Food Truck Events: A Reservation System

by S. Tessa Toney

4 February 2021

This Selection

I chose a reservation system because I had been thinking about incorporating the ability to make a reservation via a calendar of available timeslots into my 'YesToJess' project and I wanted to take a closer look at how to do that. I absolutely love the dynamic nature of code and the way it allows you to make multiple changes across the site with only a change in only one or two pieces of code or simply by adding new information. The reservation site provides a place for people of multiple positions but common interests to organize and communicate effectively and plan without having to sync up their personal schedules to accomplish it. This project was a great learning experience for me because it allowed me to practice looking at real world situations, anticipating what problems may arise from them, and find a way to help prevent them through coding.

Food Truck Events represents a company which owns multiple locations that are used for event hosting. These locations range from fields to convention centers and are in multiple states. The website allows Food Truck vendors to register, add their trucks, and schedule their trucks at upcoming events with open slots. This is helpful to the company because they don't have to track down vendors and helpful to the vendors because they don't have to spend as much time searching out hungry customers.

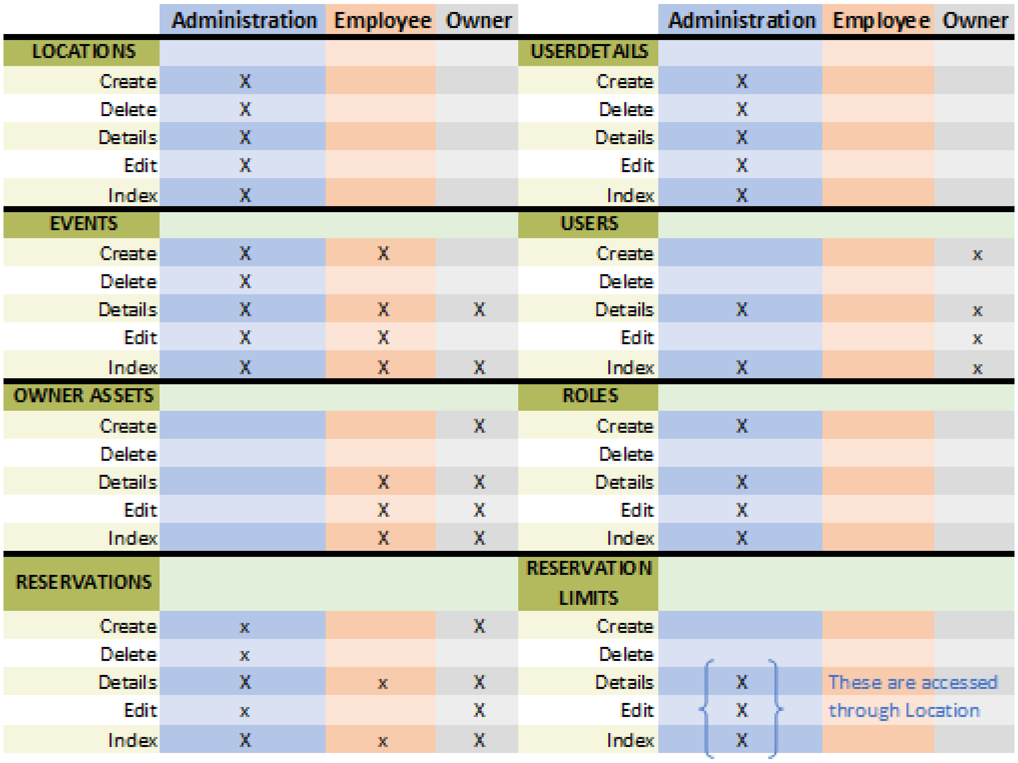
Project Requirements

Requirements for this project included a target audience of Administration, Employees, and Vendors who use the internal functions of the site for coordinating who will be a which event on what day. Clientele – particularly vendors – are extremely busy and will be accessing the site primarily through their mobile devices making a mobile friendly design and easy to use design imperative. Vendors must be able to add and schedule their own trucks only. Employees must be able to see all reservations based on location. Administrators must have the ability to schedule a reservation even if the limit for the location has been reached.

The Design & Build Process

I began the design process by creating a use case to clarify who would need access to which parts of the project and which parts to lock down. When looking more deeply at the roles I decided it was best to add a super user with full access. This allowed me to lock down areas of the site to provide privacy and security while still allowing for the necessity of changes that are unavoidable.

The first step was to implement a database schema which contained the relationships between the tables and made it easier to see how the information would flow from one area to another. This was followed by the creation of the use case scenario. I started to think about some of the problems that might come up when having owners register their own food trucks while also having Administrators with the ability to make these same reservations. Because it is not possible to have the same food truck in two places at once, I noted that it would be necessary to prevent duplicate reservations, not only for the same truck in the same location, but for the same truck on the same date in order to prevent human error and double booking. Another safety measure taken was to allow scheduling by a date associated with an event as opposed to allowing scheduling by choosing the date from a calendar yourself. This ensures that the date is in fact available and an event is occurring on that date.

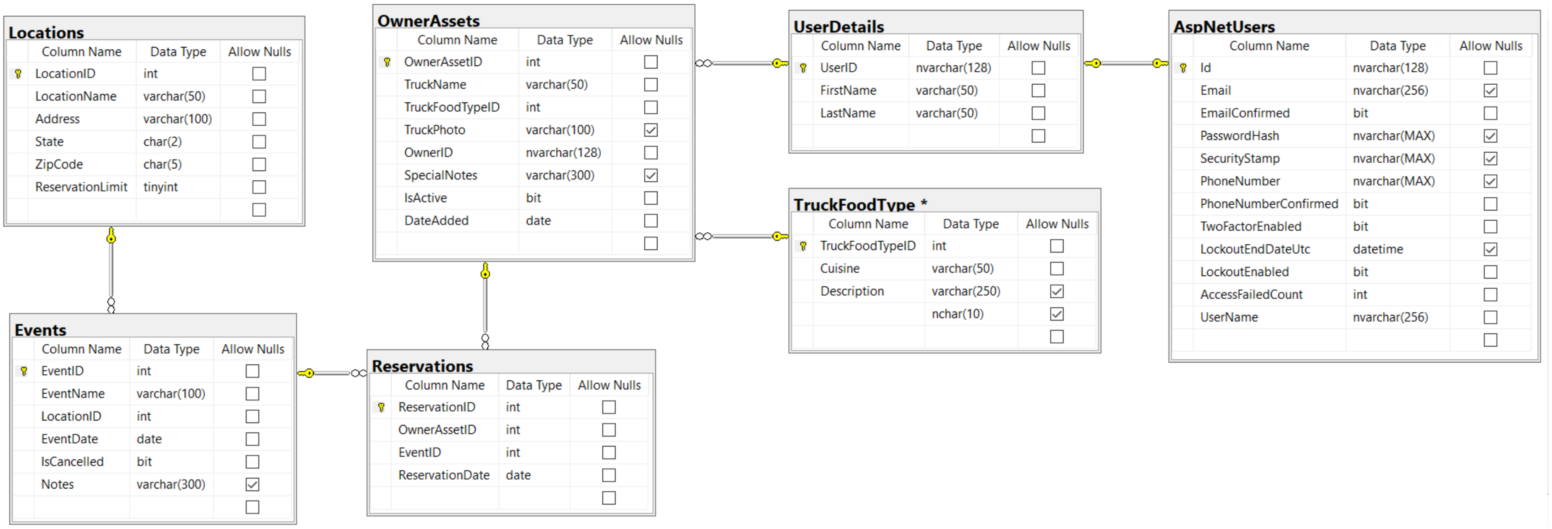


	Administration	Employee	Owner		Administration	Employee	Owner
LOCATIONS				USERDETAILS			
Create	X			Create	X		
Delete	X			Delete	X		
Details	X			Details	X		
Edit	X			Edit	X		
Index	X			Index	X		
EVENTS				USERS			
Create	X	X		Create			x
Delete	X			Delete			
Details	X	X	X	Details	X		x
Edit	X	X		Edit			x
Index	X	X	X	Index	X		x
OWNER ASSETS				ROLES			
Create			X	Create	X		
Delete				Delete			
Details		X	X	Details	X		
Edit		X	X	Edit	X		
Index		X	X	Index	X		
RESERVATIONS				RESERVATION LIMITS			
Create	x		X	Create			
Delete	x			Delete			
Details	X	x	X	Details	X	These are accessed	
Edit	x		X	Edit	X	through Location	
Index	X	x	X	Index	X		

*** ITAdmin : role has universal access

***Employees have create/edit access only to notes in OwnerAssets

FIGURE 1 : USE CASE



Locations

Column Name	Data Type	Allow Nulls
LocationID	int	☐
LocationName	varchar(50)	☐
Address	varchar(100)	☐
State	char(2)	☐
ZipCode	char(5)	☐
ReservationLimit	tinyint	☐
		☐

OwnerAssets

Column Name	Data Type	Allow Nulls
OwnerAssetID	int	☐
TruckName	varchar(50)	☐
TruckFoodTypeID	int	☐
TruckPhoto	varchar(100)	☑
OwnerID	nvarchar(128)	☐
SpecialNotes	varchar(300)	☑
IsActive	bit	☐
DateAdded	date	☐
		☐

UserDetails

Column Name	Data Type	Allow Nulls
UserID	nvarchar(128)	☐
FirstName	varchar(50)	☐
LastName	varchar(50)	☐
		☐

AspNetUsers

Column Name	Data Type	Allow Nulls
Id	nvarchar(128)	☐
Email	nvarchar(256)	☑
EmailConfirmed	bit	☐
PasswordHash	nvarchar(MAX)	☑
SecurityStamp	nvarchar(MAX)	☑
PhoneNumber	nvarchar(MAX)	☑
PhoneNumberConfirmed	bit	☐
TwoFactorEnabled	bit	☐
LockoutEndDateUtc	datetime	☑
LockoutEnabled	bit	☐
AccessFailedCount	int	☐
UserName	nvarchar(256)	☐
		☐

TruckFoodType *

Column Name	Data Type	Allow Nulls
TruckFoodTypeID	int	☐
Cuisine	varchar(50)	☐
Description	varchar(250)	☑
	nchar(10)	☑
		☐

Events

Column Name	Data Type	Allow Nulls
EventID	int	☐
EventName	varchar(100)	☐
LocationID	int	☐
EventDate	date	☐
IsCancelled	bit	☐
Notes	varchar(300)	☑
		☐

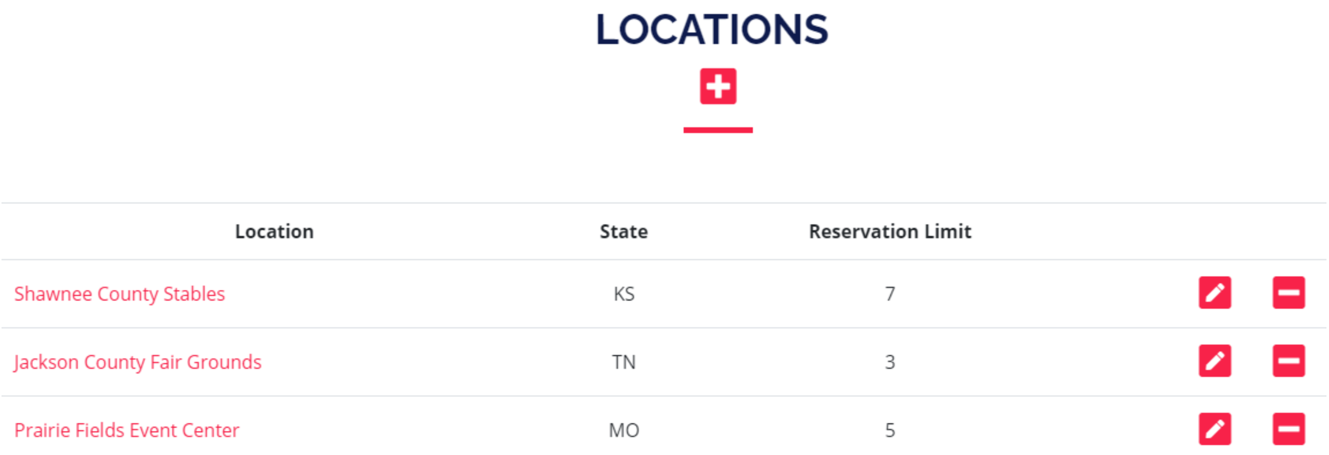
Reservations

Column Name	Data Type	Allow Nulls
ReservationID	int	☐
OwnerAssetID	int	☐
EventID	int	☐
ReservationDate	date	☐
		☐

FIGURE 2 DATABASE SCHEMA

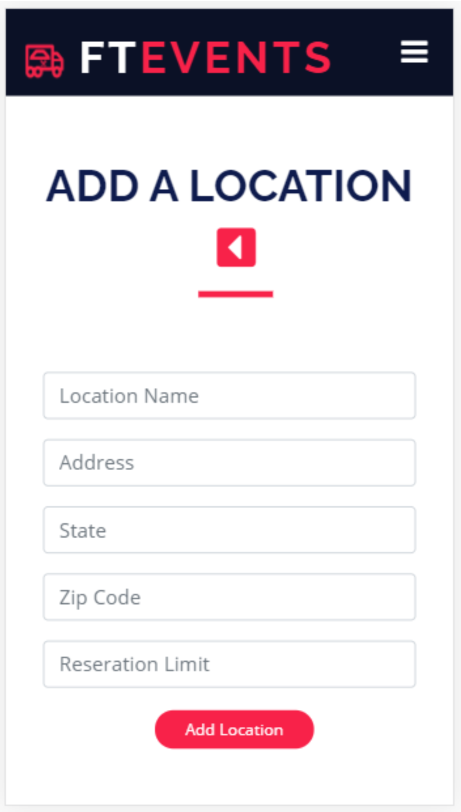
UI Design

Because the design needed to be mobile friendly and easy to use, I chose to use favicons that resembled buttons commonly seen in phone apps as opposed to links with written labels. A plus sign for adding, a minus sign for subtracting, and a pencil for editing are easier to look at and visually process quickly. Clicking on the link to get more information is a normal expectation.



LOCATIONS

Location	State	Reservation Limit		
Shawnee County Stables	KS	7	✎	−
Jackson County Fair Grounds	TN	3	✎	−
Prairie Fields Event Center	MO	5	✎	−



ADD A LOCATION

Location Name

Address

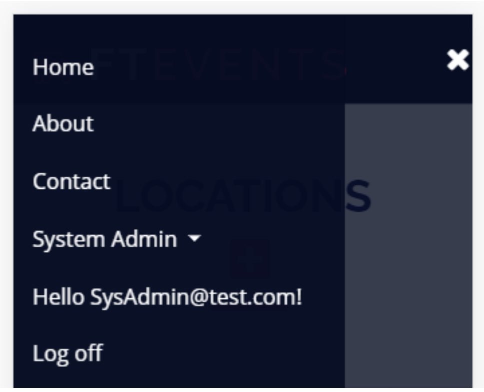
State

Zip Code

Reseration Limit

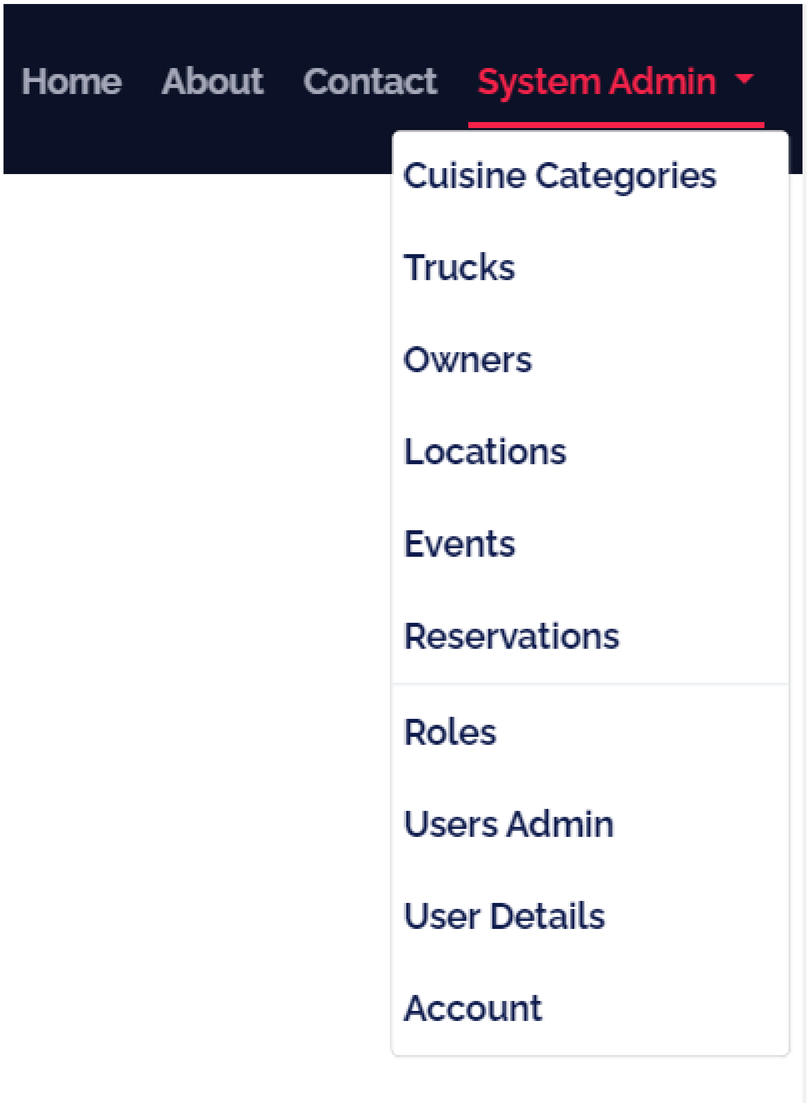
This design feature was continued in the create, edit, and delete views with the use of an arrow pointing left which commonly represents the go back feature on many apps. I chose to use placeholders instead of labels on many of the screens because this used the space of a small screen more effectively and allowed all the fields to be shown without requiring a user to scroll to get to the rest.

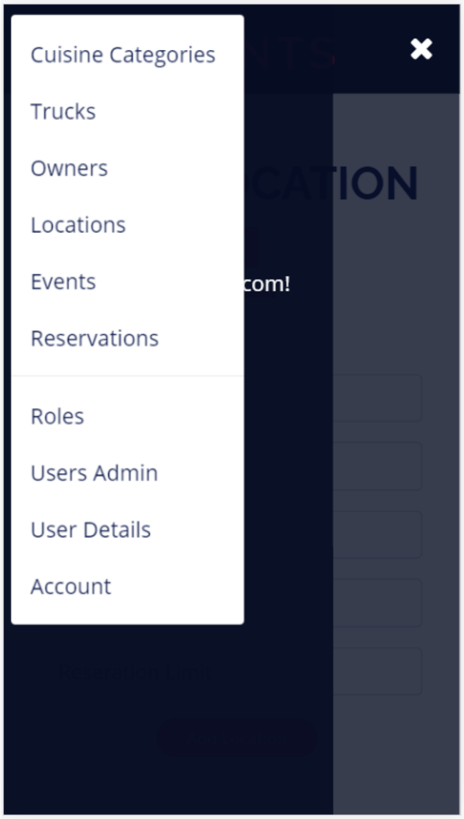
Specialized menus are provided for each user based on their role. Each role has a dropdown menu that will appear in the navigation upon logging in. Using a dropdown this way allows the user to see all necessary options without having to scroll even when in the mobile view. This dropdown also helps to condense the number of links showing in the nav on a full screen allowing for a cleaner look.

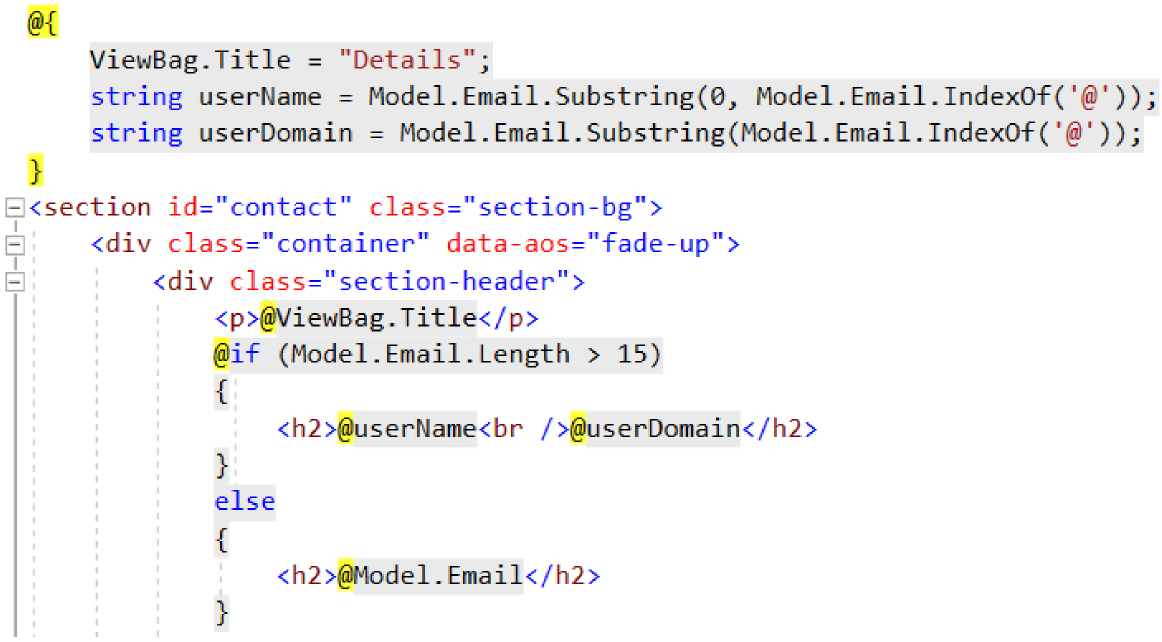


Left: Mobile View

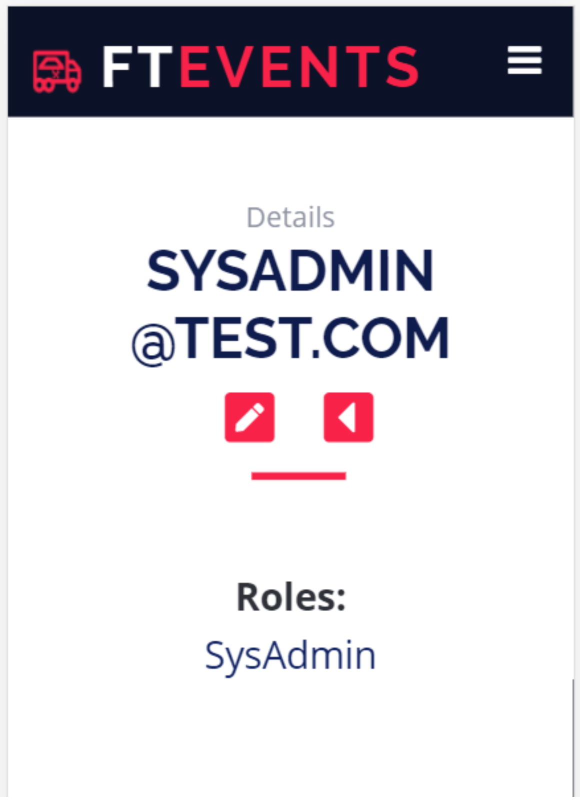
Below: Desktop View







```
@{
    ViewBag.Title = "Details";
    string userName = Model.Email.Substring(0, Model.Email.IndexOf('@'));
    string userDomain = Model.Email.Substring(Model.Email.IndexOf('@'));
}
<section id="contact" class="section-bg">
    <div class="container" data-aos="fade-up">
        <div class="section-header">
            <p>@ViewBag.Title</p>
            @if (Model.Email.Length > 15)
            {
                <h2>@userName<br />@userDomain</h2>
            }
            else
            {
                <h2>@Model.Email</h2>
            }
```



Details

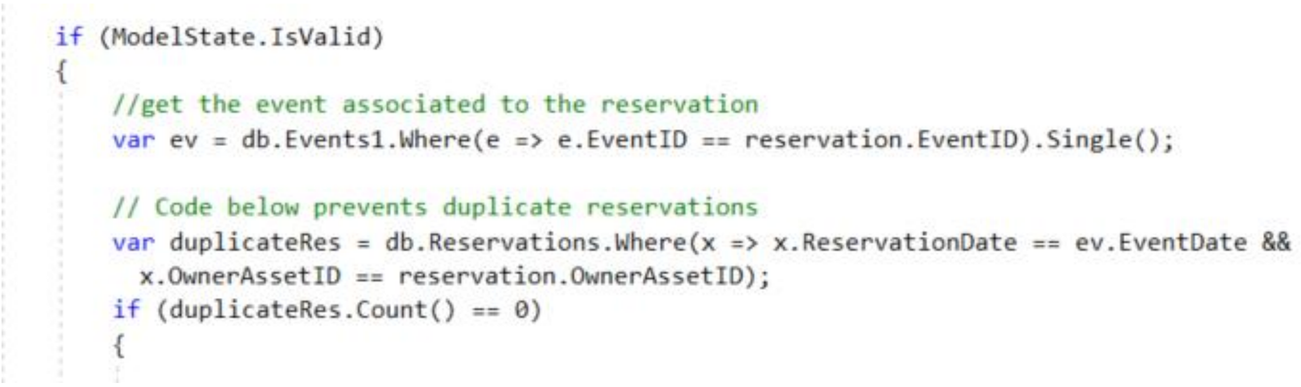
SYSADMIN
@TEST.COM

Roles:
SysAdmin

Size restrictions associated with a mobile screen also encouraged me to take some extra steps in the UI by using the C# `substring()` to effectively break up the email address of the user in a way that makes sense instead of relying on the CSS property of `word-break: break-word;` which chooses an arbitrary breakpoint based on space available. This allowed me to choose to separate the email address of the logged-in user by the @ symbol and split the two pieces onto two lines in a way that is visually appealing when the email is longer than 15 characters as noted on the right.

Functionality

It was important to try to prevent human error where possible when considering these vendors would be scheduling while on the run and on their mobile devices. There were three things that needed to be implemented to be sure trucks were not double-booking or showing up to events where there would be no space available. All of this code was written in the Reservations Controller.

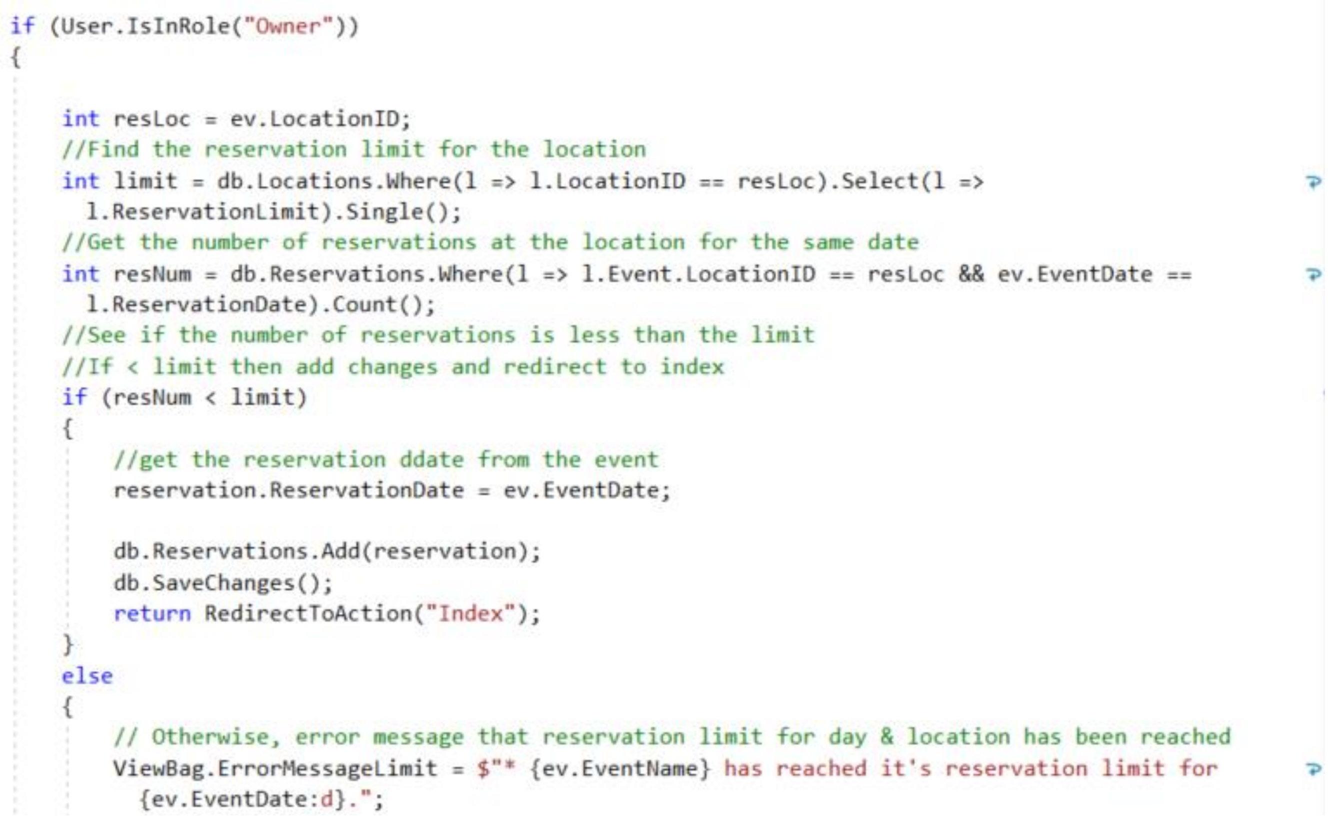


```
if (ModelState.IsValid)
{
    //get the event associated to the reservation
    var ev = db.Events1.Where(e => e.EventID == reservation.EventID).Single();

    // Code below prevents duplicate reservations
    var duplicateRes = db.Reservations.Where(x => x.ReservationDate == ev.EventDate &&
      x.OwnerAssetID == reservation.OwnerAssetID);
    if (duplicateRes.Count() == 0)
    {
```

This code identifies the event selected when a user is creating a reservation. This code checks to see if the selected truck already a reservation on the date of the reservation has being created. If a reservation already exists, the user is shown a warning stating that the truck selected already has a reservation for that day. This applies to users in all roles. Not even a Super User can schedule a truck in two places at once.

Implementing the reservation limit based on location only applies to the owners of the trucks. This means an Admin or Super User is able to schedule a truck for an event even if the limit for that location has already been met. To ensure that the owners of trucks are not able to do the same thing, the following code was used.



```
if (User.IsInRole("Owner"))
{

    int resLoc = ev.LocationID;
    //Find the reservation limit for the location
    int limit = db.Locations.Where(l => l.LocationID == resLoc).Select(l =>
      l.ReservationLimit).Single();
    //Get the number of reservations at the location for the same date
    int resNum = db.Reservations.Where(l => l.Event.LocationID == resLoc && ev.EventDate ==
      l.ReservationDate).Count();
    //See if the number of reservations is less than the limit
    //If < limit then add changes and redirect to index
    if (resNum < limit)
    {
        //get the reservation ddate from the event
        reservation.ReservationDate = ev.EventDate;

        db.Reservations.Add(reservation);
        db.SaveChanges();
        return RedirectToAction("Index");
    }
    else
    {
        // Otherwise, error message that reservation limit for day & location has been reached
        ViewBag.ErrorMessageLimit = $"* {ev.EventName} has reached it's reservation limit for
          {ev.EventDate:d}.";
```

Staying Organized

This project included the design and creation of a database, and a two-tier ASP.Net MVC project which had to be as usable on mobile as on a desktop/laptop. I used a Trello board to help me keep up with all of the steps I needed to follow to get it completed on time. I also used the board to track errors and improvements that I want to implement on the next round of development. Using Trello helped me to document small errors that I could go back and fix after finishing the functionality in progress without overlooking them later on.

I developed this project with a mind toward cross-browser compatibility. There is still a bug in the Reservations table in Chrome that relates to the use of Datatables.net not allowing the links and sorting functions to work on the right side of the table when the SysAdmin or the Admin are logged in. This is first on my list to fix in the second round of development.

S. Tessa Toney

210-918-0491 ☎

s.l.toney@outlook.com ✉

www.linkedin.com/in/s-tessa-toney in

FULL-STACK DEVELOPER

www.tessatoney.com

SUMMARY OF QUALIFICATIONS

- Solid foundational knowledge of designing and developing full-stack web applications using .NET framework.
- Proven history of leadership and teamwork.
- Clear and effective communication that encourages open dialog for improvements and clarifications.
- Ability and willingness to learn new skills and knowledge sets that will allow further progression.

TECHNICAL SKILLS

Front End: HTML5, JavaScript, jQuery, jQueryUI, CSS3, Responsive/Mobile Web Development, Bootstrap, ReactJS

Middle Tier: Visual Studio, C#.NET, ASP.NET, LINQ, MVC, EF

Back End: ADO.NET, SQL, SQL Server, SSMSE jQuery

INDEPENDENT DEVELOPMENT PROJECTS

- **Personal Site**: www.tessatoney.com
- **StoreFront:** Created a secure application for managing product data. Application is built to simulate an online store front with a shopping cart. Administrators can manage product, category and vendor data.
- **S.A.T. Scheduling Administration Tool:** Created a secure application for managing product data. Application is built to simulate an online class scheduling system. Administrators will have the ability to manage students, courses, scheduled classes, and enrollments.
- **Final Project**: Created a secure data-driven ASP.NET MVC application from design through deployment for managing the tracking and organization of hardware and software within a company. Administrators can manage employee, department data and all details relating to assigned hardware and software.

TECHNICAL TRAINING AND EDUCATION

Centriq Training, Kansas City, MO October 2020 – February 2021

Full-Stack Web Developer Program

Core Competencies:

- MVC Framework
- Trouble Shooting & Debugging
- Source Control
- Agile/Scrum (Created Team Project)
- Website Deployment
- Pair Programming
- Code Review
- Professionalism, Teamwork, Problem Solving & Effective Communication.

University of the Incarnate Word, San Antonio, TX 2013 – 2016
Bachelor of Science: Health Science / Administration

San Antonio College*,* San Antonio, TX 2010 – 2013
Associates of Arts: English

PROFESSIONAL EXPERIENCE

GURU Solutions LLC, Topeka, KS April 2018 – Current
Office Manager & Part-Time Front-End Developer

- Develop website designs with mobile response concept including internal and public functions.
- Assist in building reusable code libraries.
- Collect and define requirements then translate them into functional design documents.
- CRUD testing and documentation.
- Develop user-facing features to create available upgrades.
- Consult with customers and other developers to design programs that fit all end user requirements.
- Continuous collaboration with other developers to improve user experience.
- Track milestones and communicate progress to clients.

U.S. Census Bureau, Overland Park, KS June 2020 – October 2020
Census Field Supervisor (CFS)

- Worked on the IT onboarding and trained enumerators to use their issued devices.
- Tracked field performance of enumerators and provided field training when needed.
- Provided team members with guidance and appropriate solutions to field problems as they occurred.
- Consulted leadership on complex issues that arose in the field to ensure proper guidance was given.
- Reviewed and approved team time and expenses daily.
- Confirmed procedural compliance and quick progress while maintaining standards.

Appen, Remote November 2017 – May 2018
Web Content & Search Evaluator

- Web search evaluation using Excel spreadsheets or an annotation tool.
- Reviewed and rated web content using a web application to support measurement of search data relevance.
- Evaluations were based on detailed criteria.

UPS, San Antonio, TX August 2014 – March 2017
DWS Testing & Control Admin.

- Daily tolerance testing of scanners; produced and maintained records of testing results.
- Identified and fixed equipment problems that could be fixed – otherwise reported to and coordinated with TSG to reach a resolution.
- Created and maintained data records including visual representations of current situations.
- Coordinated with supervisors and managers in multiple locations to maintain standards across the district.
- Served as a reliable backup to cover the duties of other employees in the district including my supervisor.
- Tracked performance and revenue brought in by the DWS machines.
- Coordinated job duties in the absence of upper management.
- Worked with computer-based systems (Microsoft Office, SharePoint, etc.) to maintain records and keep everyone informed in real time.