PASTA worksheet

Stages	Sneaker company
I. Define business and security objectives	- **User-friendly experience**: The app should allow users to easily sign-up, log in, and manage their accounts, ensuring smooth navigation and interactions. - **Data privacy and security**: Ensuring users' data privacy is a major priority, and the app must handle sensitive information responsibly to build trust with customers. - **Efficient sales and payment process**: The app should facilitate quick and clear sales transactions, offering multiple payment options while ensuring proper payment handling to avoid legal issues.
II. Define the technical scope	I would prioritize evaluating the **public key infrastructure (PKI)** first, as it handles the encryption of sensitive data and key exchanges. Any weakness in the PKI implementation could lead to compromised user data, such as payment information. Ensuring that encryption methods (AES and RSA) are properly implemented is crucial to prevent potential data breaches.
III. Decompose application	Sample data flow diagram
IV. Threat analysis	- **API Attacks**: Threat actors could exploit vulnerabilities in the API, such as improper authentication or unvalidated input, to gain unauthorized access to user data or perform malicious actions like injecting harmful code. - **SQL Injection Attacks**: If the SQL queries are not properly parameterized, attackers could manipulate the search function or other database interactions to execute malicious SQL commands, compromising the database and stealing sensitive user information.

V. Vulnerability analysis	- **Insecure API Endpoints**: If the API endpoints are not secured with proper authentication mechanisms or fail to enforce encryption (e.g., using HTTP instead of HTTPS), attackers could intercept and manipulate data in transit, leading to unauthorized access or data breaches. This is a common vulnerability highlighted by OWASP's API Security Top 10. - **Lack of Input Validation (SQL Injection)**: If user inputs, such as search queries or sneaker listing details, are not properly sanitized and validated, they could be exploited via SQL injection attacks, allowing attackers to modify queries, access sensitive data, or compromise the entire database.
VI. Attack modeling	Sample attack tree diagram
VII. Risk analysis and impact	- **Implement Strong API Authentication and Encryption**: Use secure authentication methods (e.g., OAuth 2.0) and enforce encryption (HTTPS with TLS) for all API interactions to prevent unauthorized access and data interception. - **Input Validation and Sanitization**: Ensure that all user inputs (e.g., search queries, form submissions) are properly sanitized and validated to prevent SQL injection and other forms of input-based attacks. - **Use Multi-Factor Authentication (MFA)**: Adding MFA for user accounts increases security by requiring multiple forms of verification (e.g., password and mobile verification) to reduce the risk of account takeovers. - **Regular Vulnerability Scanning and Patch Management**: Regularly scan the application and infrastructure for vulnerabilities (using tools like OWASP ZAP or Burp Suite), and promptly apply security patches to fix known vulnerabilities in the software and third-party components.