ב"ה

עבודת בית 1 בקורס "אבטחת מוצרים ושירותים דיגיטאליים"

אוניברסיטת תל-אביב , תשפ"ו

הנחיות כלליות להגשה:

- יש להגיש את העבודה בזוגות
- קבוצה שבוחרת להגיש בשלשה עבודה סעיף הבונוס הופך לחובה והיא חייבת לתקן לפחות חמשה vulnerabilities ולהדגים לפחות 4 התקפות (הסעיף הזה הופך עבודה לשליש מהציון)
- תאריך ההגשה של העבודה הוא ה 25 לדצמבר 2025
- יש לציין את מספר השלב, ומס' ת.ז. של המגישים בשם הקבצים (ובכותרת של המסמך) לפי הפורמט הבא:
- <ת"ז מגיש 2>_<ת"ז מגיש 1>_<1 >
- לדוגמה, הגשת שלב 1 ע"י הסטודנטים בעלי ת"ז 123456789 ו-987654321 תיעשה בקובץ ששמו: 1_123456789_987654321

1. Background

You are a software developer tasked with creating a **secure authentication and session management library** that can be integrated into **web and mobile server-side applications**.

To simulate a real-world development workflow, you are allowed (and expected) to use a **Generative AI assistant** (e.g. ChatGPT, Claude, Gemini, Copilot, etc.) to generate the initial implementation of this library.

However, because security-sensitive code created by Gen-AI must never be trusted blindly, your task is to **critically analyze and audit** the generated implementation and identify:

- What was implemented correctly
- What was implemented incorrectly

- What is missing
- What is vulnerable and why

This assignment develops your skills in:

- Secure authentication design
- Session management best practices
- AI-assisted development
- Security code review
- Threat modeling

2. Your Assigned Context (Unique per student)

You MUST include **both** of the following in your prompt and report:

- **Your student ID: ____________________**
- **Your assigned application context: ____________________**

Examples of contexts (you will be assigned one):

- Online banking system
- Medical patient portal
- University LMS
- E-commerce platform
- IoT smart-home control system
- Military communication platform
- Voting system
- Cryptocurrency wallet
- Ride-sharing app
- Dating platform

Your context must influence your requirements and threat model.

3. Part 1 – Prompt Engineering (25%)

Write a **complete and precise prompt** to give to a Gen-AI tool, instructing it to generate a secure authentication and session management library with the following properties:

3.1 Your prompt MUST include

- Your **student ID**
- Your **application context**
- The **programming language** to be used
- Requirement that the library must support **web and mobile server side applications**
- Request for **clean, modular, well-documented code**
- Request for **example usage code**
- Request for **design explanation / documentation**

3.2 Mandatory security requirements

Your prompt must require **all** of the following mechanisms:

Authentication requirements

- Secure password hashing (using: bcrypt, scrypt, Argon2, etc.)
- Password complexity policy enforcement
- Multi-Factor Authentication (TOTP or WebAuthn)
- Brute-force login protection (rate-limiting or temporary lockout)
- Email or message-based account verification flow
- Secure password reset mechanism
- Protection against credential-stuffing

Session management requirements

- Cryptographically secure random session ID generation
- Session expiration (absolute timeout)
- Idle session timeout
- Session renewal/rotation after login
- Secure logout and full session invalidation

- Automatic invalidation on password change
- Token-based authentication support (e.g. JWT)

Cookie / token security requirements

- `HttpOnly`, `Secure`, `SameSite` options
- No localStorage for tokens
- Short-lived access tokens
- Secure refresh token handling

3.3 Strict forbidden elements (must be included in prompt)

Your prompt **must forbid**:

- Hard-coded secrets
- Plaintext password storage
- MD5 / SHA-1
- Predictable/random-weak session tokens
- Local Storage for auth tokens
- Infinite-lifetime tokens
- Static encryption keys

Note: Gen-AI will often violate one or more of these – you will analyze that in Part 2.

3.4 What to submit for Part 1

1. The **exact prompt** you wrote
2. The **AI's full response (code and explanation)**

These will be graded on:

- Completeness
- Security awareness
- Precision and clarity
- Coverage of requirements

- Alignment with your application context

4. Part 2 – Security Analysis of Generated Code (75%)

You must analyze the **actual code produced by the AI**.

Your analysis must follow the **exact structure below** for each requirement.

Required Analysis Format (Mandatory Template)

For each of the following 15 items, your report must include:

- **Requirement -** explain its importance
- **Location in code** (file + approx. line numbers)
- **The code snippet**
- **Your security analysis**
- **Classification (one only):**
  - ✅ Secure
  - ⚠ Partially Secure
  - ❌ Insecure
  - ❌ Missing
- If vulnerable, describe:
  - **Attack scenario**
  - **Possible impact**
  - **How to fix it**

Requirements to analyze

You must analyze **all** of the following in the generated code:

1. Password hashing algorithm & configuration
2. Password policy enforcement (length/complexity)

3. Brute-force / rate limiting protection
4. Multi-factor authentication implementation
5. Password reset token generation
6. Password reset validation & expiration
7. Session/token generation method
8. Token entropy & randomness quality
9. Token expiration mechanism
10. Session invalidation on logout
11. Session invalidation on password change
12. Cookie / token storage configuration
13. Protection against session fixation
14. Privilege separation / role checking
15. Cryptographic key management

Failure to analyze **any one** of these results in automatic point loss.

---

5. Extension Task (Bonus – up to +10%)

This part is **optional** and intended for advanced students.

Choose ONE of the following:

Option A — Secure Refactoring (recommended)

Fix **at least two (2) critical vulnerabilities** you found in the generated code and:

- Provide corrected code
- Provide a before/after comparison
- Explain why the new implementation is secure

Option B — Attack Demonstration

Select **one vulnerable mechanism** and:

- Demonstrate (theoretically or practically) how it could be exploited
- Provide a step-by-step attack explanation

6. Submission format

You must submit a **single PDF** containing:

1. Cover page (Name + ID + Context + AI tool used)
2. Your Prompt
3. AI Generated Code (as a separate file)
4. Structured Security Analysis
5. (Optional) Extension Task

7. Grading Breakdown

Section	**Weight**
Prompt quality	**25%**
Code security analysis	**75%**
Extension task (if completed)	**+10% bonus**

8. Evaluation Focus

You will be graded on:

- Security understanding
- Depth of analysis
- Correct vulnerability identification
- Technical accuracy

- Quality of reasoning

Not on:

- How “nice” the AI code is
- Whether you used the “best” AI tool