system that serves as a smart home security and home automation (as seen on the [case figure](#) below).



Below is a table showing the current features of the system that makes it to be vulnerable and the mitigations that can be applied (as referenced from (Touqueer, et al., 2021), (Borgini, 2021), (Apriorit, 2022), (Anand, et al., 2020), (Abdullah, et al., 2019))



Figure 4: My Contribution to 1st Part of the Group Assessment (B)

Features of the Current System	Risks Accompanied	Potential Vulnerabilities	Possible Mitigations
It relies solely on digits on the phone's keypad to access the security system <ul style="list-style-type: none"> • Unauthorized access. • Spoofing • Man-in-the-middle Attacks • Installation of malicious software • Fines and lawsuits that could lead to damaged reputations, bankruptcy and losses 	<ul style="list-style-type: none"> • Unauthorized access. • Spoofing • Man-in-the-middle Attacks • Installation of malicious software • Fines and lawsuits that could lead to damaged reputations, bankruptcy and losses 	<ul style="list-style-type: none"> • Lack of Multi-Factor Authentication • Lack of authorization • Unencrypted communication • Not enough security enforcing features • Lack of data privacy and certified compliances like GDPR, ISO 27001, ISO 27017, ISO 	<ul style="list-style-type: none"> • Multi-Factor Authentication • Implement changing of passwords • Implement complex passwords • Limit number of log-in attempts • User Access controls • Authorizations • Session management • Implement data

Figure 5: My Contribution to 1st Part of the Group Assessment (C)

		27017, ISO 27018, etc	• Implement data privacy
The system's functionality is dependent on the Wi-Fi connection only,	<ul style="list-style-type: none"> • Wi-Fi dependency • Network attack • Denial-of-Service (DoS) and Denial-of-Sleep (DoSL) attacks 	<ul style="list-style-type: none"> • System is down and security is compromised once Wi-Fi connection is lost or weak • Insecure network • Unencrypted communication 	<ul style="list-style-type: none"> • Set-up other system connectivity e.g., Local Area Connection • Firewalls like Next-generation firewall • Limit device or network bandwidth • Backup connectivity options like 4G or 3G, to ensure that the system remains operational even if the Wi-Fi connection is lost.

Figure 6: My Contribution to 1st Part of the Group Assessment (D)

			• Intrusion Detection and Prevention Systems • Implementation of secure socket layer (SSL) Certificates, • Data Encryption • Network segmentation
Lack of security tests that make room for the system's improvements	<ul style="list-style-type: none"> • More prone to breaches 	<ul style="list-style-type: none"> • Lack of security tests and scanning 	<ul style="list-style-type: none"> • Regular security and backup testing, and scanning for threats helps in reinforcing the system
Lack of data storage security	<ul style="list-style-type: none"> • Injection attacks • Tampering 	<ul style="list-style-type: none"> • Unsecure data storage 	<ul style="list-style-type: none"> • Secure databases • Antivirus

Figure 7: My Contribution to 1st Part of the Group Assessment (E)

			<ul style="list-style-type: none"> • Data encryption
Lack of Security Updates	<ul style="list-style-type: none"> • More prone to breaches 	<ul style="list-style-type: none"> • Lack of Security Updates and patches 	<ul style="list-style-type: none"> • Regular and automatic System and hardware updates
Unsecured device management	<ul style="list-style-type: none"> • Unauthorised factory-resetting of devices • Installation of malicious software and updates • Software and firmware risks and attacks 	<ul style="list-style-type: none"> • Malicious software updates • Device breaches • Weak firmware or software, servers, backend application 	<ul style="list-style-type: none"> • Use of secure updating mechanisms like digital signatures • Practising secure Programming practices • System centralization • Implementing secure device management protocols

Figure 8: My Contribution to 1st Part of the Group Assessment (F)

			<ul style="list-style-type: none"> • Limiting the number of device management access points • Ensure tamper-resistant hardware
Human Error	<ul style="list-style-type: none"> • Breaches • Social engineering 	<ul style="list-style-type: none"> • Human errors 	<ul style="list-style-type: none"> • Cybersecurity training on users

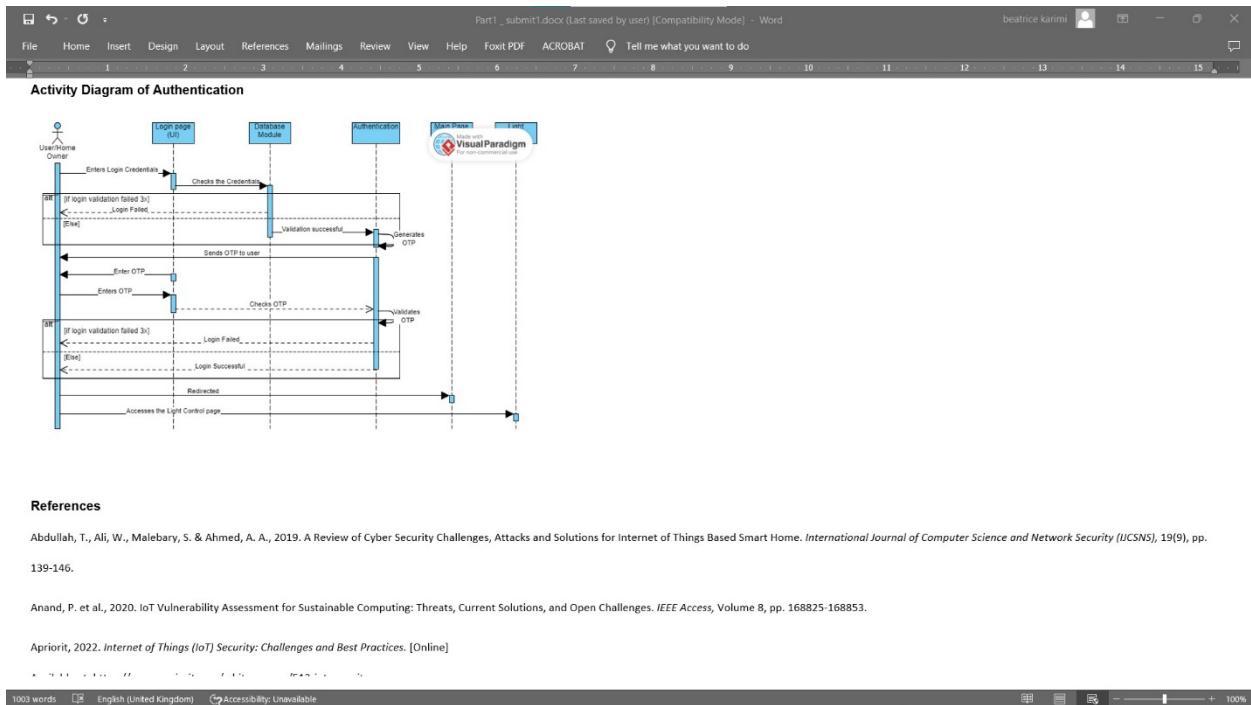
Scrum -Sprint 1: with the use of Python language

1. Implement a user interface that will centralize the system
2. Implement Multi-Factor Authorization
3. Implement change of password
4. Validation of complex passwords
5. Access control and Authorization
6. Session Management
7. Prove the chosen *thesis question* by performing tests
8. Cookies and certificates csrf token

Activity Diagram of Authentication



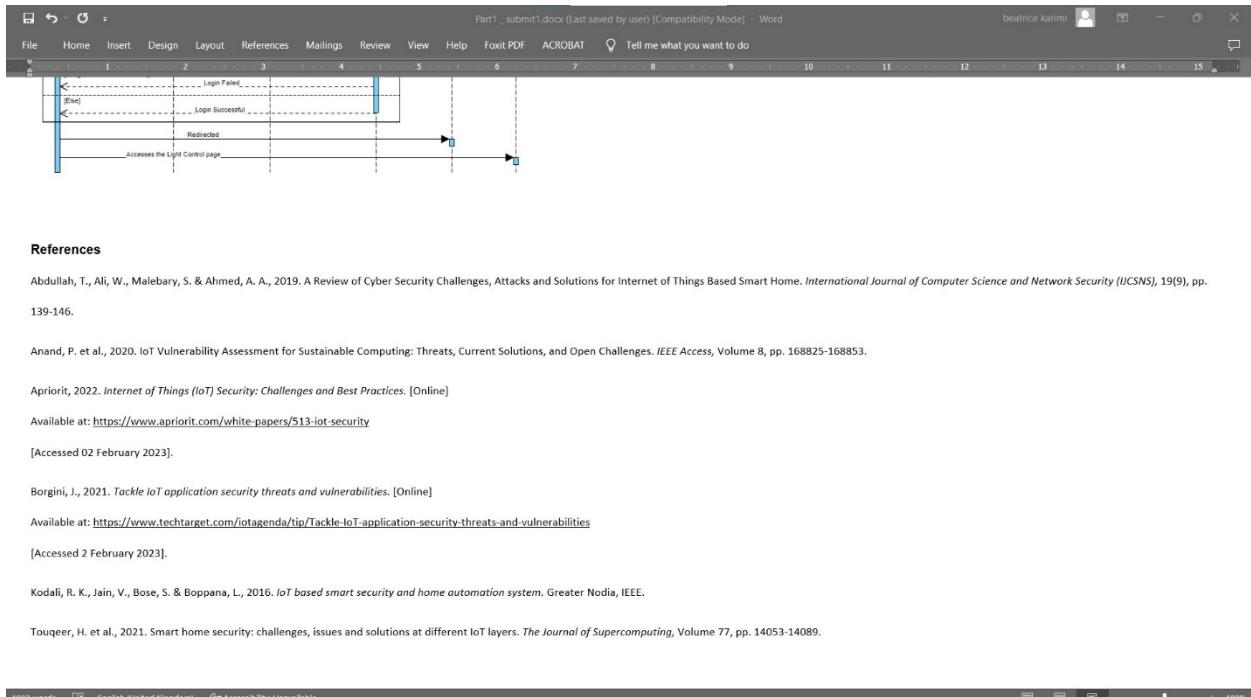
Figure 9: My Contribution to 1st Part of the Group Assessment (G)



References

- Abdullah, T., Ali, W., Malebary, S. & Ahmed, A. A., 2019. A Review of Cyber Security Challenges, Attacks and Solutions for Internet of Things Based Smart Home. *International Journal of Computer Science and Network Security (IJCNS)*, 19(9), pp. 139-146.
- Anand, P. et al., 2020. IoT Vulnerability Assessment for Sustainable Computing: Threats, Current Solutions, and Open Challenges. *IEEE Access*, Volume 8, pp. 168825-168853.
- Apriorit, 2022. *Internet of Things (IoT) Security: Challenges and Best Practices*. [Online]

Figure 10: My Contribution to 1st Part of the Group Assessment (H)



References

- Abdullah, T., Ali, W., Malebary, S. & Ahmed, A. A., 2019. A Review of Cyber Security Challenges, Attacks and Solutions for Internet of Things Based Smart Home. *International Journal of Computer Science and Network Security (IJCNS)*, 19(9), pp. 139-146.
- Anand, P. et al., 2020. IoT Vulnerability Assessment for Sustainable Computing: Threats, Current Solutions, and Open Challenges. *IEEE Access*, Volume 8, pp. 168825-168853.
- Apriorit, 2022. *Internet of Things (IoT) Security: Challenges and Best Practices*. [Online]
- Available at: <https://www.apriorit.com/white-papers/513-iot-security>
- [Accessed 02 February 2023].
- Borgini, J., 2021. *Tackle IoT application security threats and vulnerabilities*. [Online]
- Available at: <https://www.techtarget.com/iotagenda/tip/Tackle-IoT-application-security-threats-and-vulnerabilities>
- [Accessed 2 February 2023].
- Kodali, R. K., Jain, V., Bose, S. & Boppana, L., 2016. *IoT based smart security and home automation system*. Greater Noida, IEEE.
- Touqeer, H. et al., 2021. Smart home security: challenges, issues and solutions at different IoT layers. *The Journal of Supercomputing*, Volume 77, pp. 14053-14089.

Figure 11: VS Code when Dockerizing the Django App

The screenshot shows the Visual Studio Code interface with the following details:

- Explorer View:** Shows the project structure with files like .dockerignore, dockerfile, README.md, settings.py, views.py, and docker-compose.yml.
- Dockerfile Content:**

```

1 FROM python:3.11
2
3 # setup environment variable
4 #ENV code=/home/app/webapp
5
6 # set work directory
7 RUN mkdir -p /code
8
9 # where your code lives
10 WORKDIR /code
11
12 # set environment variables
13 ENV PYTHONUNBUFFERED=1
14 ENV PYTHONIOENCODING=UTF-8
15
16 # install dependencies
17 #RUN pip install pipenv
18 #RUN pipenv install django
19 RUN pip install --upgrade pip
20 RUN pip install djangorestframework
21 RUN python -m pip install --upgrade setuptools pip wheel
22
23 # copy whole project to your docker home directory.
24 COPY . .
25
26 COPY req.txt .
27

```
- docker-compose.yml Content:**

```

version: '3.11'
services:
  app:
    build: .
    volumes:
      - ./django:/code
    ports:
      - "8000:8000"
    image: app:django
    container_name: djangoGrp2_container
    command: python manage.py runserver 0.0.0.0:8000

```
- Terminal Tab:** Shows the output of commands related to Docker container creation and Django server startup.
- Output Tab:** Displays logs from the Django application.
- Status Bar:** Shows the current workspace status including RAM usage, CPU usage, and connection to the Docker Hub.

Figure 12: Docker Images of the Django App and its Snyk Test

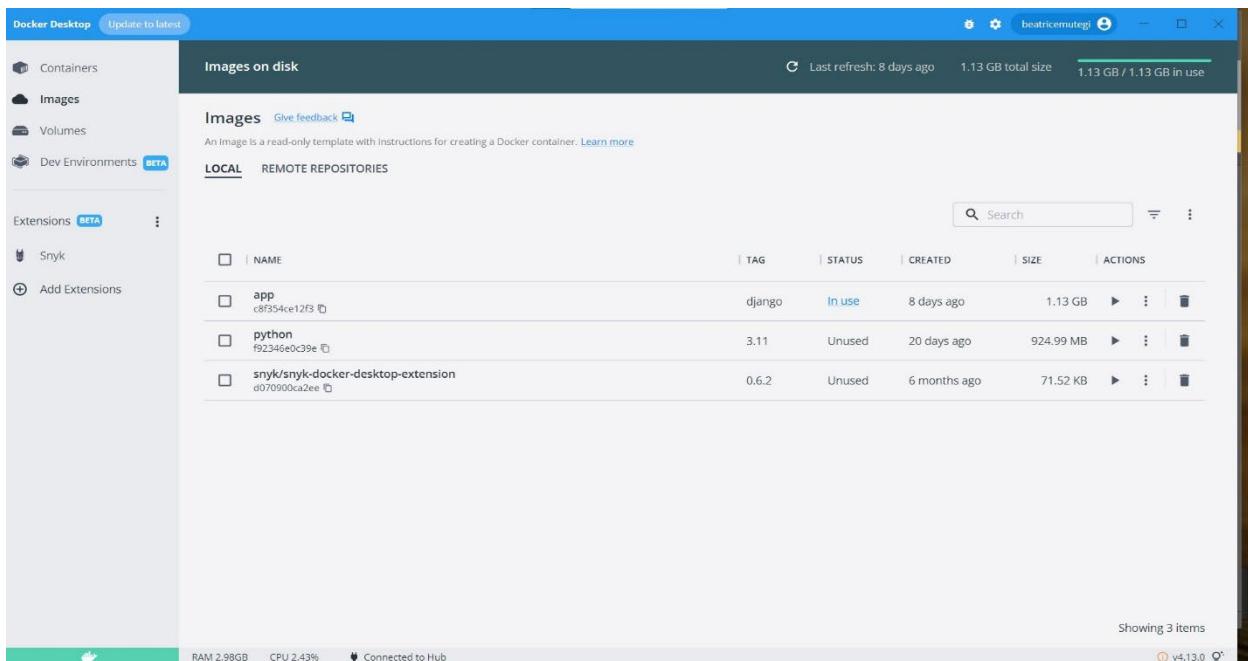


Figure 13:The Snyk Test Results of the Django App's Docker Image

The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** Shows the project structure with files like .dockerignore, dockerfile, docker-compose.yml, README.md, settings.py, views.py, and various Python files.
- Code Editor:** Displays the `dockerfile` content:

```
FROM python:3.11
# setup environment variable
#ENV code=/home/app/webapp
# set work directory
#RUN mkdir -p /code
```
- Terminal:** Shows the command PS C:\Users\biotronics\Desktop\Code\Code> followed by the Snyk test output.
- Output:** Shows the Snyk test results, including a critical severity vulnerability found in `aom/libaom`.
- Problems:** Lists 276 issues found across 276 vulnerabilities in the base image `python:3.11.2-bullseye`.
- Recommendations:** Lists alternative image types for upgrading the base image.

Figure 14: Docker Container of the Django App

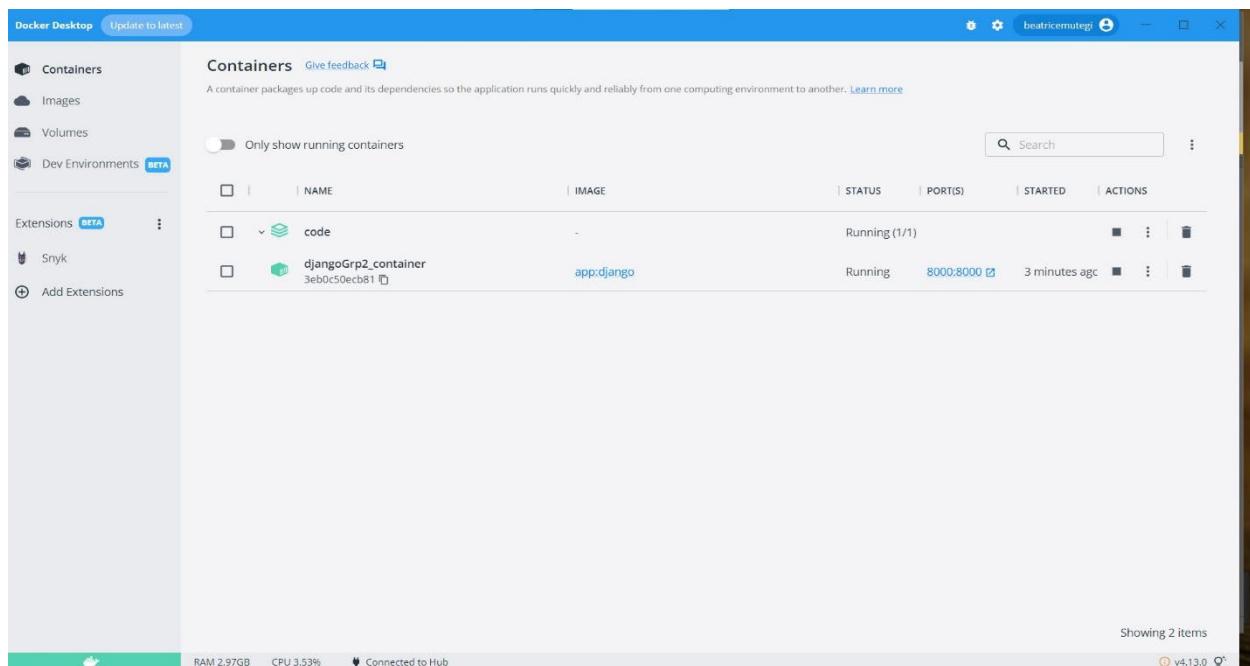


Figure 15: Running the Django App's Docker Container

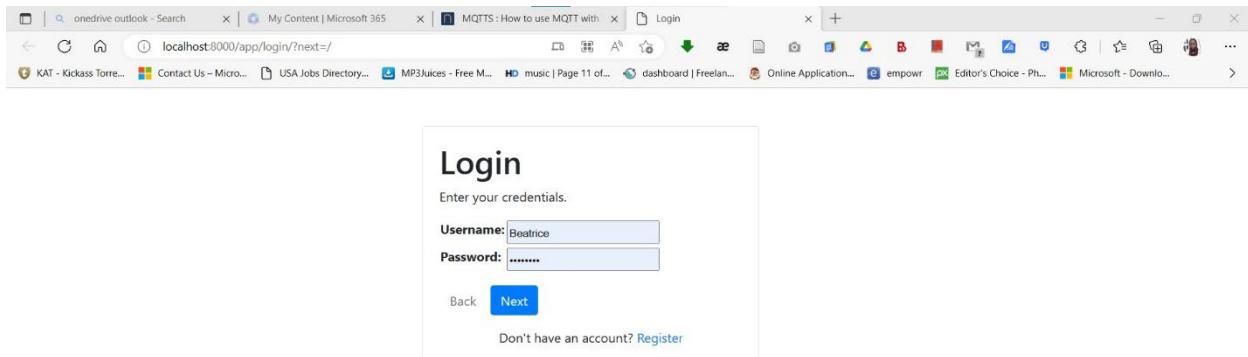


Figure 16: MQTT VS Code of the light control by motion folder.

Inclusive of the files: lightControl.py, motionDevice.py, running a pytest, mosquitto.config, password and the docker files.

```

motionDevice.py - Light control by motion - Visual Studio Code [Administrator]
File Edit Selection View Go Run Terminal Help
lightControl.py | motionDevice.py | ...
lightControl.py ...
# This Python script uses the MQTT (Message Queuing Telemetry Trans
# The script consists of two parts: one for the motion sensor devic
# This is the second part of the script (lightControl.py), which run
# Import libraries
# Define the on_connect and on_message functions for the MQTT clien
def on_connect(client, userdata, flags, rc):
    print("Connected to broker with result code "+str(rc))
    # Subscribe to the "motion_sensor" and "light_control" topics
    client.subscribe("motion_sensor")
    client.subscribe("light_control")
    def on_message(client, userdata, message):
        # Extract the topic and payload from the received message
        global light_state
        topic = message.topic
        payload = message.payload.decode()
        print("Received message: "+payload+" on topic: "+topic)
        if topic == "light_control" and payload == "on":
            light_state = "on"
        elif topic == "light_control" and payload == "off":
            light_state = "off"
        # Publish the current light state back to the "light_control" topic
        client.publish("light_control", light_state)
    # Assign the on_connect and on_message functions to the client
    client.on_connect = on_connect
    client.on_message = on_message
client.loop_forever()

motionDevice.py ...
# This Python script uses the MQTT (Message Queuing Telemetry Trans
# The script consists of two parts: one for the motion sensor devic
# This is the first part of the script (motionDevice.py), which run
# Import of the libraries
# Define the on_connect and on_publish functions for the MQTT clien
def on_connect(client, userdata, flags, rc): #rc (return code) is u
    print("Connected to broker with result code "+str(rc))
    # Create MQTT client instance
    client = mqtt.Client()
    # Assign the on_connect and on_publish functions to it
    client.on_connect = on_connect
    client.on_publish = on_publish
    client.connect("localhost", 1883, 60)
    client.publish("motion_sensor", "Device_001")
    client.publish("light_control", "motion_sensor")
    client.loop_forever()

lightControl.py ...
# This Python script uses the MQTT (Message Queuing Telemetry Trans
# The script consists of two parts: one for the motion sensor devic
# This is the second part of the script (lightControl.py), which run
# Import libraries
# Define the on_connect and on_message functions for the MQTT clien
def on_connect(client, userdata, flags, rc):
    print("Connected to broker with result code "+str(rc))
    # Subscribe to the "motion_sensor" and "light_control" topics
    client.subscribe("motion_sensor")
    client.subscribe("light_control")
    def on_message(client, userdata, message):
        # Extract the topic and payload from the received message
        global light_state
        topic = message.topic
        payload = message.payload.decode()
        print("Received message: "+payload+" on topic: "+topic)
        if topic == "light_control" and payload == "on":
            light_state = "on"
        elif topic == "light_control" and payload == "off":
            light_state = "off"
        # Publish the current light state back to the "light_control" topic
        client.publish("light_control", light_state)
    # Assign the on_connect and on_message functions to the client
    client.on_connect = on_connect
    client.on_message = on_message
client.loop_forever()

RECENT FILES: motionDevice.py, lightControl.py, mosquito.config, passwd, README.md, settings.json, cert, config, mosquito.config, passwd, requirements.txt, dockerignore, docker-compose.yml, dockerfile, lightControl.py, motionDevice.py, readme.md, requirements.txt

```

Figure 17: Running motionDevice.py script

```

motionDevice.py - Light control by motion - Visual Studio Code [Administrator]

File Edit Selection View Go Run Terminal Help motionDevice.py - Light control by motion - Visual Studio Code [Administrator]
EXPLORER ... mosquito.config lightControl.py docker-compose.yml 2 readme.md ...
LIGHT CONTROL BY MOT...
> idea
> v
> .gitignore
> CACHEDIR.TAG
> README.md
> .vscode
> settings.json
> certs
> config
> mosquito.config
> passwd
> venv
> .dockerignore
> docker-compose... 2
> dockerfile
> lightControl.py
> motionDevice.py
> readme.md
> requirements.txt

1 # This Python script uses the MQTT (Message Queuing Telemetry Trans...
2 # The script consists of two parts: one for the motion sensor device...
3
4 # This is the second part of the script (lightControl.py), which run...
5
6
7
8 # Import libraries
9 import paho.mqtt.client as mqtt #for MQTT protocol
10 import json #to convert the python dictionary into a JSON string th...
11
12 # Set up the initial state of the light
13 light_state = "off"
14
15 # Define the on_connect and on message functions for the MQTT client
16 def on_connect(client, userdata, flags, rc):
17     print("connected to broker with result code "+str(rc))
18
19     # Subscribe to the "motion_sensor" and "light_control" topics
20     client.subscribe("motion_sensor")
21     client.subscribe("light_control")
22
23     def on_message(client, userdata, message):
24         # Extract the topic and payload from the received message
25         global light_state
26         topic = message.topic
27         if topic == "motion_sensor":
28             if message.payload.decode() == "1":
29                 light_state = "on"
30             else:
31                 light_state = "off"
32         elif topic == "light_control":
33             if message.payload.decode() == "1":
34                 client.publish("motion_sensor", "1")
35             else:
36                 client.publish("motion_sensor", "0")
37
38     client.on_message = on_message
39
40     client.loop_forever()
41
42     # Publish messages
43     client.publish("light_control", "1")
44
45     # Disconnect from the broker
46     client.disconnect()

motionDevice.py ...
1 # This Python script uses the MQTT (Message Queuing Telemetry Trans...
2 # The script consists of two parts: one for the motion sensor device...
3
4 # This is the first part of the script (motionDevice.py), which run...
5
6
7
8 # Import of the libraries
9 import paho.mqtt.client as mqtt #for MQTT protocol
10 import time #to simulate IoT delays
11 import random #to create random id
12 import json #to convert the python dictionary into a JSON string th...
13
14 # Set up the device's ID and type
15 device_id = "Device_001"
16 device_type = "motion_sensor"
17
18 # Define the on_connect and on_publish functions for the MQTT client
19 def on_connect(client, userdata, flags, rc): rc (return code) is u...
20 | print("connected to broker with result code "+str(rc))
21
22 def on_publish(client, userdata, mid): #mid value is an integer tha...
23 | print("Message published with mid "+str(mid))

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL SQL CONSOLE AZURE
powershell
cmd
Python

PS C:\Users\biotronics\Desktop\Light control by motion> & C:/Users/biotronics/AppData/Local/Programs/Python/Python311/python.exe "c:/Users/biotronics/Desktop/Light control by motion/motionDevice.py"
Publishing: {"device_id": "Device_001", "device_type": "motion_sensor", "motion_detected": false}
Message published with mid 1
No motion detected.
Publishing: {"device_id": "Device_001", "device_type": "motion_sensor", "motion_detected": false}
Message published with mid 2
No motion detected.
Publishing: {"device_id": "Device_001", "device_type": "motion_sensor", "motion_detected": true}
Message published with mid 3
Motion detected! Lights on
Message published with mid 4
Publishing: {"device_id": "Device_001", "device_type": "motion_sensor", "motion_detected": false}
Message published with mid 5
No motion detected.

Ln 16, Col 30 Spaces: 4 UTF-8 CR LF ⚡ Python 3.11.0 64-bit ⚡ Go Live ⚡ Prettier ⚡

```

Figure 18: Running lightControl.py script

```

lightControl.py - Light control by motion - Visual Studio Code [Administrator]
File Edit Selection View Go Run Terminal Help lightControl.py - Light control by motion - Visual Studio Code [Administrator]
EXPLORER ... passwd mosquito.config lightControl.py docker-compose.yml 2 readme.md ...
LIGHT CONTROL BY MOT...
> idea
> v
> .gitignore
> CACHEDIR.TAG
> README.md
> .vscode
> settings.json
> certs
> config
> mosquito.config
> passwd
> venv
> .dockerignore
> docker-compose... 2
> dockerfile
> lightControl.py
> motionDevice.py
> readme.md
> requirements.txt

4 #This is the second part of the script (lightControl.py), which runs on the light control system.
5
6
7
8 #Import libraries
9 import paho.mqtt.client as mqtt #for MQTT protocol
10 import json #to convert the python dictionary into a JSON string that can be written into a file
11
12 # Set up the initial state of the light
13 light_state = "off"
14
15 # Define the on_connect and on message functions for the MQTT client
16 def on_connect(client, userdata, flags, rc):
17     print("Connected to broker with result code "+str(rc))
18
19     # Subscribe to the "motion_sensor" and "light_control" topics
20     client.subscribe("motion_sensor")
21     client.subscribe("light_control")
22
23     def on_message(client, userdata, message):
24         # Extract the topic and payload from the received message
25         global light_state
26         topic = message.topic
27         if topic == "motion_sensor":
28             if message.payload.decode() == "1":
29                 light_state = "on"
30             else:
31                 light_state = "off"
32         elif topic == "light_control":
33             if message.payload.decode() == "1":
34                 client.publish("motion_sensor", "1")
35             else:
36                 client.publish("motion_sensor", "0")
37
38     client.on_message = on_message
39
40     client.loop_forever()
41
42     # Publish messages
43     client.publish("light_control", "1")
44
45     # Disconnect from the broker
46     client.disconnect()

lightControl.py ...
4 #This is the second part of the script (lightControl.py), which runs on the light control system.
5
6
7
8 #Import libraries
9 import paho.mqtt.client as mqtt #for MQTT protocol
10 import time #to simulate IoT delays
11 import random #to create random id
12 import json #to convert the python dictionary into a JSON string that can be written into a file
13
14 # Set up the device's ID and type
15 device_id = "Device_001"
16 device_type = "light_control"
17
18 # Define the on_connect and on_publish functions for the MQTT client
19 def on_connect(client, userdata, flags, rc): rc (return code) is u...
20 | print("connected to broker with result code "+str(rc))
21
22 def on_publish(client, userdata, mid): #mid value is an integer tha...
23 | print("Message published with mid "+str(mid))

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL SQL CONSOLE AZURE
powershell
cmd
Python

PS C:\Users\biotronics\Desktop\Light control by motion> & C:/Users/biotronics/AppData/Local/Programs/Python/Python311/python.exe "c:/Users/biotronics/Desktop/Light control by motion/lightControl.py"
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\biotronics\Desktop\Light control by motion> Connected to broker with result code 0

Ln 20, Col 38 Spaces: 4 UTF-8 CR LF ⚡ Python 3.11.0 64-bit ⚡ Go Live ⚡ Prettier ⚡

```