

## **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

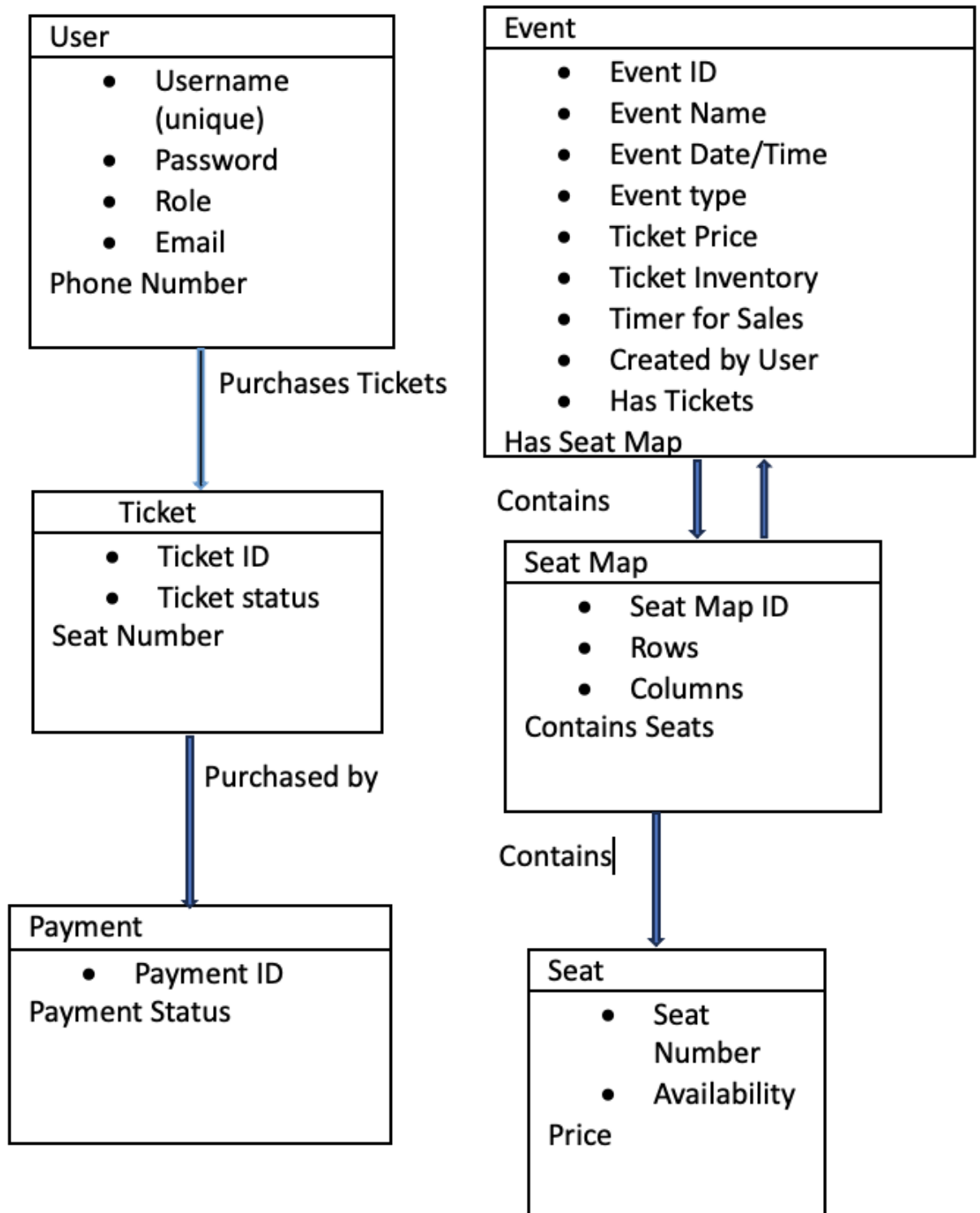
### Fully Dressed Use Case #1

<b>Use Case Name:</b>	Checking Availability
<b>Scope:</b>	The system is under design
<b>Level:</b>	“User - Goal” or subfunction
<b>Stakeholders and Interest:</b>	People trying to find how many tickets are left
<b>Primary Actor:</b>	Any person trying to get tickets for a CofC event
<b>Preconditions:</b>	There must be tickets available for people to buy for the event
<b>Success Guarantee:</b>	If they have bought a ticket, then the counter of how many available tickets must go down
<b>Main Success Guarantee:</b>	Searching for a CofC event, visually showing the amount of tickets left, if available allow them to buy the ticket
<b>Extensions:</b>	A failure would involve when people are buying the ticket, the availability counter is not updating
<b>Special Requirements:</b>	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.
<b>Technology and Data Variations List:</b>	Use PayPal to buy the tickets, Using Apple Pay instead of manually typing the numbers on the user’s card
<b>Frequency of Occurrence:</b>	Could be nearly continuous
<b>Miscellaneous:</b>	Open Issues (None at the moment, will update per project)

## Fully Dressed Use Case #2

<b>Use Case Name:</b>	Purchase ticket
<b>Scope:</b>	The system is under design
<b>Level:</b>	“User - Goal” or subfunction
<b>Stakeholders and Interest:</b>	People trying to purchase tickets for a CofC event
<b>Primary Actor:</b>	Any person trying to purchase tickets for a CofC event
<b>Preconditions:</b>	The person must have a cougar card, debit card, credit card, PayPal account, or Apple Pay to purchase tickets.
<b>Success Guarantee:</b>	If they have bought the ticket then the screen will display a receipt
<b>Main Success Guarantee:</b>	The user will have the receipt of the purchased ticket saved on their in-app account
<b>Extensions:</b>	A failure would involve the transaction not going through due to insufficient funds or a program error
<b>Special Requirements:</b>	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
<b>Technology and Data Variations List:</b>	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
<b>Frequency of Occurrence:</b>	Could be nearly continuous
<b>Miscellaneous:</b>	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 device (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: <ul style="list-style-type: none"><li>- CofC Employee 20% off.</li><li>- CofC Alumna 10% off.</li><li>- CofC Student free.</li></ul>	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: <ul style="list-style-type: none"><li>- Buy a ticket, get 1 free for a friend.</li><li>- Free ticket after a certain amount of games attended.</li></ul>	High  Each CofC event uses different rules depending on the kind of event, day, and hour.	CofC Event policy.

### 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

