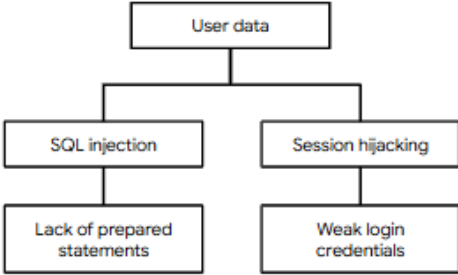


# PASTA

Stages	Online Sneaker Company
1. Define business and security objectives.	<p>Business requirements to be analyzed:</p> <ul style="list-style-type: none"><li>• <i>Users can create member profiles internally or by connecting external accounts.</i></li><li>• <i>The application must process financial transactions.</i></li><li>• <i>The application must be PCI-DSS compliant.</i></li></ul>
2. Define the technical scope	<p>List of technologies used by application:</p> <ul style="list-style-type: none"><li>• <i>Application programming interface (API)</i></li><li>• <i>Public key infrastructure (PKI)</i></li><li>• <i>SHA-256</i></li><li>• <i>SQL</i></li></ul> <p><i>APIs facilitate data sharing between customers, partners and employees, so they should be prioritized. They handle a large amount of sensitive data while connecting multiple users and systems. However, details such as which APIs are used should be considered before prioritizing one technology over another. Therefore, they may be more prone to security vulnerabilities because there is a larger attack surface.</i></p>
3. Break down the application	<p><i>Sample data flow diagram</i></p> <p><b>Data flow diagram</b></p> <p><b>Note:</b> This data flow diagram represents a single process. Data flow diagrams for an application like this are normally much more complex.</p> <pre>graph LR; User[User] -- "Searching for sneakers for sale." --&gt; Process((Product search process)); Process -- "Listings of current inventory." --&gt; Database[Database];</pre>
4. Threat analysis	<ul style="list-style-type: none"><li>• <i>Injection</i></li><li>• <i>Session hijacking</i></li></ul>
5. Vulnerability analysis	<ul style="list-style-type: none"><li>• <i>Lack of prepared statements.</i></li><li>• <i>Internal API token</i></li></ul>

6. Attack modeling	<p><i>Sample Attack Tree Diagram</i></p> <p style="text-align: center;"><b>Sample attack tree</b></p> <p><b>Note:</b> Applications like this normally have large, complex attack trees with many branches.</p>  <pre> graph TD     A[User data] --&gt; B[SQL injection]     A --&gt; C[Session hijacking]     B --&gt; D[Lack of prepared statements]     C --&gt; E[Weak login credentials] </pre>
7. Risk and impact analysis.	<p>4 security checks that can reduce the risk are:  <i>SHA-256, incident response procedures, password policy, principle of least privilege</i></p>

### Stage 1: Define business and security objectives

**Summary:** These objectives are defined at the beginning by asking general questions about the purpose of the application. For example, how does the app make the business money? Understanding the answer to these questions helps guide the detailed work that follows.

**Recommendations:** A shopping app like this will need to process payments. Based on this description, we know that certain technologies are needed to keep information private and secure, and that everything will have to comply with PCI-DSS standards.

### Stage 2: Define the technical scope

**Summary:** The goal here is to understand the attack surface by identifying the technologies the application uses and understanding their dependencies.

**Recommendations:** APIs facilitate the exchange of data between customers, partners, and employees, so they should be prioritized. Additionally, they handle a lot of sensitive data while connecting multiple users and systems to each other. However, details such as which APIs are being used should be considered before prioritizing one technology over another. Thus, they may be more prone to security vulnerabilities because there is a larger attack surface.

### Stage 3: Decompose the application

**Summary:** The goal is to review how the application works and how Security Controls are currently implemented.

**Recommendations:** The data flow diagram shows how a typical search request passes through multiple layers. Something that could be checked is that the MySQL database is using prepared statements when entering queries.

### Stage 4: Threat Analysis

**Summary:** The main objective of the fourth stage is to consider the types of threats that could affect your application. Another thing to keep in mind is the types of data the application processes.

**Recommendations:** Injection attacks are common in SQL databases. Session hijacking is

possible because the application communicates cookies between multiple layers. It is important to consider the technological attack surface and any relevant threats to your product to effectively implement your Information Security responsibilities.

#### **Stage 5: Vulnerability Analysis**

**Summary:** It consists of associating resource vulnerabilities with potential threats. The goal is to identify what is wrong with the application design or its code base based on your Security tests.

**Recommendations:** The lack of prepared statements can make our SQL database vulnerable to injection attacks. And session hijacking is possible if cookies are mishandled between input and output sources.

#### **Stage 6: Attack Model**

**Summary:** At this stage, the objective is to link the threats and vulnerabilities identified in the previous steps using the structure of an attack tree, which makes it possible to demonstrate that the potential threats that have been identified are really viable. Resources such as MITER ATT&CK and the CVE® List are useful references for this.

**Recommendations:** This example shows how user data is vulnerable to the attacks identified above. Like the sample data flow diagram, a real attack tree for a mobile app would be much more complex than this.

#### **Stage 7: Risk and impact analysis**

**Summary:** The goal of the final stage of PASTA is to identify ways to mitigate the risks that were identified from stages 4 to 6 and plan for remaining risks that cannot be remedied.

**Recommendations:** SHA-256, Incident Response Procedures, password policy and the principle of least privilege are some examples of technical, operational, and management controls that can be implemented before release to reduce risk.