CSCI 360 T D5 SMH + A
Table of Contents

Abuse Cases **1**
Abuse Case 1 1
Abuse Case 2 1
Domain Model **1**
System Sequence Diagram **1**
Operations Contract **1**
Class Diagram **1**

Abuse Cases

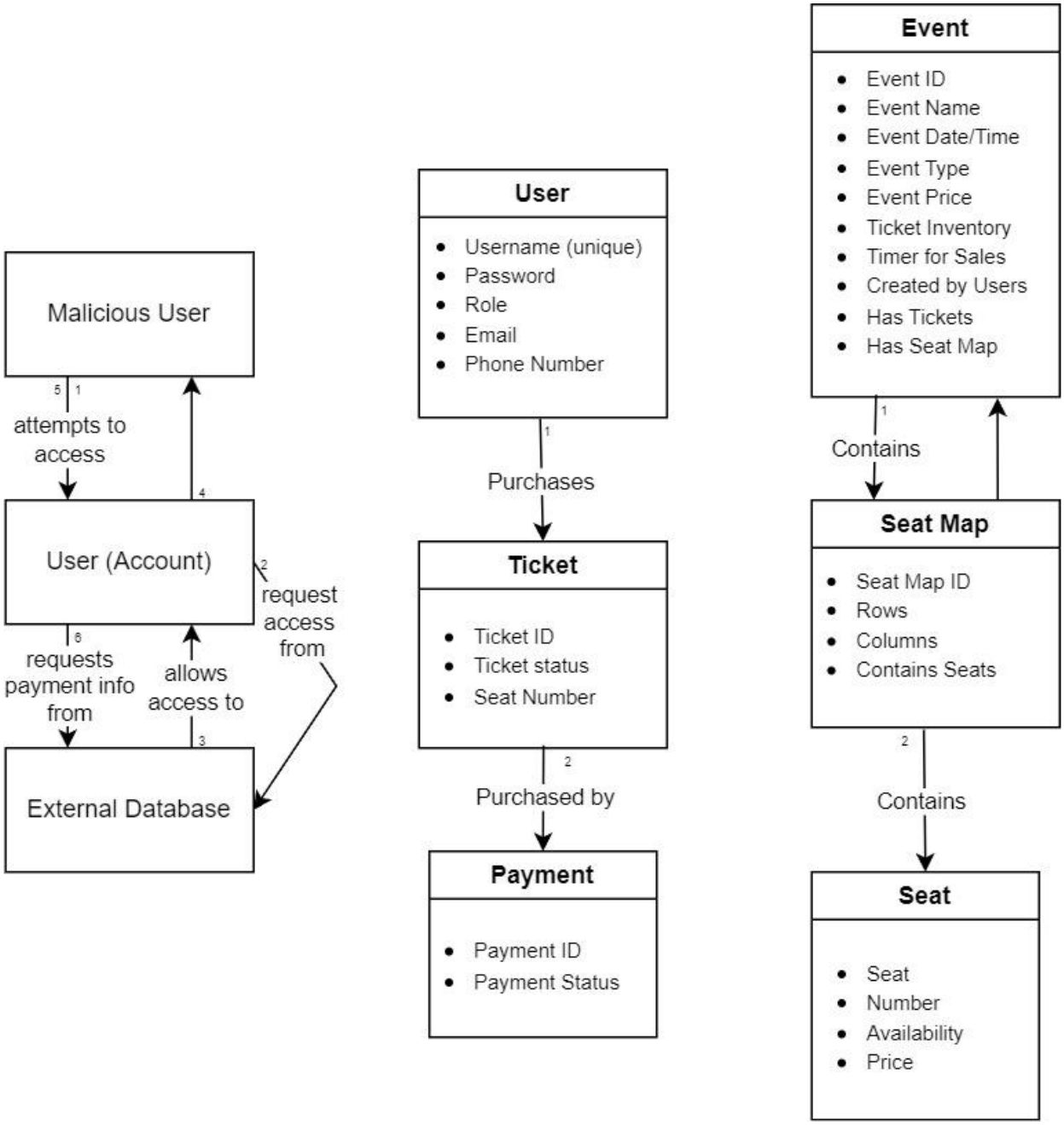
Abuse Case 1 - Stolen Credentials

A malicious user gains access to the credentials of another user's account and attempts to log into the system as if they were said user. This can be done in order to change or steal information of the account owner.

Abuse Case 2 - Pin Required to Reuse Card Stored in Account

If a malicious user gains access to another user's account and wishes to purchase tickets using the credit card already used on the account, the malicious user is able to do so.

Domain Model



Operations Contract

Abuse Case 1 - Stolen Credentials

Name: access data()
Use Case: Stolen Credentials
Precondition: Malicious user has logged in successfully.
Postcondition: System provides user data.

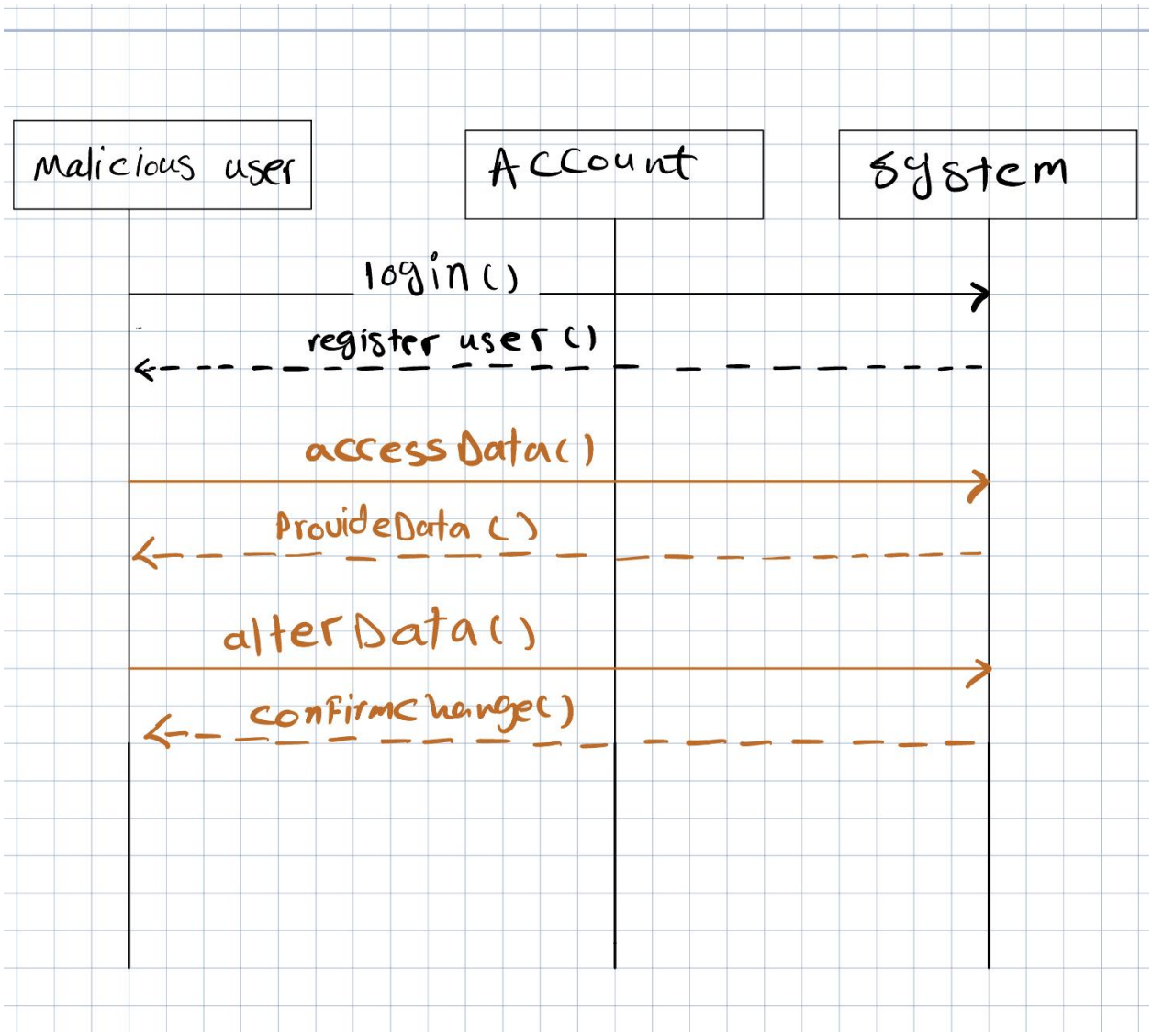
Name: alterData()
Use Case: Stolen Credentials
Precondition: Malicious user has accessed user data.
Postcondition: System changes data and confirms the alteration.

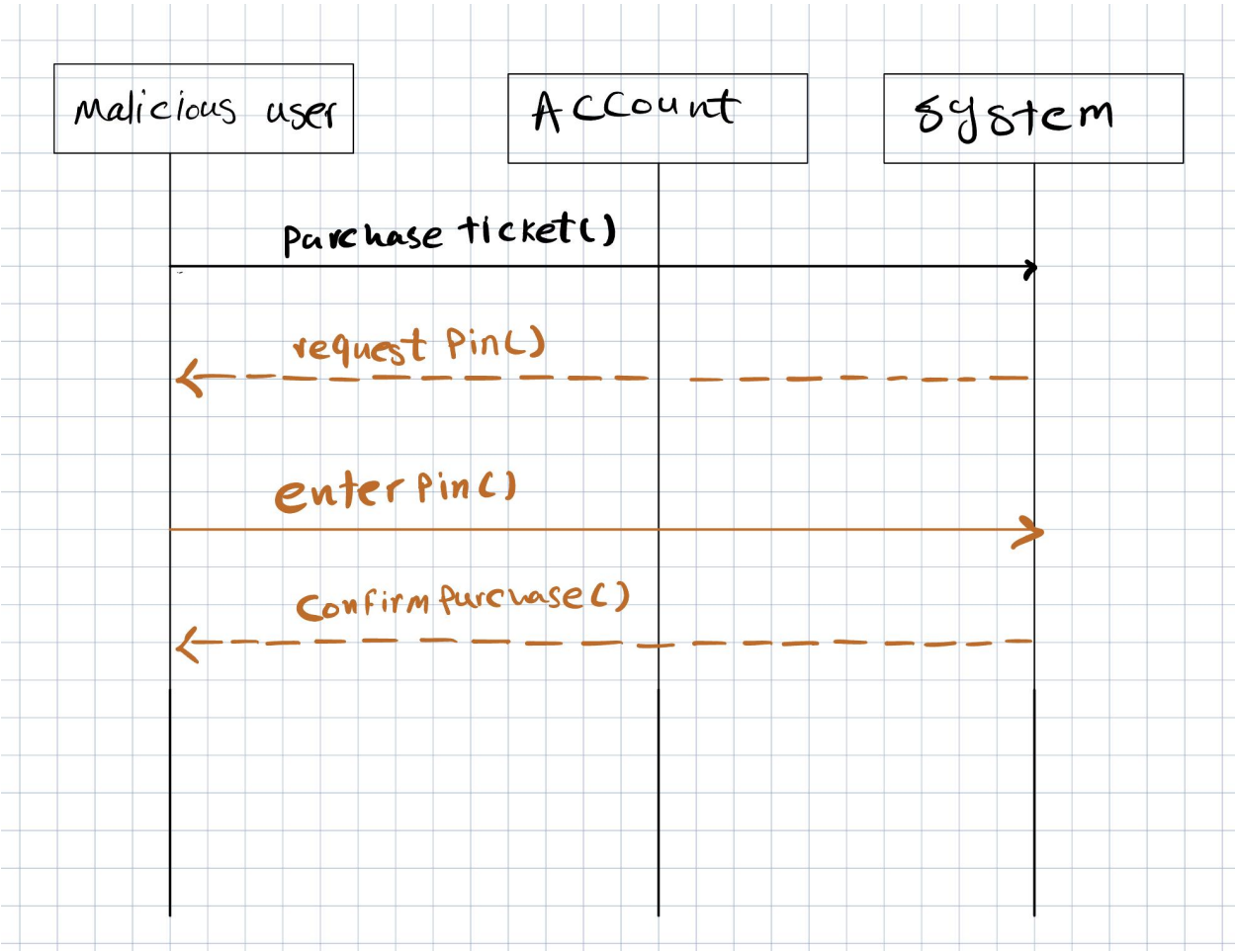
Abuse Case 2 - Pin Required to Reuse Card Stored in Account

Name: purchaseTicket()
Use Case: Pin Required to Reuse Card Stored in Account
Precondition: Malicious user has logged in successfully.
Postcondition: System prompts for the PIN of the stored card.
Operation Contract: enterPIN()

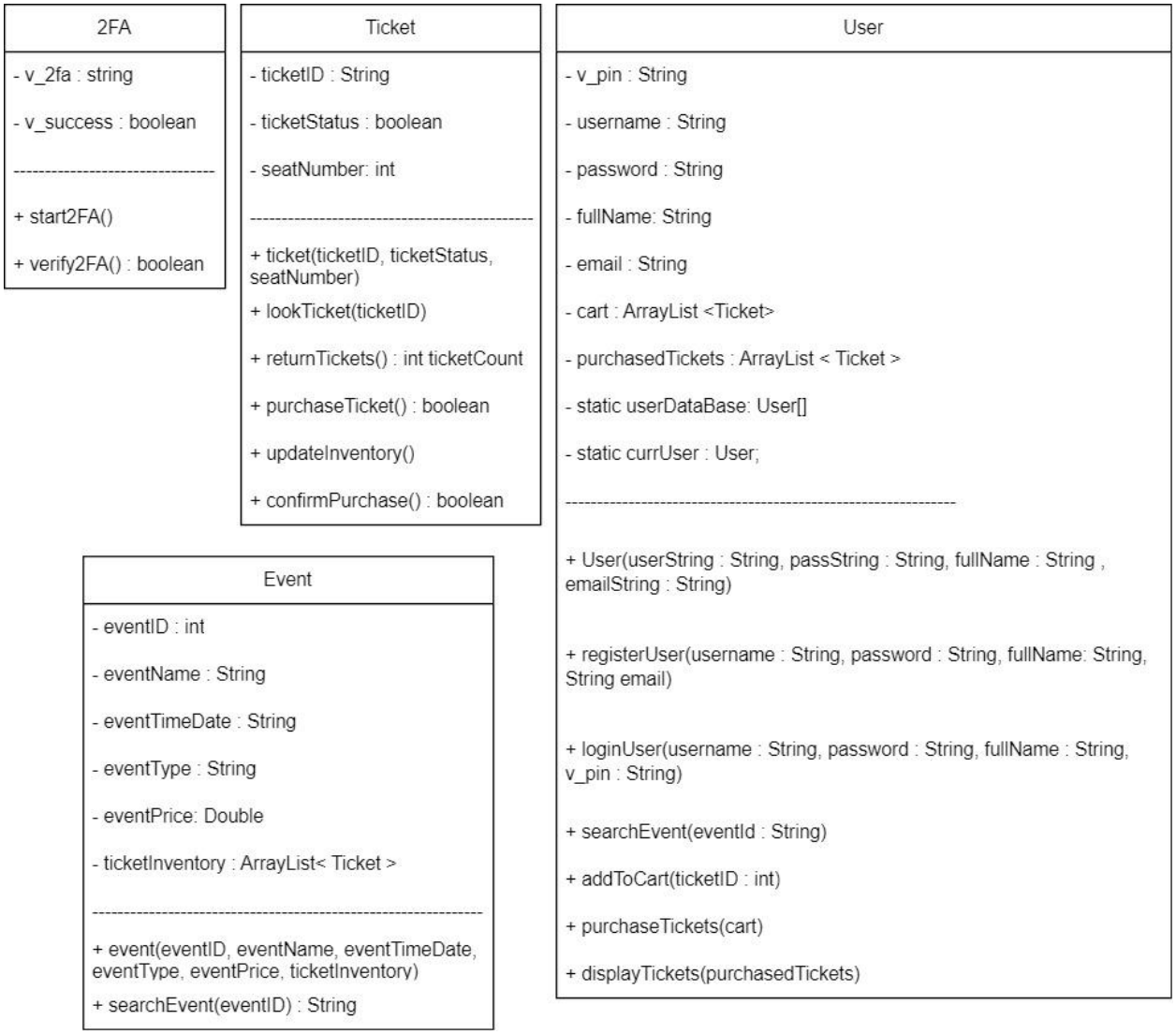
Name: enterPIN()
Use Case: Pin Required to Reuse Card Stored in Account
Precondition: System has prompted for PIN.
Postcondition: If PIN is correct, system confirms ticket purchase.

System Sequence Diagram




Malicious user
Account
System
Purchase ticket()
request Pin()
enter Pin()
Confirm purchase()

Class Diagram



TP D6

Designing Security In Project

Security is needed in any project development, especially in systems that handle sensitive information and transactions. In this paper, we will discuss how to design security measures to address two specific abuse cases: Stolen Credentials and Unauthorized Card Usage. These two scenarios pose significant threats to the integrity and confidentiality of a system, making them crucial areas to block against potential breaches.

The way that our team (SMH + A) is implementing security in our project is by our two abuse cases. Our two abuse cases are: Abuse Case 1 - Stolen Credentials, a malicious user gains access to the credentials of another user's account and attempts to log into the system as if they were said user. This can be done in order to change or steal information of the account owner. Abuse Case 2 - Pin Required to Reuse Card Stored in Account, if a malicious user gains access to another user's account and wishes to purchase tickets using the credit card already used on the account, the malicious user is able to do so.

The first abuse case, "Stolen Credentials," involves a malicious user gaining access to another user's account by acquiring their login credentials. To mitigate this risk, an authentication mechanism is essential. A multi-pronged approach is recommended. Strong Password Policies: encourage users to create strong and unique passwords by implementing policies that require a combination of upper and lower case letters, numbers, and special characters. Educating users on password best practices can also enhance security. Multi-Factor Authentication (MFA): Implementing MFA adds an extra layer of security by requiring users to provide one authentication factor. This could include something they know (password), something they have

(a mobile device), and something they are (face scan). MFA significantly reduces the risk of unauthorized access even if credentials are compromised. One class that we have implemented would be the accessData(). All systems usually have an account tab where you can find out their email address and other things in their daily lives like their address. Allowing the user to require some form of authentication to this tab would allow these malicious acts to not work.

The second abuse case, "Pin Required to Reuse Card Stored in Account," involves a malicious user gaining access to another user's account and attempting to make purchases using the stored credit card information. To mitigate this risk, we can implement the following measure: Authorization is a crucial aspect of security, ensuring that users can only access and perform actions they are explicitly authorized to do. PIN or Secure Code: require users to input a PIN or another secure code for making payments or altering payment methods. This additional layer of protection can prevent unauthorized transactions, even if the user's primary credentials are compromised. A class that we have added to this abuse case would be the requestPin(). If a malicious user gets into the account, to possibly buy a ton of tickets to some events, the user would have to enter a pin to complete the transaction. Having this Pin class, would make sure that the malicious user could not get a hold of the tickets and possibly commit fraud on someone else's account.

The impact of security features on users can vary depending on how they are implemented and communicated. In general, implementing robust security features can have both positive and negative effects on users: enhanced trust: when users perceive that a system has strong security measures in place, they are more likely to trust the platform with their sensitive information. This trust can lead to increased user satisfaction and loyalty. Also, protection of personal data: Strong security features protect users' personal and financial data from

unauthorized access, theft, or misuse. This assurance is a significant benefit for users who are concerned about their privacy and data security. Finally, a negative impact would be the learning curve. Introducing new security measures may require users to learn new processes or adapt to changes in their interaction with the system. This learning curve can be perceived as a negative impact, particularly if the changes are not adequately communicated and explained.

In conclusion, designing security in a project is a needed endeavor that involves threat modeling and authorization controls. While addressing specific abuse cases like "Stolen Credentials" and "Pin Required to Reuse Card Stored in Account" is important, it's essential to consider other threats and vulnerabilities. By following these security design principles and continually updating security measures, projects can better protect user data and maintain proper continuity.