Vision :

To implement a ticketing system with a working website that connects to the CofC events database, and allows users to purchase tickets for events. The system will then email the user a receipt of their purchase. The system will:

* Require a secure login & database
  + Unique login with 15 character password
* Allow event goers to search for events & purchase tickets
* Process payments & email receipts
* Update the current ticket inventory
  + if time permits allow an event goer to reserve a seat.

USE CASES :

Use Case UC1:

Scope: Web-based

Level: User-goal

Primary Actor: Eventgoer

Stakeholders and Interest:

-Eventgoer: Wants to attend events, having a website that shows all of CofC events will allow them to attend them more efficiently.

-Students at College of Charleston: Students want to know what is happening at CofC and where they can buy their tickets.

-College of Charleston: Want to sell their tickets, and maximize capacity at events

-People who sell tickets: They want people to buy their tickets efficiently and at maximum capacity.

-Players: Want people to attend their events to show support and help fund their sport programs.

-Alumni: Want to know when events are happening and have easier access to buy them. Allows them to form groups easier to attend events.

-Fan interested in the teams playing: Want to see when their favorite teams are playing, and buy tickets to attend those events.

Precondition: Eventgoer hasn't made an account on website.

Success guarantees: Eventgoer information is saved. Able to see events happening and buy tickets. Able to use their Stripe account to buy tickets.

Main Success Scenario (or Basic Flow):

1. Eventgoer enters website.
2. Eventgoer clicks on "Make Account".
3. Prompt pops up, allows them to put in a email and password.
4. Eventgoer clicks on finish.
5. Verification email is sent.
6. Eventgoer responds to email, account is verified.
7. Eventgoer is able to go back to website, and login in.
8. Able to browse events and click "buy tickets" for events.

Extensions (or Alternative Flows):

1. Eventgoer goes to website to buy tickets.
2. Eventgoer selects an event he wants to attend, and selects to buy tickets.
3. Gets prompted to make and account or continue as guest.
4. Eventgoer clicks on make account. Gets prompted to enter email and password.
5. Enter username and password, and submits account.
6. Answers verification email.
7. Eventgoer goes back to original website and logins.

UC2

Scope: Web-based

Level: User-goal

Primary Actor: Fan interested seeing his team playing.

Stakeholders and Interest:

-Eventgoer: Wants to attend events, having a website that shows all of CofC events will allow them to attend them more efficiently.

-Students at College of Charleston: Students want to know what is happening at CofC and where they can buy their tickets.

-College of Charleston: Want to sell their tickets, and maximimize capacity at events

-People who sell tickets: They want people to buy their tickets efficiently and at maximum capacity.

-Players: Want people to attend their events to show support and help fund their sport programs.

-Alumni: Want to know when events are happening and have easier access to buy them. Allows them to form groups easier to attend events.

-Fan interested in the teams playing: Want to see when their favorite teams are playing, and buy tickets to attend those events.

Precondition: Eventgoer has an already existing validated account.

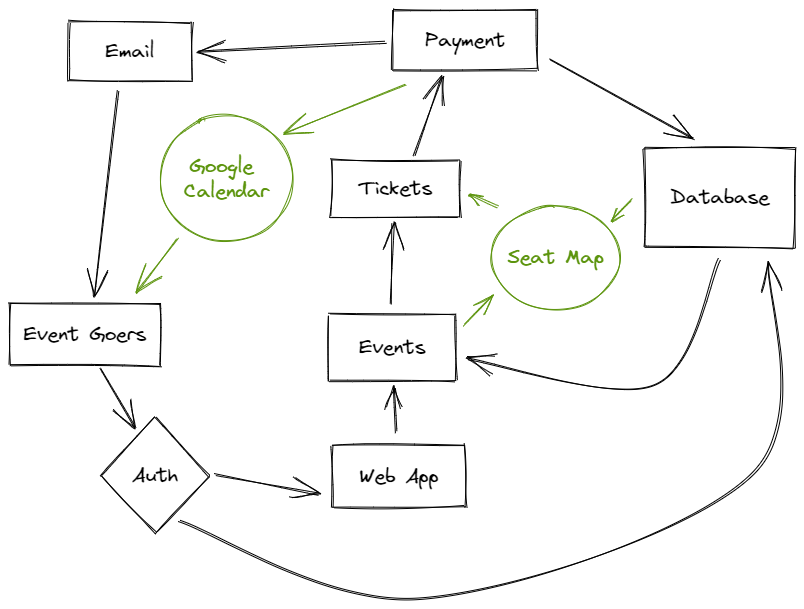
Successes Guarantee: They are able to find their favorite team and buy tickets to attend those events.

Main Success Scenario:

1. Fan enters website.
2. Fan clicks to login and enter their credentials to login.
3. Fan logins. Fan goes to search for their team.
4. Fan finds their favorite team and dates they'll be playing.
5. Fan selects "Add tickets to cart".
6. Fan clicks on cart, and then checkout.
7. Shows fan tickets they are buying, over-all cost, and asking to select Stripe to pay for it.
8. Fan selects Strips, verifies credentials, and checks out.
9. Sales is completed, fan gets more ticket information via email. Fan is happy.

Extensions:

1. Fan enters website.
2. Fan tries to search for team.
3. Team doesn't exist. Gets prompted showing their is no matches.
4. Fan manually searchers for team, by clicking on calendar.
5. Finds team. Buys tickets and is happy.



SUPPLEMENT SPECS

### **Revision History**

Ver: Inception Draft - Sept 14th, 2022 - First draft. To be refined - John Lloyd

### **Introduction**

This document is the repository of all (Name to be determined) requirements not captured in the use cases.

### **Functionality**

***Logging and Error Handling*** Log errors for handling bugs / security issues ***Authentication*** Users will need to be able to securely login ***Payment*** Users will need a secure form of payment - To keep the app from needing to be PCI compliant, a secondary payer such as Stripe or Paypal can be used.

### **Usability**

***Human Factors*** Users may be visually impaired. Therefore:

* Critical information such as pricing, times, and seats should be very visible.
* Color blindness should be taken in account when designing Users may often not pay attention to their choices - Validations should be used for things like seating and dates.

### **Reliability**

A Blue/Green app deployment strategy can be employed to easily roll back if a new version of an application fails.

### **Performance**

The application should be scalable for larger or unexpected events. Auto scaling could be used to reduce costs for small events, but increase bandwidth for large events. More analysis is needed to determine the best tools for scalability.

### **Supportability**

The application should be small enough that bugs or updates could be pushed out quickly. As stated in Reliability, Blue/Green deployment can be used to make quick roll outs or roll backs.

### **Implementation Constraints**

We are unsure of the API for the college's event database. This will determine what API Fetch methods we need to use.

### **Purchased Components**

Some SaaS may be used such as Stripe or AWS Amplify.

### **Free open Source Components**

React libraries

### **Software Interfaces**

Colleges event management database, AWS, Stripe/Paypal.

### **Legal Issues**

To avoid needing to be PCI compliant, we will not be storing the users credit card information. We will also be using AWS amplify for masking the users private information such as username, passwords, and emails. Taxes may need to be calculated on sales. This information needs to be analysed.

GLOSSERY

**AWS (Amazon Web Services)** - Suite of cloud computing software used for web application development, hosting, and security.

**Blue/Green Deployment** - Method of deploying web applications, two instances of hte application are run at the same when deploying a new build. If the new build fails, users can easily be redirected to the old build with no downtime.

**Eventgoer** - A term used for our users, who plan to purchase tickets for an event.

**SaaS (Software as a Service)** - A type of software that is charged by subscription.