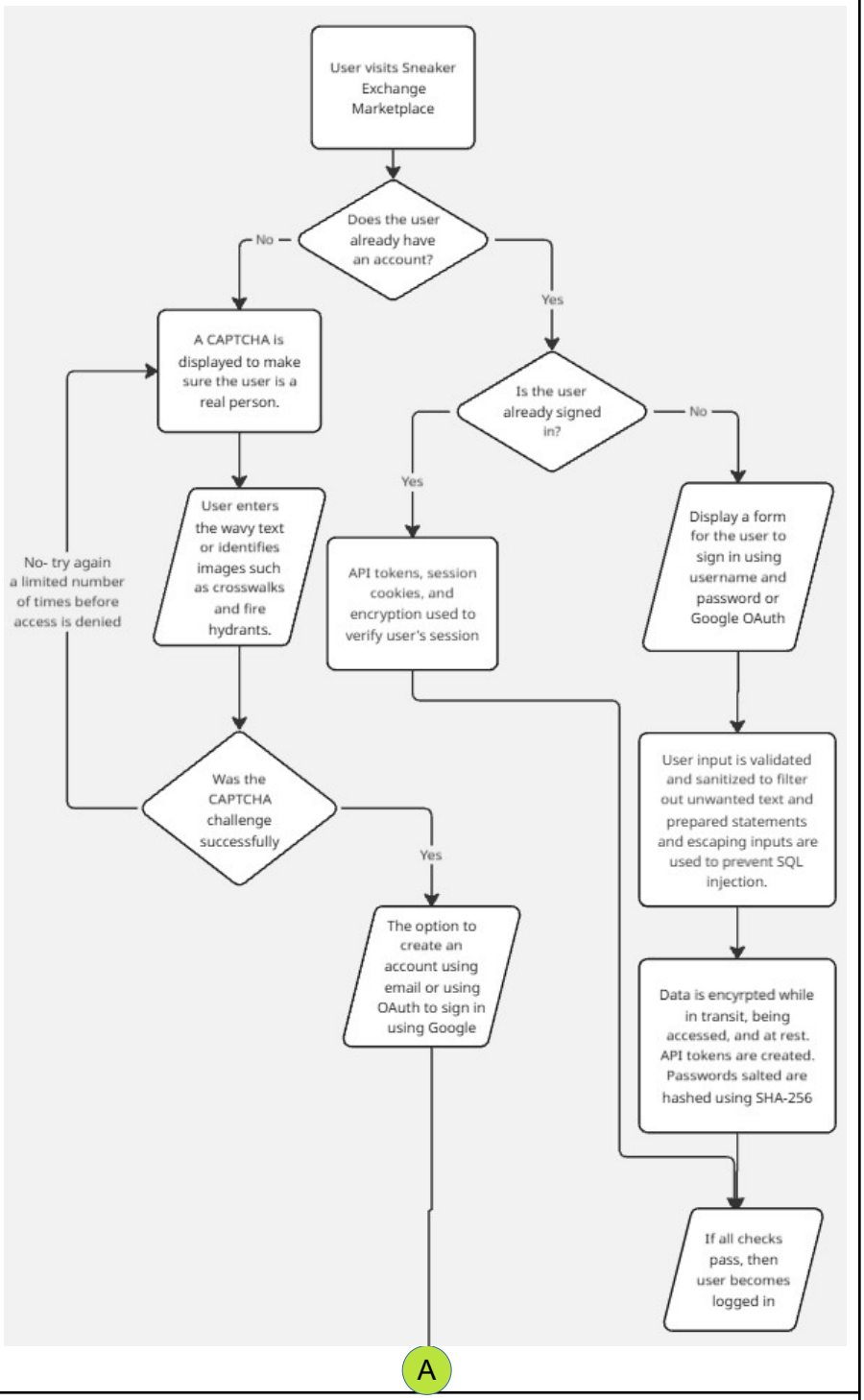
Jason Ash

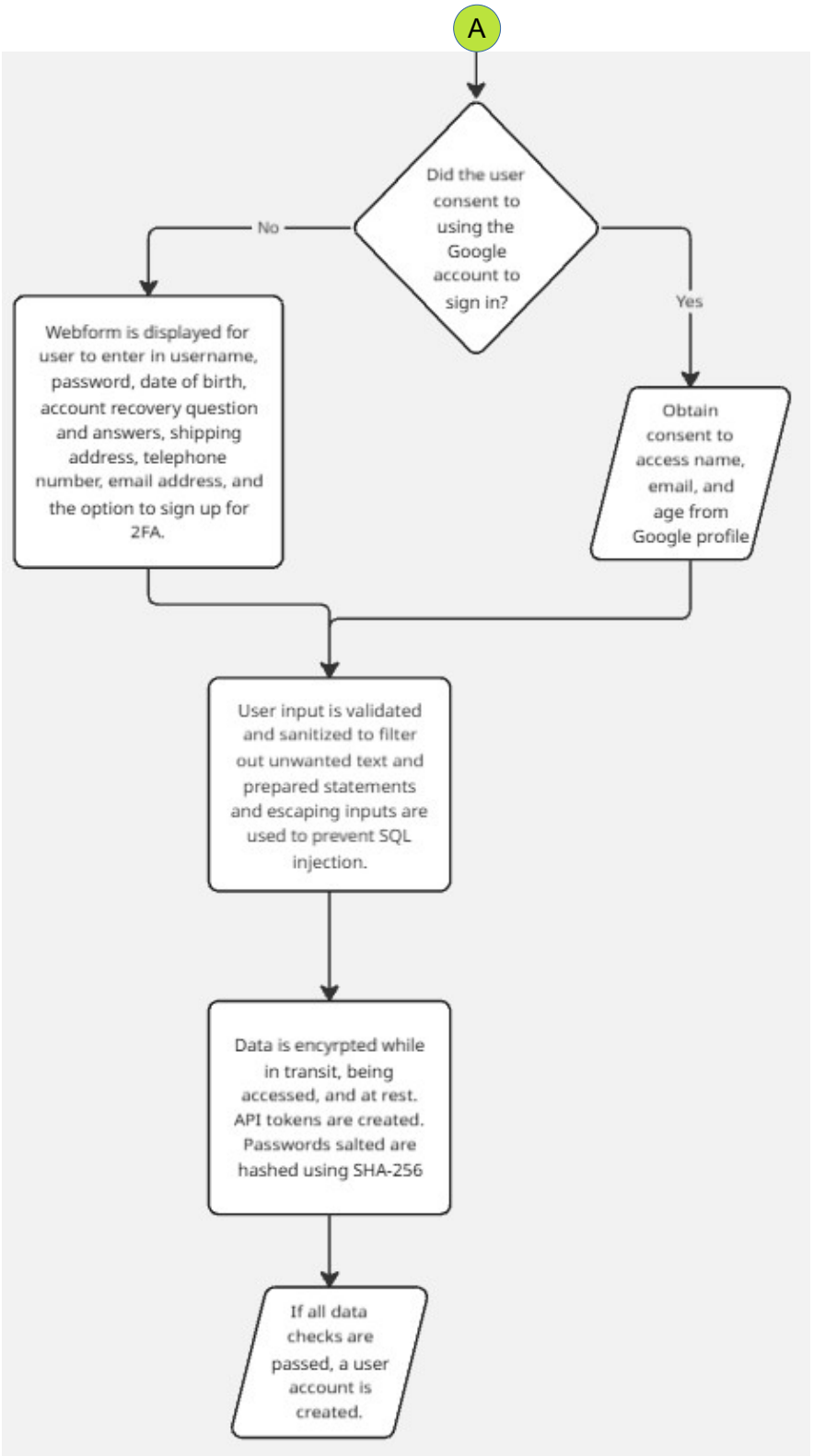
04 January, 2026

PASTA worksheet

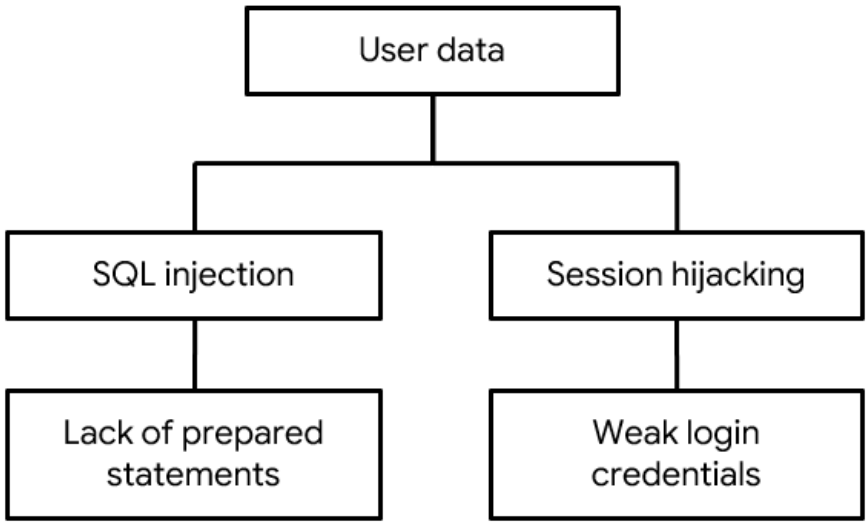
Stages	Sneaker company
I. Define business and security objectives	*Make **2-3 notes** of specific business requirements that will be analyzed.* ● *Will the app process transactions?* Yes, the app will process payments for buyers using credit cards, PayPal, Google Pay, Amazon Pay, and others. Therefore, keeping these payment methods secure, private, and still easy to use is a top priority. ● *Does it do a lot of back-end processing?* Yes, a lot of back-end processing will occur including querying the database of sneakers when a buyer performs a search. Also, sneakers for sale will be added to the database when a seller lists them. The back-end will also process logins; account creation, setup, and management; messages exchanged between buyers and sellers; storing feedback for buyers and sellers; and processing payments through the methods described above. All of this information needs to be kept private and secure. ● *Are there industry regulations that need to be considered?* PCI DSS is a major industry regulation to consider when processing payments. Also, if doing business in the EU, GDPR General Data Protection Regulations will have to be followed.
II. Define the technical scope	*List of technologies used by the application:* ○ API tokens use encrypted blocks of code that contains information about a user and site permissions. These are transmitted between a user's device and the server. OAuth could also be used to allow users to sign in with their Google or Facebook credentials instead of creating a separate login for the sneakers marketplace app. Also, making sure that secure API components are used an avoiding insecure ones, such as vulnerable versions of Log4j, will be essential in maintaining a secure application and Website. ○ PKI: Using asymmetric and symmetric encryption and digital certificates when appropriate is vital. AES will be used to encrypt sensitive information, such as credit card information. RSA encryption will be used to exchange keys between the app and a user's device. ○ AES: As mentioned above, AES will used for encrypting sensitive information, such as credit card information.

	○ SHA-256 hashing will be used together with salting to protect sensitive user data, such as passwords and credit card numbers. ○ SQL injection will be prevented by sanitizing and validating input and filtering out unwanted text received through web forms. Escaping inputs will also be used to prevent someone from inserting code that a program is not expecting. *Prepared statements that executes SQL statements before passing them to a database will be used.* *Write **2-3 sentences** (40-60 words) that describe why you choose to prioritize that technology over the others.* The reason that I chose these technologies was mainly due to my limited experience with cybersecurity. Therefore, I can only describe technologies that I am aware of. The ones that I would prioritize above both from a security standpoint and since I have more notes on the topic than the others are: Using secure APIs, PKI encryption, preventing SQL injection, and SHA-256 hashing and salting.
III. Decompose application	Note: Only some general steps in account creation are shown in the data flow diagram because data flow in a marketplace app could be quite extensive and is probably more detailed than this. I created this diagram from scratch on Miro.com.


User visits Sneaker Exchange Marketplace
Does the user already have an account?
No
Yes
A CAPTCHA is displayed to make sure the user is a real person.
Is the user already signed in?
No
Yes
User enters the wavy text or identifies images such as crosswalks and fire hydrants.
API tokens, session cookies, and encryption used to verify user's session
Display a form for the user to sign in using username and password or Google OAuth
No- try again a limited number of times before access is denied
Was the CAPTCHA challenge successfully
Yes
User input is validated and sanitized to filter out unwanted text and prepared statements and escaping inputs are used to prevent SQL injection.
The option to create an account using email or using OAuth to sign in using Google
Data is encyrpted while in transit, being accessed, and at rest. API tokens are created. Passwords salted are hashed using SHA-256
If all checks pass, then user becomes logged in
A


A
Did the user consent to using the Google account to sign in?
No
Yes
Webform is displayed for user to enter in username, password, date of birth, account recovery question and answers, shipping address, telephone number, email address, and the option to sign up for 2FA.
Obtain consent to access name, email, and age from Google profile
User input is validated and sanitized to filter out unwanted text and prepared statements and escaping inputs are used to prevent SQL injection.
Data is encyrpted while in transit, being accessed, and at rest. API tokens are created. Passwords salted are hashed using SHA-256
If all data checks are passed, a user account is created.

IV. Threat analysis	*List **2 types of threats** in the PASTA worksheet that are risks to the information being handled by the application.* ● *What are the internal threats?* ○ If the principle of least privilege, separation of duties, and MFA are not used, a phishing email could cause an employee to accidentally give their credentials to malicious actors. ○ If access to employees no longer with the company is not revoked immediately upon their departure, then a disgruntled employee might try to delete, alter, or steal data from the database. ● *What are the external threats?* ○ If input is not validated and sanitized and prepared statements are not used, this could leave the internal database vulnerable to SQL injection attacks. ○ If passwords are not hashed using a strong encryption algorithm, then login credentials could be threatened by hash collisions and credential stuffing, and if passwords are left unsalted, then rainbow tables could be used to hack into employee or customer accounts.
V. Vulnerability analysis	*List **2 vulnerabilities** in the PASTA worksheet that could be exploited.* ● *Could there be things wrong with the codebase?* ○ If insecure APIs or programming practices are used and the co debase does not undergo regular security audits, then hackers could use these vulnerabilities to steal sensitive information. ○ Hardcoding sensitive data such as API keys, passwords, and token must never occur. ○ The build environment must be kept secure and hardened to prevent the injection of malicious code and data theft. ● *Could there be weaknesses in the database?* ○ Hashing and salting could be used to prevent hackers from using rainbow tables to break into employee and customer accounts. ○ A vulnerability scanner could find misconfigurations and programming flaws in technologies used in the network including databases. ● *Could there be flaws in the network?* ○ Public key infrastructure and digital certificates could prevent keys that are lost or stolen from compromising the network and data stored therein. ○ MFA reduces the risk of an attacker gaining authentication by requiring something the employee has (such as a code from a DUO app) or a biometric characteristic, such as a finger print or face scan. ○ The principle of least privilege prevent an attacker from getting administrative access to a database if regular user credentials are compromised. ○ Server and firewall configuration must be regularly audited and patches and update applied immediately (unless doing so would

	be counterindicated) to prevent security flaws in the system including devices connected to the network from being exploited.
VI. Attack modeling	 (This attack tree is the sample provided with the assignment)
VII. Risk analysis and impact	*List **4 security controls** that you've learned about that can reduce risk.* ○ Using secure API components could reduce vulnerabilities, such as Log4j. ○ SHA-256 hashing and salting passwords prevents rainbow table attacks. ○ Sanitizing and validating input and using prepared statements protects against SQL injection. ○ Following the principle of least privilege and requiring MFA hardens against stolen credentials or brute-force attacks from being used by a malicious actor.