# SIT314 – Week 11 Evidence Report

Student: Thien Phu Pham | Student ID: s223244873

Unit: SIT314 / SIT729 – Internet of Things (IoT)

Week: 11 |

## 1. Week 11 Class Discussion Evidence

Topic: Security, Interoperability and Legal Issues in Scalable IoT

During the discussion, we examined how IoT ecosystems demand secure, interoperable, and compliant frameworks to operate effectively at scale.

Key Learning Points:  
- Security Layers: IoT threats occur at device (tampering), network (replay, spoofing), and cloud (application-level ransomware) layers.  
- Interoperability: Achieved through open standards such as MQTT and LoRaWAN to prevent vendor lock-in and simplify integration.  
- Legal Challenges: Focus on data ownership, privacy, and global compliance (e.g., GDPR, HIPAA).

Reflection: This discussion made me aware that IoT scalability depends equally on security assurance and regulatory compliance. Future IoT projects must embed encryption, user consent, and secure API design to align with legal expectations.

## 2. Week 11 Group Activity Evidence

Case Study: Retina Scanning Banking System for ATM Withdrawals

Our group evaluated the risks and interoperability challenges of introducing retina-based biometric identification in Australian banks.

Security Issues:

- Biometric template compromise: Irreversible once leaked; needs HSM storage and cancelable templates.  
- Spoofing and liveness detection: Mitigates fake retina images via pupil response checks.  
- Data transmission: TLS 1.3 and mTLS prevent MITM attacks.  
- Coercion protection: Use duress PIN signals and multi-factor authentication.

Interoperability Issues:

- Template standards: Adopt ISO/IEC 19794-6 for cross-vendor compatibility.  
- Federation between banks: Requires PKI trust and auditing mechanisms.  
- Legacy ATMs: Retrofit modules to enable retina capture hardware.

Legal and Rights Issues:

- Privacy Act 1988: Biometric data = sensitive information; must obtain express consent and minimise use.  
- APRA CPS 234: Mandates cyber resilience and third-party risk management.  
- Disability Discrimination Act 1992: Alternative methods must exist for users with eye conditions.

Reflection: This task illustrated how technical design, compliance, and accessibility converge in real-world IoT deployment.

## 3. Week 11 Technical Task Evidence

## Overview

This section documents the security improvements applied to a Node.js (Express) web server for IoT use. The evidence follows the exact order from the task instructions: (1) Baseline HTTP, (2) Baseline headers before Helmet, (3) HTTPS with self‑signed certificate, and (4) Headers after enabling Helmet. It then provides a brief step‑by‑step summary and a reflection on what was learned.

## Evidence (ordered by the instructions)

### Screenshot 1 — Baseline HTTP server running (server.js)

What it shows: The plain Express server is reachable at http://localhost:3050/ and returns “hello world”. This establishes the unsecured baseline (no TLS). Why it matters: Confirms the app works before adding security layers.

A screenshot of a phone

AI-generated content may be incorrect.

### Screenshot 2 — Baseline headers BEFORE Helmet (Postman)

What it shows:

A GET to http://localhost:3050/ returns minimal headers (e.g., X‑Powered‑By: Express) and lacks common security headers such as X‑Content‑Type‑Options, X‑Frame‑Options, Referrer‑Policy, and a strict Content‑Security‑Policy (CSP). Why it matters: Captures the “before” state to compare the effect of Helmet later. Note: The 404 status in the screenshot is acceptable for demonstrating headers—the header set is what we compare.

A screenshot of a computer

AI-generated content may be incorrect.

### Screenshot 3 — HTTPS with self‑signed certificate (serversecure.js)

What it shows:

Visiting https://localhost:3060/ prompts a browser warning (“connection is not secure”), which is expected with a self‑signed certificate in development. Why it matters: Proves the app now serves over HTTPS; production should use a CA‑signed certificate so browsers trust it automatically.

A screenshot of a computer

AI-generated content may be incorrect.

### Screenshot 4 — Headers AFTER enabling Helmet (Postman)

What it shows:

A GET to http://localhost:3050/ now includes multiple defensive security headers, e.g., Content‑Security‑Policy (locks down resources), X‑Frame‑Options: SAMEORIGIN (mitigates clickjacking), X‑Content‑Type‑Options: nosniff (prevents MIME sniffing), Referrer‑Policy (reduces data leakage), X‑DNS‑Prefetch‑Control, X‑Permitted‑Cross‑Domain‑Policies, and Strict‑Transport‑Security (effective over HTTPS). Why it matters: Demonstrates concrete hardening from Helmet compared to the baseline.

A screenshot of a computer

AI-generated content may be incorrect.

## Step‑by‑Step (Brief)

• Baseline HTTP: Created server.js (Express), started on port 3050, verified “hello world” in the browser (Screenshot 1).

• Headers (before): Queried http://localhost:3050/ in Postman to capture baseline headers without Helmet (Screenshot 2).

• Enable HTTPS: Generated self‑signed certs with OpenSSL, created serversecure.js, served on port 3060; acknowledged the self‑signed browser warning (Screenshot 3).

• Helmet hardening: Installed and conScreenshotd Helmet; re‑queried the server and verified additional security headers now appear (Screenshot 4).

## Reflection

Defense‑in‑depth: Moving from plain HTTP to HTTPS protects confidentiality and integrity of traffic, while Helmet adds layered mitigations against common web risks (XSS, clickjacking, MIME sniffing).

Dev vs. Prod: Self‑signed certificates are suitable for local testing but cause warnings; production requires a CA‑signed certificate and enables HSTS safely over HTTPS.

Evidence‑driven verification: Comparing headers before vs. after made security improvements tangible, independent of response status codes or body content.

Gotchas resolved: Quoted Windows paths for folders with spaces, added OpenSSL to PATH, and ensured Helmet version compatibility (pin helmet@5 for per‑middleware API).

Technical Reflection: This exercise demonstrated the importance of defence-in-depth for IoT applications.

## 4. How I Will Apply This Learning

- Implement multi-layer IoT security (HTTPS + Helmet + encryption).  
- Enforce privacy-by-design with data minimisation and consent.  
- Adopt interoperability standards (ISO/IEC and MQTT).  
- Follow legal frameworks (Privacy Act, GDPR).

## 5. Evidence Summary Table

|  |  |  |
| --- | --- | --- |
| Evidence Type | Description | File Reference |
| Class Discussion | Summary of IoT security & compliance principles | Week11\_Class\_Discussion.docx |
| Group Activity | Retina scanning banking risk analysis | week\_11\_group\_activity\_form.docx |
| Technical Task | HTTPS & Helmet implementation with comparison | Week11\_Technical\_Task\_Report\_with\_Evidence.docx |

## Conclusion

This improved report integrates annotated visual evidence with textual analysis, fulfilling the Week 11 requirements for class, group, and technical activities. It demonstrates conceptual understanding and practical implementation of security, interoperability, and legal principles in IoT applications.