PASTA worksheet

So here's the deal. Selling sneakers online sounds simple until you realize every hacker on the planet would love to get their hands on customer credit cards, order data, and basically any excuse to break into your backend. That's where the PASTA model kicks in.

Stages	Sneaker company
I. Define business and security objectives	*The app isn't just listing sneakers. It's processing real financial transactions, storing user profiles (sometimes linked to external accounts), and needs to be squeaky clean on PCI-DSS compliance.*
II. Define the technical scope	List of technologies used by the application: ● *API* ● *PKI* ● *AES* ● *SHA-256* ● *SQL* The app is built with API's, PKI, AES encryption, SHA-256 hashing, and SQL running the show. Out of all these, the API get the spotlight because it's the VIP entrance for data coming and going. If the API gets sloppy, that's like leaving the club doors wide open for freeloaders. Its job is to connect users, partners, and employees, which means its attack surface is larger than it should be.
III. Decompose application	Think of this as a blueprint for how people shop for sneakers in the app. The data flow diagram shows how a user searches, retrieves inventory listings, and interacts with backend systems. It's not as glamorous as it sounds, but if the data pipeline gets messed with, the entire app is toast.
IV. Threat analysis	Two big threats come to mind: injection attacks and session hijacking. Injection lets attackers slip in malicious commands like they own the place, while session hijacking is basically someone sliding into your account mid-session to wreak havoc.
V. Vulnerability analysis	*The weak links? Lack of prepared SQL statements and broken API token. It's like leaving a neon sign that says "Hack me" on the database. Both are low effort exploits with high payouts for attackers.*

VI. Attack modeling	The attack tree is where the horror movie plays out. SQL injection, weak logins, and stolen sessions all branch out from careless coding and poor security hygiene.
VII. Risk analysis and impact	We keep the chaos under control with a few straightforward moves: use SHA-256 for hashing, enforce a no-nonsense password policy, stick to least privilege access, and have an incident response plan ready for when things inevitably hit the fan.
