## PASTA worksheet

| **Stages** | **Sneaker company** |
| --- | --- |
| **I. Define business and security objectives** | Make **2-3 notes** of specific business requirements that will be analyzed.   * The app will process financial transactions for buying and selling sneakers. * It will handle significant back-end processing for order management and user accounts. * Compliance with industry regulations like PCI DSS for payment data security is required. |
| **II. Define the technical scope** | * API for app backend communication. * PKI for secure authentication and encryption. * AES for data encryption. * SHA-256 for hashing sensitive data. * SQL for database management.   These technologies are prioritized to ensure secure data transmission, storage, and payment handling. API enables modular communication, PKI facilitates trust and verification, AES secures stored data, SHA-256 protects integrity of sensitive info, and SQL supports structured data management. |
| **III. Decompose application** | [Sample data flow diagram](https://docs.google.com/presentation/d/1ol7y79popTFfNHM-90ES-H-i1Lpd0YNvPShxBlXozjg/template/preview?resourcekey=0-DZAkf7Vzh2PXsP-j3oXV-g)Data flow from user mobile app → API → backend server → SQL database.   * User authentication via PKI certificates. * Payment processing data encrypted with AES. * Stored user credentials and transaction records |
| **IV. Threat analysis** | * Internal threats: Insider attacks or employee mishandling of sensitive data. * External threats: Hackers attempting SQL injection, man-in-the-middle attacks on API, or data theft. |
| **V. Vulnerability analysis** | * Potential codebase issues such as insecure coding practices leading to injection attacks. * Database weaknesses like unpatched vulnerabilities or poor access controls. * Network flaws such as unencrypted communication or weak TLS protocols. |
| **VI. Attack modeling** | [Sample attack tree diagram](https://docs.google.com/presentation/d/1FmWLyHgmq9XQoVuMxOym2PHO8IuedCkan4moYnI-EJ0/template/preview?usp=sharing&resourcekey=0-zYPY7AhPJdcClXamlAfOag)   * Breaking API authentication. * Exploiting SQL injection to access backend data. * Intercepting unencrypted network traffic. * Phishing users to steal credentials. |
| **VII. Risk analysis and impact** | Security controls to reduce risk:   * Implement multi-factor authentication for access. * Use parameterized queries to prevent SQL injection. * Encrypt all data in transit and at rest using TLS and AES. * Regular security audits and patch management to address vulnerabilities. |