Business Vision

Purpose: Creating a mobile application that will be used as a ticketing system for all the College of Charleston events. This system will allow users to purchase and post College of Charleston events.

Overview: Implement software (to be determined) for College of Charleston events that interact with a database.

Technical requirements:

- Must be built using the Java (or equivalent) object oriented language.
- Must include unit tests for each applicable methods.

Must Have's (MVP):

- Secure login
    - Unique username
    - Password must be 15 characters and include lower case letter, upper case letter, number, and special character
- Functions of the system
    - Allow users to search for events
    - Allow users to purchase tickets for events, if they are available
    - Mock process payment
    - Display message of payment success
    - Update ticket inventory

Stretch goals:

- Add purchased tickets event date/time and info to the user's Google Calendar
- Allow users to choose their seats (if seated event)
- Update seat map

Summary

1. Register user: cofc email: username, password | cofc student involved
2. Log-in use case: username, password | backend, user involved
3. Find an event: User searches for an event: location, duration of the event, number of tickets available
4. Paying for event: payment available: cougar card, debits, credit card | enter card information and submit payment
5. Event cancelation: cancel, submit
6. Updating ticket availability

7. Contact us: issues including payment not going through, event canceled/refund, change of time and location
8. Manage profile: picture, contact info

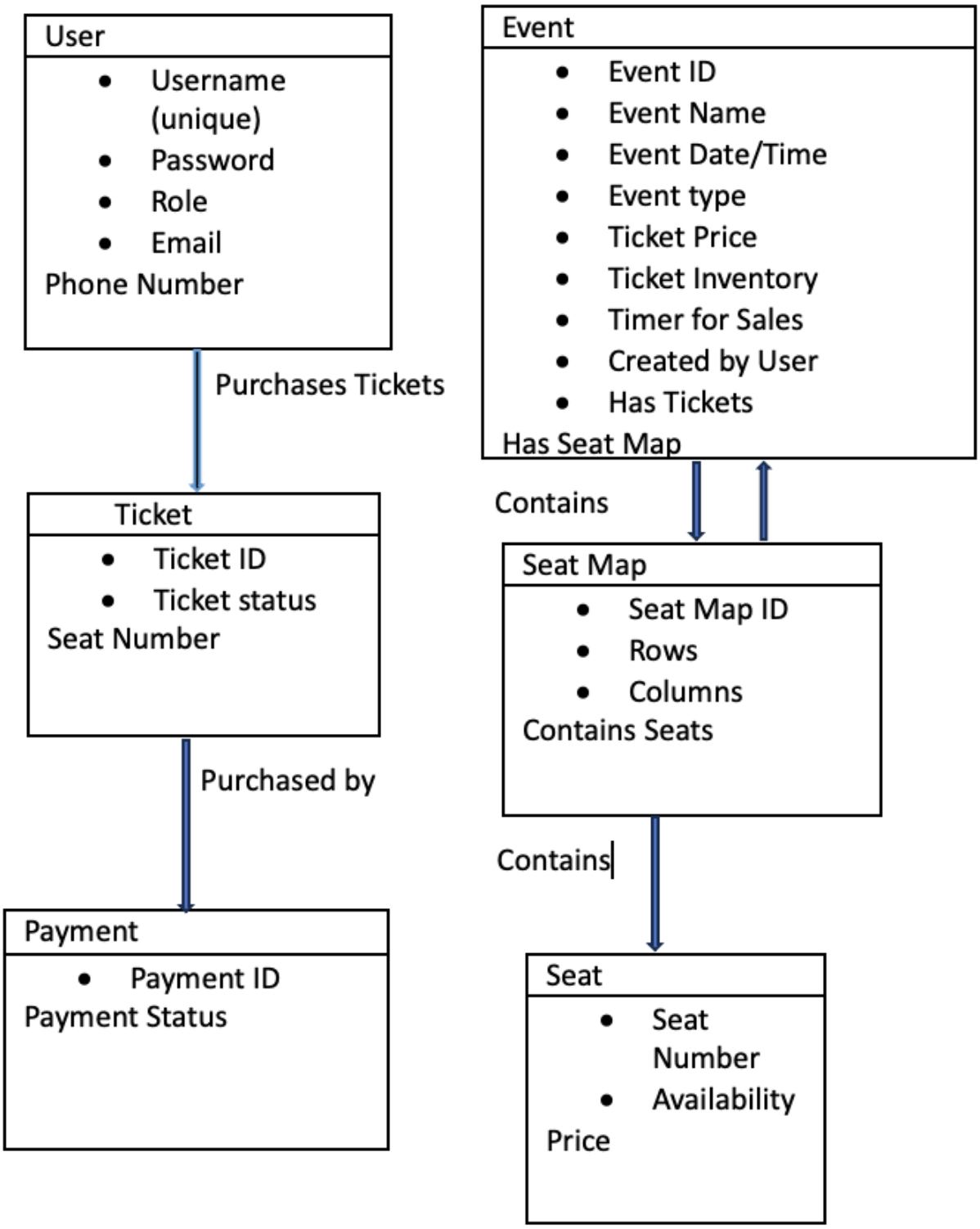
Fully Dressed Use Case #1

Use Case Name:	Checking Availability
Scope:	The system is under design
Level:	“User - Goal” or subfunction
Stakeholders and Interest:	People trying to find how many tickets are left
Primary Actor:	Any person trying to get tickets for a CofC event
Preconditions:	There must be tickets available for people to buy for the event
Success Guarantee:	If they have bought a ticket, then the counter of how many available tickets must go down
Main Success Guarantee:	Searching for a CofC event, visually showing the amount of tickets left, if available allow them to buy the ticket
Extensions:	A failure would involve when people are buying the ticket, the availability counter is not updating
Special Requirements:	Adding a timer for sales, showing how many tickets are left without clicking on the event, and pictures for the background of the event so that it has more appeal to the users.
Technology and Data Variations List:	Use PayPal to buy the tickets, Using Apple Pay instead of manually typing the numbers on the user’s card
Frequency of Occurrence:	Could be nearly continuous
Miscellaneous:	Open Issues (None at the moment, will update per project)

Fully Dressed Use Case #2

Use Case Name:	Purchase ticket
Scope:	The system is under design
Level:	“User - Goal” or subfunction
Stakeholders and Interest:	People trying to purchase tickets for a CofC event
Primary Actor:	Any person trying to purchase tickets for a CofC event
Preconditions:	The person must have a cougar card, debit card, credit card, PayPal account, or Apple Pay to purchase tickets.
Success Guarantee:	If they have bought the ticket then the screen will display a receipt
Main Success Guarantee:	The user will have the receipt of the purchased ticket saved on their in-app account
Extensions:	A failure would involve the transaction not going through due to insufficient funds or a program error
Special Requirements:	Adding a timer for sales, having the user select the number of tickets they are purchasing, and having the user enter their email and phone number while purchasing
Technology and Data Variations List:	Use PayPal to buy the tickets, Using Apple Pay instead of manually typing the numbers on the user’s card, and using CofC ID to get access to free tickets
Frequency of Occurrence:	Could be nearly continuous
Miscellaneous:	Open Issues (None at the moment, will update per project)

Domain Model



SUPPLEMENTARY SPECIFICATION

Revision History:

Version:	Date:	Description:	Author:
Inception Draft	September 14, 2023	First Draft To Be Refined	SMH + A

Introduction:

Document for all requirements not captured in the use cases

Functionality:

(Functionality common across many use cases)

Logging and Error Handling:

Log all errors to persistent storage

Security:

All usage requires you to be logged in using a username and a special password

Usability

Human Factors:

Customers trying to buy tickets for events at the College of Charleston should feel an easy way to access the tickets and know the price of them. Therefore:

- Text should be visible enough from 1 meter.
- Use colors that work with the different kinds of color blindness.
- App should follow standard design conventions; such as, well known button icons.

Allowing the UI to be very easy to follow will make it flow relatively well and the user will be able to check the availability of the tickets as well as buying the tickets.

Reliability

Recoverability:

In the event where there are issues with outside sources; such as, payment authorization, the team will use their own local solution to complete the sale of the ticket. (Figure out the specifics of this later)

Performance:

People purchasing tickets and checking the availability of tickets want to have the system work very quickly so that they can get the ticket and be done. We need the system to have the authorization be under a minute

Supportability

Adaptability:

Each user in the application has a different business rule depending on the need during the process of the sale and how the events are output into the system.

Configurability:

Unsure of configurations currently. Will be discussed soon.

Implementation Constraints:

Undecided on what technology the team will be using.

Purchased Components

None at the moment, will be changed in the future.

Free Open Source Components

Once the team has decided on the technology, the open source components of that software will be used to optimize the development process.

Interfaces

Noteworthy Hardware and Interface

Hardware:

- Touchscreen device (cell phone, tablet, laptop, etc.)
- Non-touchscreen devices (desktop, laptop, etc.)

Software Interfaces

Software Interface will be a web application that is run on a browser.

Application-Specific Domain (Business) Rules:

ID	Rule	Changeability	Source
RULE1	Purchaser discount rules. Examples: - CofC Employee 20% off. - CofC Alumna 10% off. - CofC Student free.	High Depending on the event and who it is made for the rules may vary.	CofC Event policy.
RULE2	Sale (transaction-level) discount rules: Applies to pre-tax total. Examples: - Buy a ticket, get 1 free for a friend. - Free ticket	High Each CofC event uses different rules depending on the kind of event, day, and hour.	CofC Event policy.

	after a certain amount of games attended.		

Legal Issues

The team will be using open source components to prevent licensing and legal restrictions to allow for the sale of College of Charleston events and tickets to events. All state tax rules will be applied, by law, but can change often.

Information in Domains of Interest

Pricing:

Pricing must be disclosed on the website prior to being purchased.

Credit and Payment Handling:

When the payment is process and approved, this payment is to then go to the seller and not the buyer.

Sales Tax:

Sales tax will be static and be added to every purchase.

GLOSSARY

analysis:

An investigation of a domain that results in describing its static and dynamic characteristics. It emphasizes questions of “what,” rather than “how.”

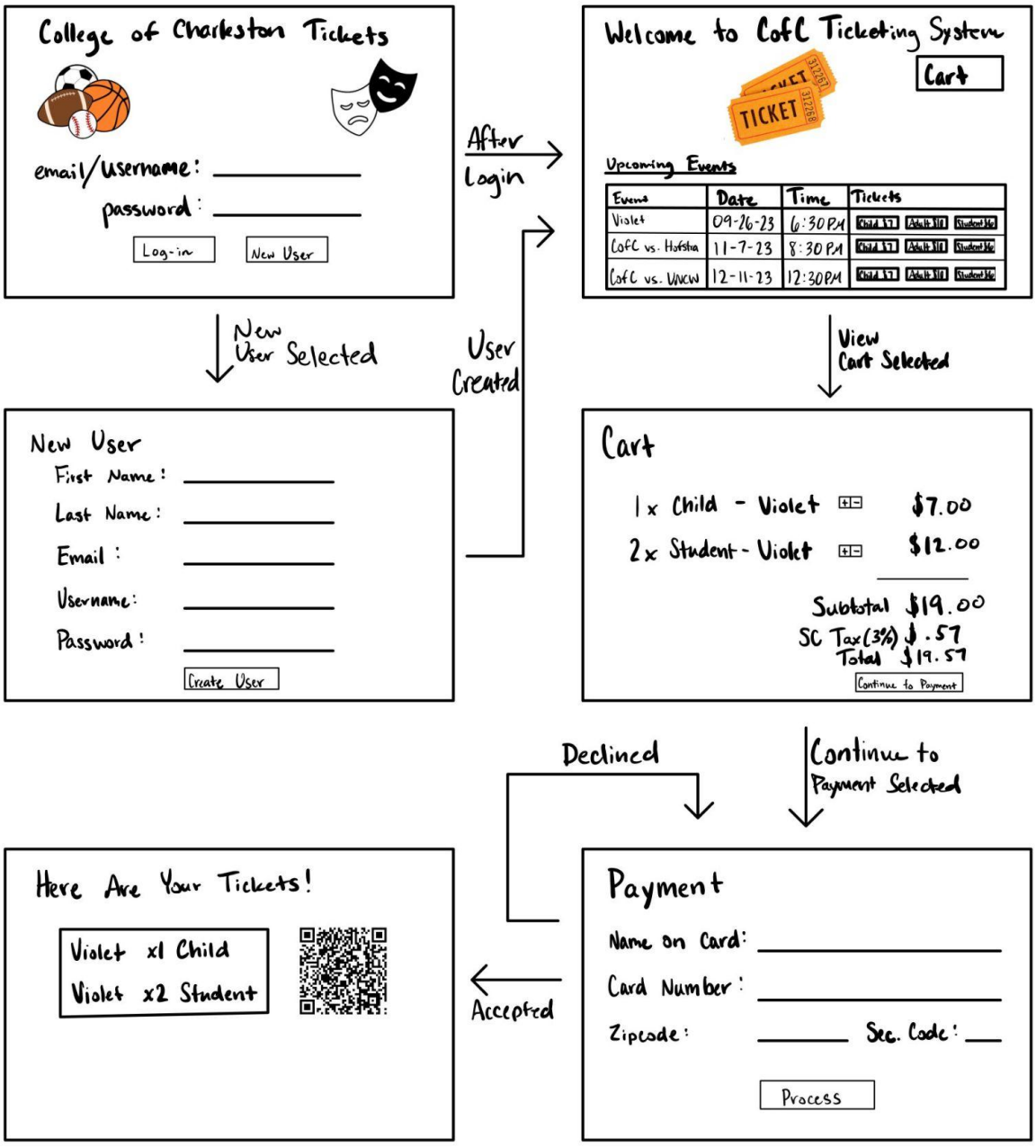
availability:

The quality or state of being available. In terms of the ticketing system, if there are tickets available for the desired event.

ticket:

A certificate or token showing that a fare or admission fee has been paid.

PROTOTYPE


College of Charleston Tickets
email/username:
password:
Log-in
New User
After Login
Welcome to CofC Ticketing System
Cart
TICKET
Upcoming Events
Event
Date
Time
Tickets
Violet
09-26-23
6:30PM
Child $7
Adult $10
Student $6
CofC vs. Hofstra
11-7-23
8:30PM
CofC vs. UNCW
12-11-23
12:30PM
New User Selected
User Created
View Cart Selected
New User
First Name:
Last Name:
Email:
Username:
Password:
Create User
Cart
1 x Child - Violet
$7.00
2 x Student - Violet
$12.00
Subtotal $19.00
SC Tax (3%) $.57
Total $19.57
Continue to Payment
Declined
Continue to Payment Selected
Here Are Your Tickets!
Violet x1 Child
Violet x2 Student
Accepted
Payment
Name on Card:
Card Number:
Zipcode:
Sec. Code:
Process

TP D3

Fully Dressed Use Case #3

Use Case Name:	Login Accessibility
Scope:	The system is under design
Level:	“User - Goal” or subfunction
Stakeholders and Interest:	People trying to login to find/get tickets for a CofC event
Primary Actor:	Any person logging in to find/purchase tickets for a CofC event
Preconditions:	The person must have a unique username and a password that must be 15 characters and include lower case letter, upper case letter, number, and special character
Success Guarantee:	If login is successful, then the user will be able to find events on the homepage and possibly purchase tickets
Main Success Guarantee:	If they have fulfilled the necessary requirements for a unique username and a password with all of the following conditions, then they will be accepted into the system
Extensions:	A failure would include them not having all the requirements for the password and/or the username is already taken
Special Requirements:	Using Face ID to sign into your account
Technology and Data Variations List:	Using Face ID to sign into your account and Fingerprint to sign into your account
Frequency of Occurrence:	Could be nearly continuous
Miscellaneous:	Open Issues (None at the moment, will update per project)

Operations Contract: Check ticket availability

Operation	searchEvent(eventName: String)
Cross Reference	Check ticket availability
Pre-conditions	The user must be logged in and on the homepage of the application. The user must have either selected or typed in the specific event they are looking for.
Post-conditions	If they have fulfilled the necessary requirements for searching for the event, then the user will be able to select the event on the application.

Operation	searchTickets(ticketID: String)
Cross Reference	Check ticket availability
Pre-conditions	There is a user, user is logged in, user searched for an event, and has selected the event of their choosing
Post-conditions	Ticket info is updated and returned if there are tickets available.

Operation	returnTickets()
Cross Reference	Check ticket availability
Pre-conditions	The ticket info
Post-conditions	The ticket info and availability is sent and displayed to the user

Operations Contract: Login Accessibility

Operation	registerUserr()
Cross Reference	Creating a login for a user (login accessibility)
Pre-conditions	The person must have a unique username and a password that must be 15 characters and include lower case letter, upper case letter, number, and special character
Post-conditions	If they have fulfilled the necessary requirements for a unique username and a password with all of the following conditions, then they will be accepted into the system

Operation	loginUser()
Cross Reference	The system has accepted the user into the system (login accessibility)
Pre-conditions	The person must have entered their unique username and password into the system.
Post-conditions	If they have fulfilled the necessary requirements for their own unique username and a password with all of the following conditions, then they will be accepted into the application.

Operations Contract: Purchase tickets

Operation	lookUpTicket(event tickets : string)
Cross Reference	Purchase tickets
Pre-conditions	The user has logged in, the system is running, and the ticket provided is valid and corresponds to an existing ticket.
Post-conditions	If the operation is successful and a valid ticket is found. The ticket details contain detailed information about the ticket that will be returned.

Operation	displayTickets (list of Tickets: string) Ticket list (Ticket ID, Event Name, Ticket Price, Availability)
Cross Reference	Purchase tickets
Pre-conditions	the system is running and accessible,the user looks up for tickets with valid event names.

Post-conditions	If the operation is successful, the tickets list contains information about available tickets which is Ticket list (Ticket ID, Event Name, Ticket Price, Availability). But if the operation is not successful it will display a message stating no availability, try another event name or date for example.

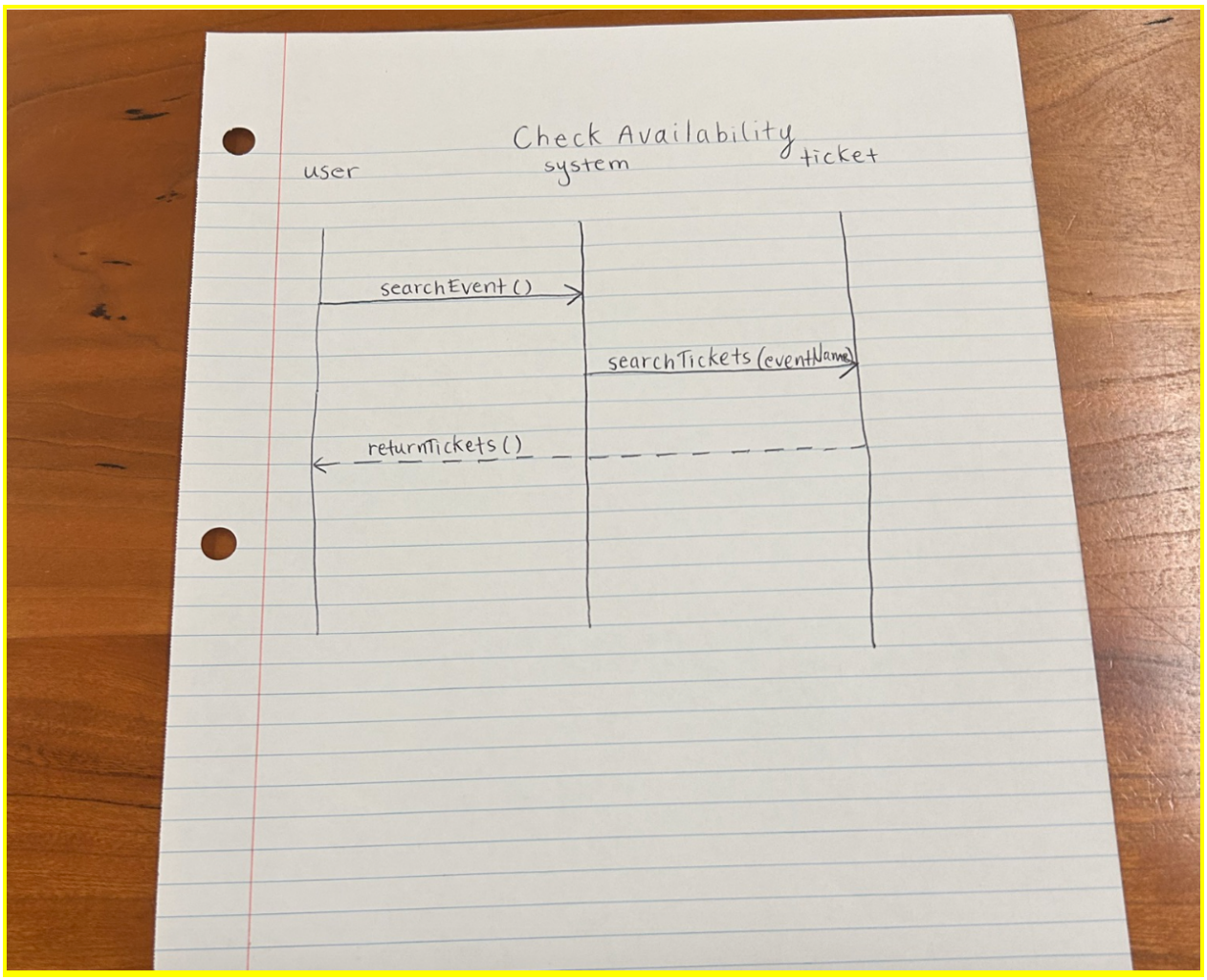
Operation	AddtoCart (ticketID: int)
Cross Reference	Purchase tickets
Pre-conditions	The TicketID provided exists in the database of available tickets. The user is authorized to modify the specified shopping cart. - The ticket is not already in the user's cart (unless the system supports multiple quantities of the same ticket in a cart).
Post-conditions	If the operation is successful, the specified ticket is added to the user's shopping cart.

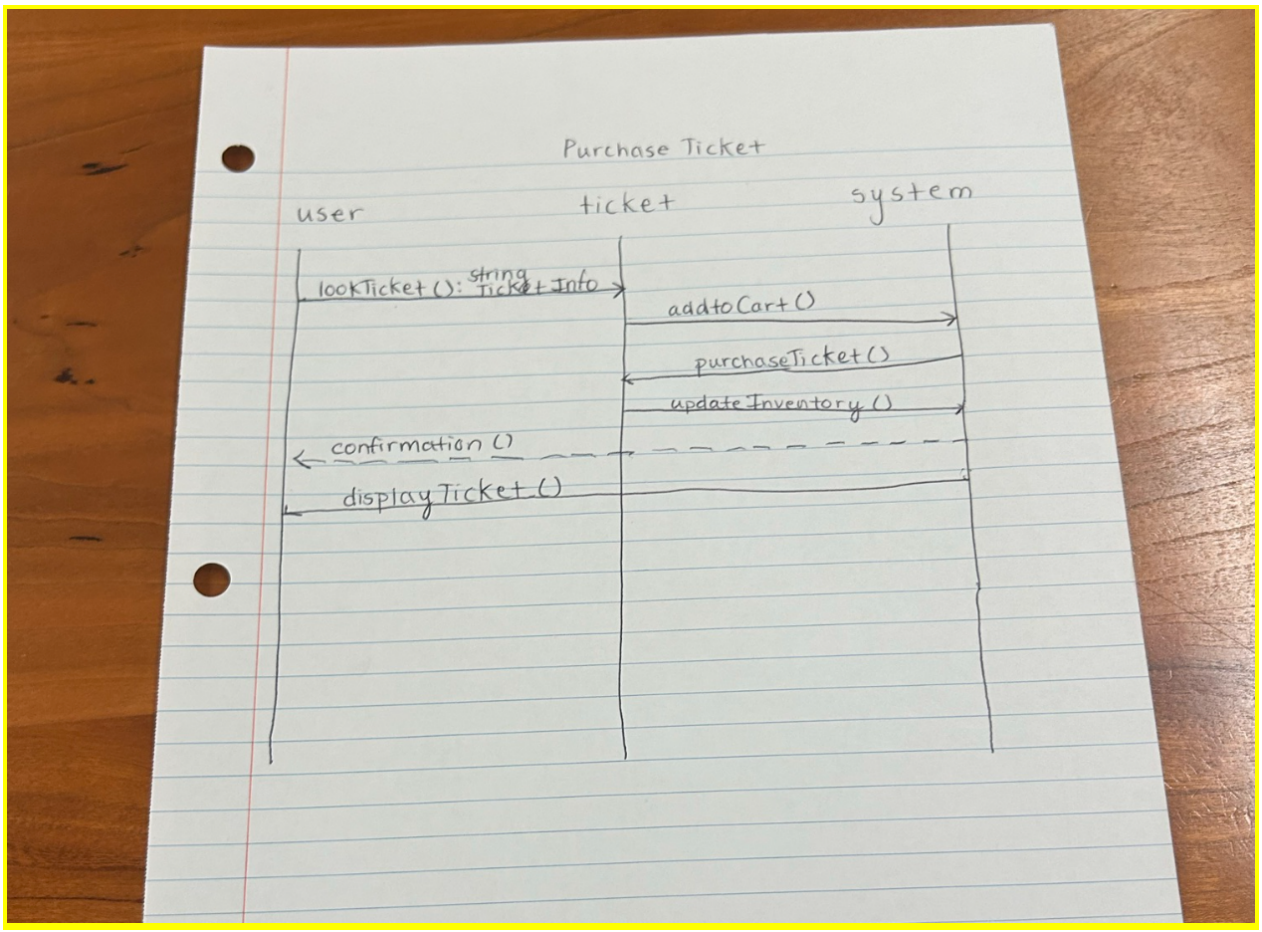
Operation	PurchaseTicket(ticketID: int)
Cross Reference	Purchase ticket
Pre-conditions	The user has logged in, added tickets to their cart, and then selected “Purchase Tickets” then the ticket will be saved in the system as ready for purchase.
Post-conditions	The tickets added to cart will be ready for

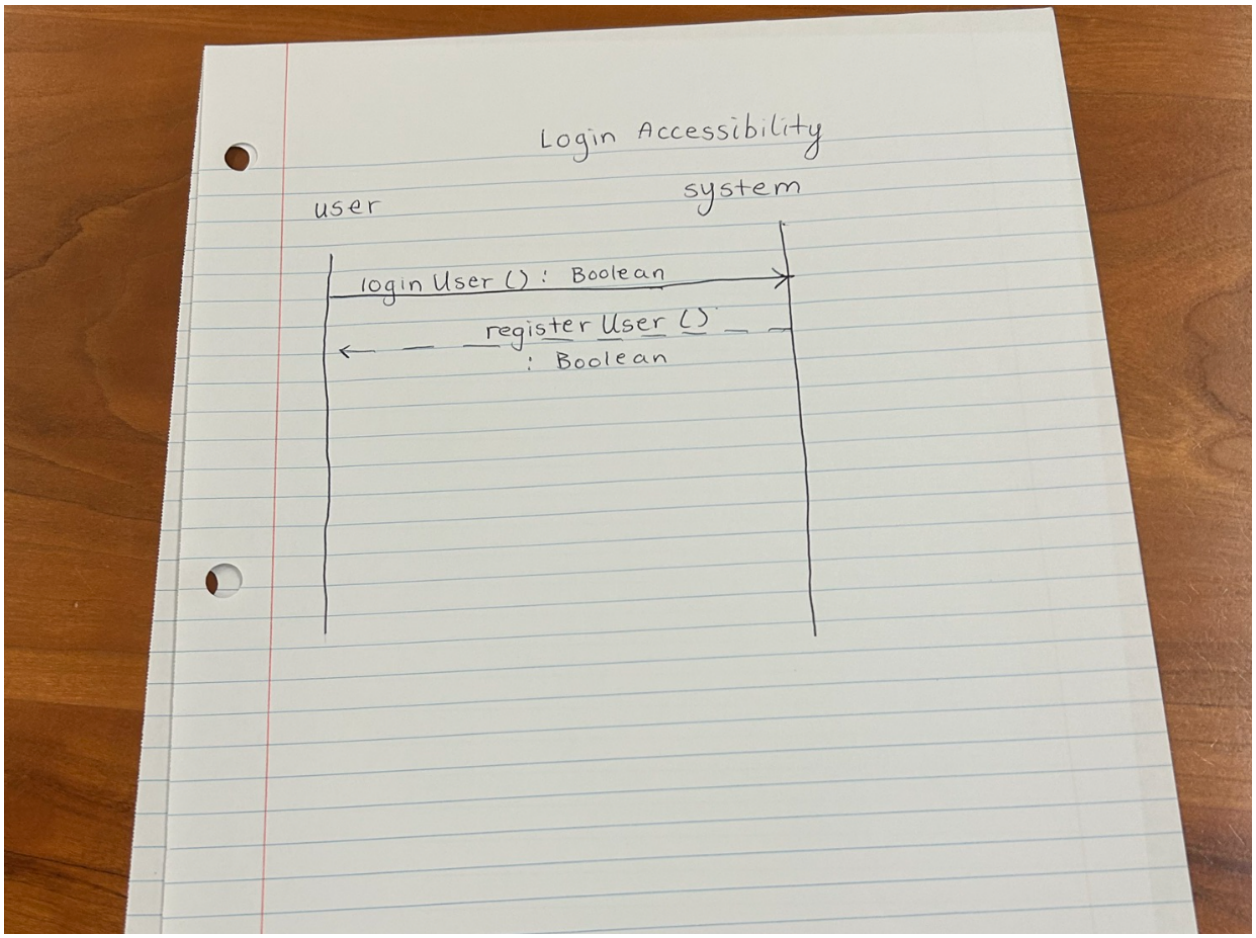
	purchase

Operation	confirmation () return (confirmationID: int)
Cross Reference	Purchase tickets
Pre-conditions	The system is running. The Transaction provided is valid and corresponds to an existing, pending transaction, order, or action.
Post-conditions	If the operation is successful, the specified payment transaction is marked as completed.

Operation	UpdateInventory(ticketID: int)
Cross Reference	Purchase tickets
Pre-conditions	The user has purchased Tickets, and the ticket inventories need to be updated.
Post-conditions	The ticket inventories are updated and are confirmed.


Check Availability
user
system
ticket
searchEvent()
searchTickets(eventName)
returnTickets()


Purchase Ticket
user
ticket
system
lookTicket(): string TicketInfo
addtoCart()
purchaseTicket()
updateInventory()
confirmation()
displayTicket()


Login Accessibility
user
system
loginUser() : Boolean
registerUser()
: Boolean