Spencer Pearson

Rocket Ship Coffee Job Board

Final Project Presentation

February 4th, 2021

Rocket Fuel Coffee

About My Job Board Project

I chose the Job Board project because it was the perfect vehicle to highlight my strengths in design and branding. I had already created a fictitious coffee shop in my previous StoreFront project, so I decided the logical next step would be to create a fully functional Job Board application this company could use.

The Job Board also gave me the ability to flex my logic and coding skills, as the application needs to provide a different experience depending on which user is interacting with it at any given time. I built in custom logic that filters data for managers at their unique location, I change the Call To Action and FAQ sections of the home page based on the user's role and I built logic to automatically email an applicant when the status of their application.

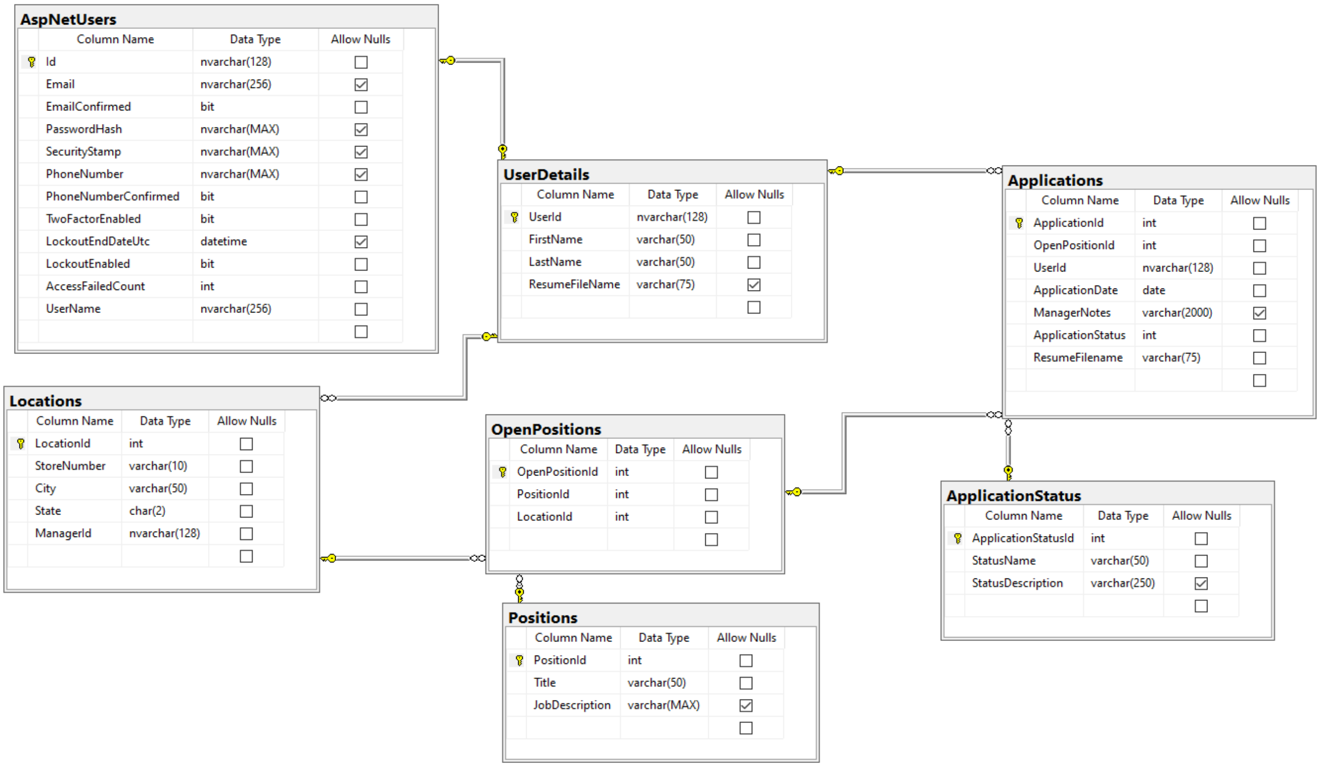
> ***"This project is the perfect vehicle for me to showcase both my logical coding abilities and my visual design skills"***

Web design has always been of great interest to me, but I have felt limited in the past by WordPress Themes and WISIWYG web editors. I came to Centriq because I wanted to be able to get into the guts of a web application and not only fine-tune the way a page looked, but also update the core functionality in different ways. My classwork at Centriq represented some of the first programming exercises I'd ever encountered.

I've never learned so much in such a short amount of time. With the knowledge and experience I gained at Centriq, I can now do far more than just make a pretty page of graphics, images and text. I now know how to design and build an application and its associated database from the ground up, and can build an application that uses relational tables and logic to make it possible for anyone to interact with the application simply and securely.

Job Board Project Requirements:

- Create an application that allows a company to share job openings between their multiple locations with the general public
- Create a user role system within the application that allows:
  - Managers to post job openings at their locations and accept or reject applications that are submitted for those openings
  - Applicants to upload a resume and apply for any open position with an easy one-click apply feature
  - Anonymous users to read basic information and create an applicant account
  - An administrator who has full CRUD functionality for locations, positions, users and every other feature of the app
- The target user audience is very mobile-forward in their use of web applications. As such, the app should look nice and function smoothly on mobile devices.
- Specific requirements:
  - One-click apply functionality for applicants
  - Auto-generate the Application date for all applications
  - Provide an easy-to-read list of job openings to applicants
  - Give managers a list of applications filtered for their location



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)	☐
		☐

UserDetails

Column Name	Data Type	Allow Nulls
UserId	nvarchar(128)	☐
FirstName	varchar(50)	☐
LastName	varchar(50)	☐
ResumeFileName	varchar(75)	☑
		☐

Applications

Column Name	Data Type	Allow Nulls
ApplicationId	int	☐
OpenPositionId	int	☐
UserId	nvarchar(128)	☐
ApplicationDate	date	☐
ManagerNotes	varchar(2000)	☑
ApplicationStatus	int	☐
ResumeFilename	varchar(75)	☐
		☐

Locations

Column Name	Data Type	Allow Nulls
LocationId	int	☐
StoreNumber	varchar(10)	☐
City	varchar(50)	☐
State	char(2)	☐
ManagerId	nvarchar(128)	☐
		☐

OpenPositions

Column Name	Data Type	Allow Nulls
OpenPositionId	int	☐
PositionId	int	☐
LocationId	int	☐
		☐

ApplicationStatus

Column Name	Data Type	Allow Nulls
ApplicationStatusId	int	☐
StatusName	varchar(50)	☐
StatusDescription	varchar(250)	☑
		☐

Positions

Column Name	Data Type	Allow Nulls
PositionId	int	☐
Title	varchar(50)	☐
JobDescription	varchar(MAX)	☑
		☐

A snapshot of the schema of my database for the Job Board application. I included a 1:1 relationship in UserDetails that ties to the AspNetUsers table for identity.

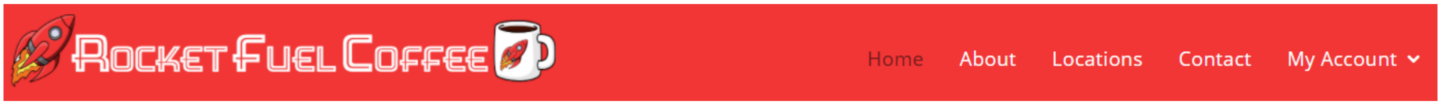
User Roles (Can/Can't Do):

Role:	UserDetails	Locations	OpenPositions	Positions	Applications	ApplicationStatus
Anon	Hidden	Can view	Hidden	Hidden	Hidden	Hidden
Applicant	Can view and edit their own	Can view	Can view, sort by location	Hidden	Can view their apps	Controller and views hidden
Manager	Can view and edit their own	Can view and edit	Full CRUD functionality	Can create and edit	Can view*, edit, and delete*	Hidden
Admin	Full CRUD functionality	Full CRUD functionality	Full CRUD functionality	Full CRUD functionality	Full CRUD functionality	Full CRUD functionality

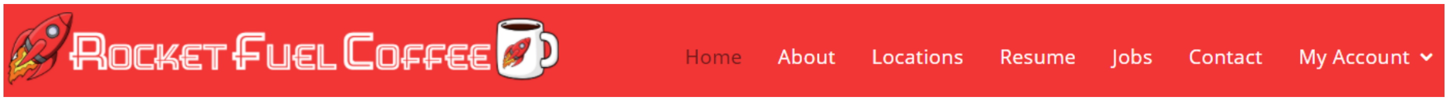
*Managers can view all applications submitted for their location and can delete applications 6 months after the application date.

Navigation based on user roles:

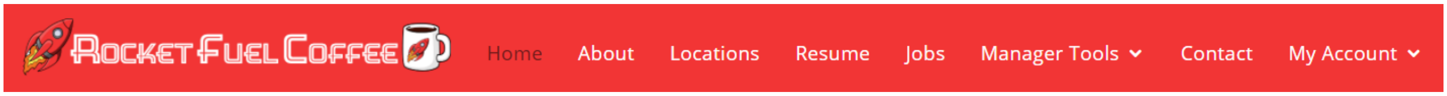
Anonymous user:



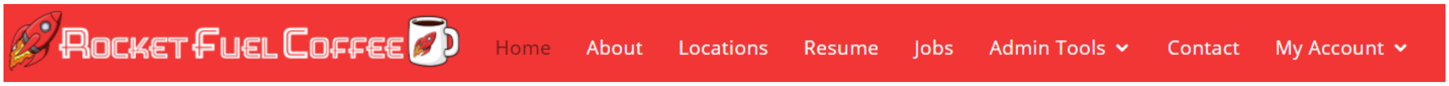
Applicant:



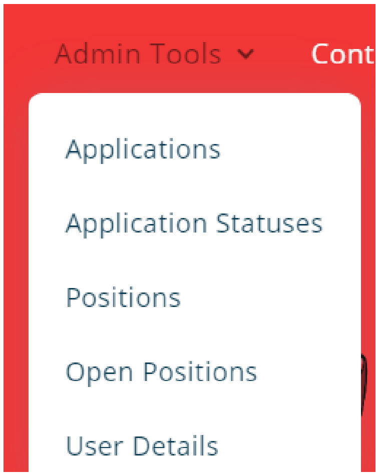
Manager:



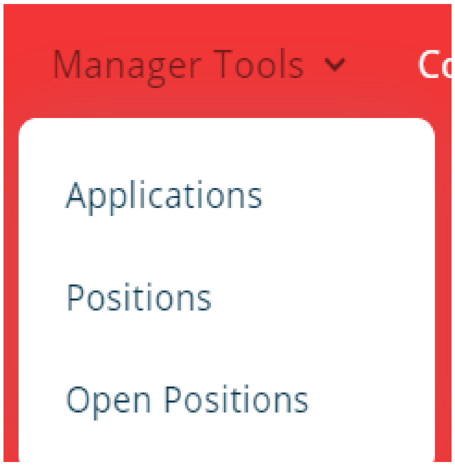
Admin:



The navigation automatically updates based upon the role the user is in. This gives a clean, consistent feel across all roles, while giving access to roles that need extra views and hiding views from unauthorized users. Additional functionality for Managers and Admin are stored in a separate dropdown to avoid navbar clutter:



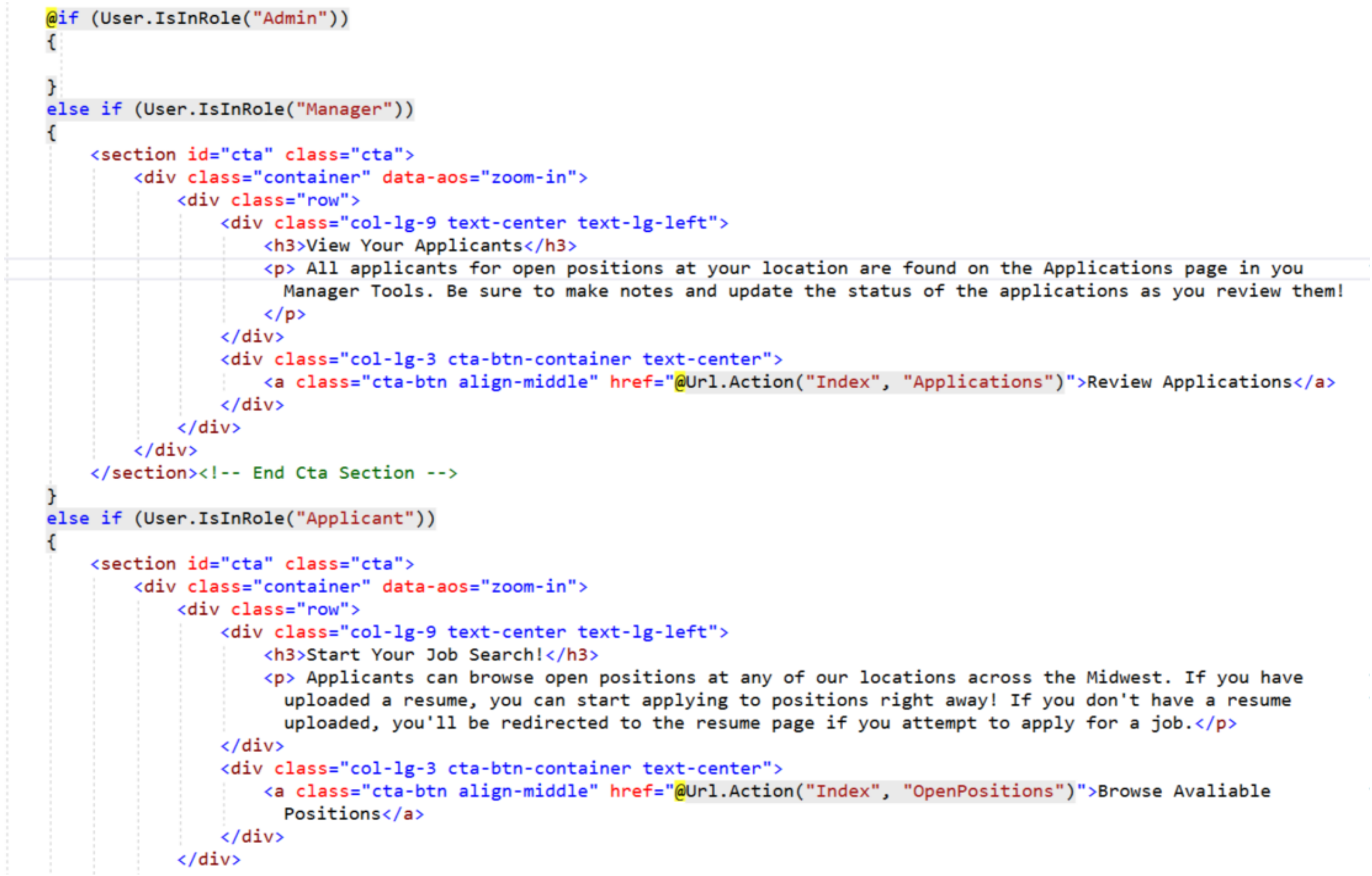
Dropdown menu for Admin Tools



Dropdown menu for Manager Tools

Unique code and logic:

There are several pieces of functionality that are unique to my Job Board project. One of the simplest to implement yet most elegantly rendered pieces is the Call To Action and the FAQ section on my Index view. The action and questions will change based on the role of the current user. Here is a section of the Call To Action code for brevity:



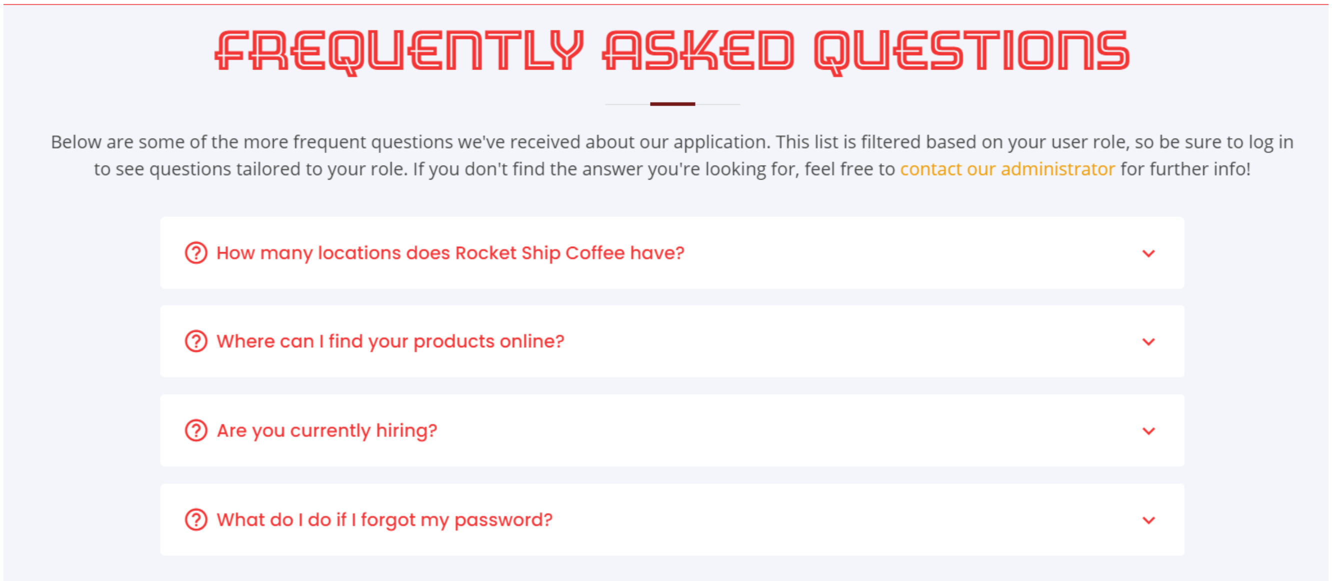
```
@if (User.IsInRole("Admin"))
{

}
else if (User.IsInRole("Manager"))
{
    <section id="cta" class="cta">
        <div class="container" data-aos="zoom-in">
            <div class="row">
                <div class="col-lg-9 text-center text-lg-left">
                    <h3>View Your Applicants</h3>
                    <p> All applicants for open positions at your location are found on the Applications page in you
                      Manager Tools. Be sure to make notes and update the status of the applications as you review them!
                    </p>
                </div>
                <div class="col-lg-3 cta-btn-container text-center">
                    <a class="cta-btn align-middle" href="@Url.Action("Index", "Applications")">Review Applications</a>
                </div>
            </div>
        </div>
    </section><!-- End Cta Section -->
}
else if (User.IsInRole("Applicant"))
{
    <section id="cta" class="cta">
        <div class="container" data-aos="zoom-in">
            <div class="row">
                <div class="col-lg-9 text-center text-lg-left">
                    <h3>Start Your Job Search!</h3>
                    <p> Applicants can browse open positions at any of our locations across the Midwest. If you have
                      uploaded a resume, you can start applying to positions right away! If you don't have a resume
                      uploaded, you'll be redirected to the resume page if you attempt to apply for a job.</p>
                </div>
                <div class="col-lg-3 cta-btn-container text-center">
                    <a class="cta-btn align-middle" href="@Url.Action("Index", "OpenPositions")">Browse Avaliable
                      Positions</a>
                </div>
            </div>
```

CTA Section: An Admin is not shown any call to action. Managers are prompted to review applications while Applicants are prompted to browse jobs. Not pictured: Anonymous users are prompted to the contact page.

This functionality gives the user some additional direction on how to use the app based on the most common action for their role. The FAQ Section also renders different questions based upon the role, as shown on the next page.

All users are shown a basic set of questions:

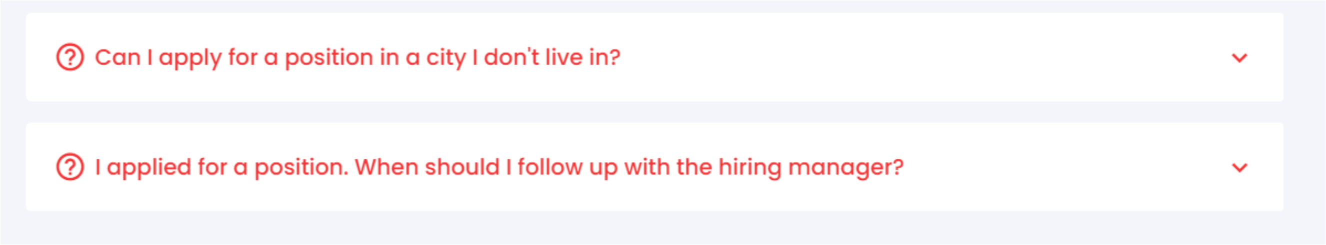


FREQUENTLY ASKED QUESTIONS

Below are some of the more frequent questions we've received about our application. This list is filtered based on your user role, so be sure to log in to see questions tailored to your role. If you don't find the answer you're looking for, feel free to contact our administrator for further info!

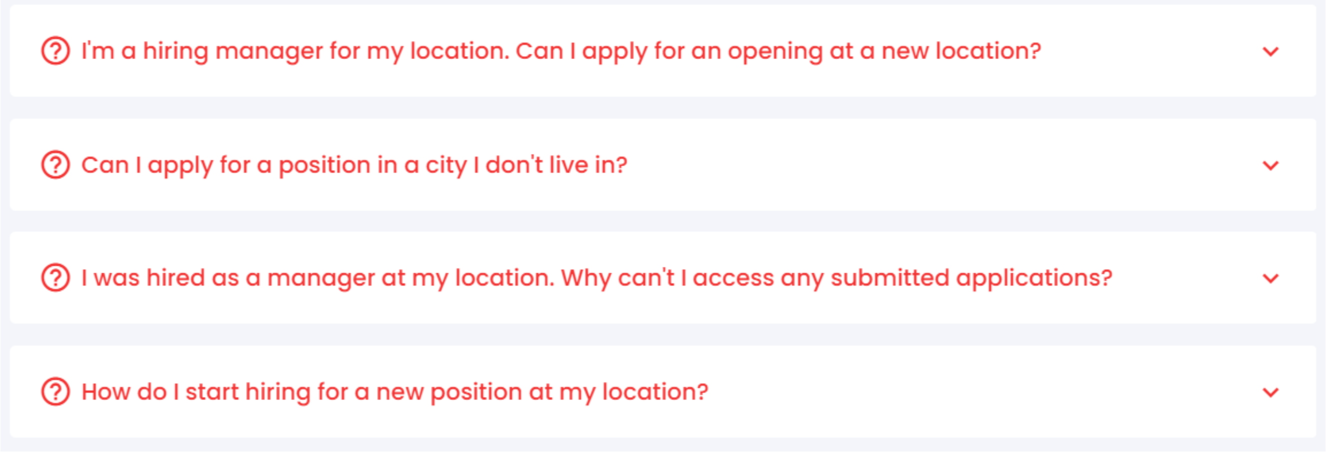
- How many locations does Rocket Ship Coffee have?
- Where can I find your products online?
- Are you currently hiring?
- What do I do if I forgot my password?

Applicants will see an additional two questions:



- Can I apply for a position in a city I don't live in?
- I applied for a position. When should I follow up with the hiring manager?

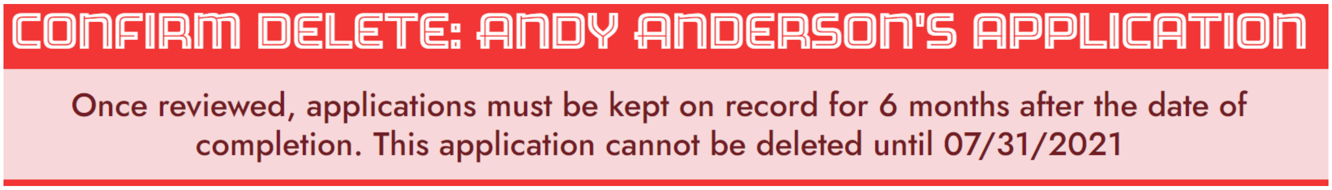
And managers will see additional questions unique to their role as well:



- I'm a hiring manager for my location. Can I apply for an opening at a new location?
- Can I apply for a position in a city I don't live in?
- I was hired as a manager at my location. Why can't I access any submitted applications?
- How do I start hiring for a new position at my location?

Admins are not shown any additional FAQs as they are expected to have a more substantial depth of knowledge about how the application works.

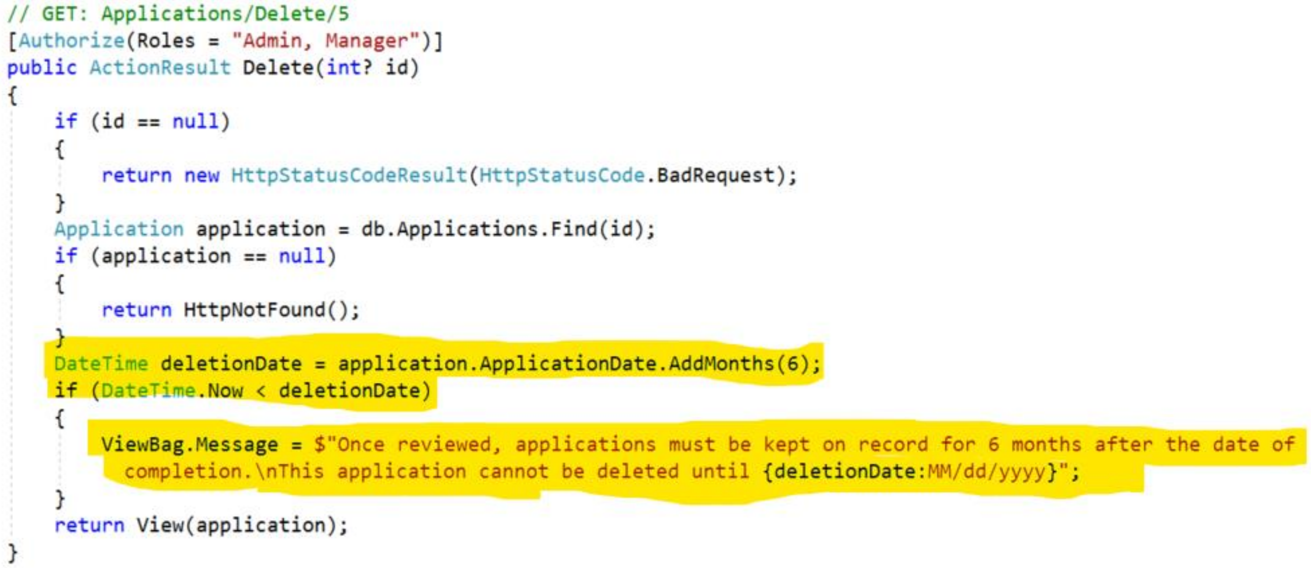
Many businesses are required to keep applications on file for a minimum period of time. Thus, I added functionality that will warn Managers and Admins if they enter the 'delete' view of Applications within 6 months of the application date:



CONFIRM DELETE: ANDY ANDERSON'S APPLICATION

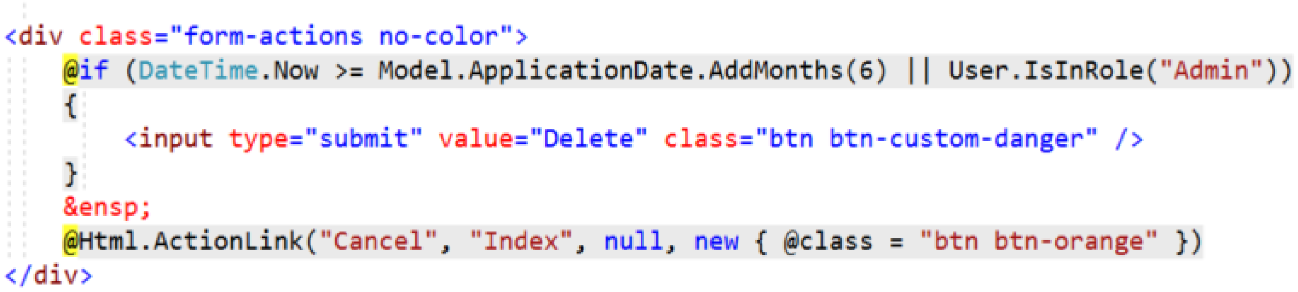
Once reviewed, applications must be kept on record for 6 months after the date of completion. This application cannot be deleted until 07/31/2021

Admins can ignore the message and delete the application, but the manager's 'delete' button will be hidden from the view until the date displayed in the warning.



```
// GET: Applications/Delete/5
[Authorize(Roles = "Admin, Manager")]
public ActionResult Delete(int? id)
{
    if (id == null)
    {
        return new HttpStatusCodeResult(HttpStatusCode.BadRequest);
    }
    Application application = db.Applications.Find(id);
    if (application == null)
    {
        return HttpNotFound();
    }
    DateTime deletionDate = application.ApplicationDate.AddMonths(6);
    if (DateTime.Now < deletionDate)
    {
        ViewBag.Message = $"Once reviewed, applications must be kept on record for 6 months after the date of
          completion.\nThis application cannot be deleted until {deletionDate:MM/dd/yyyy}";
    }
    return View(application);
}
```

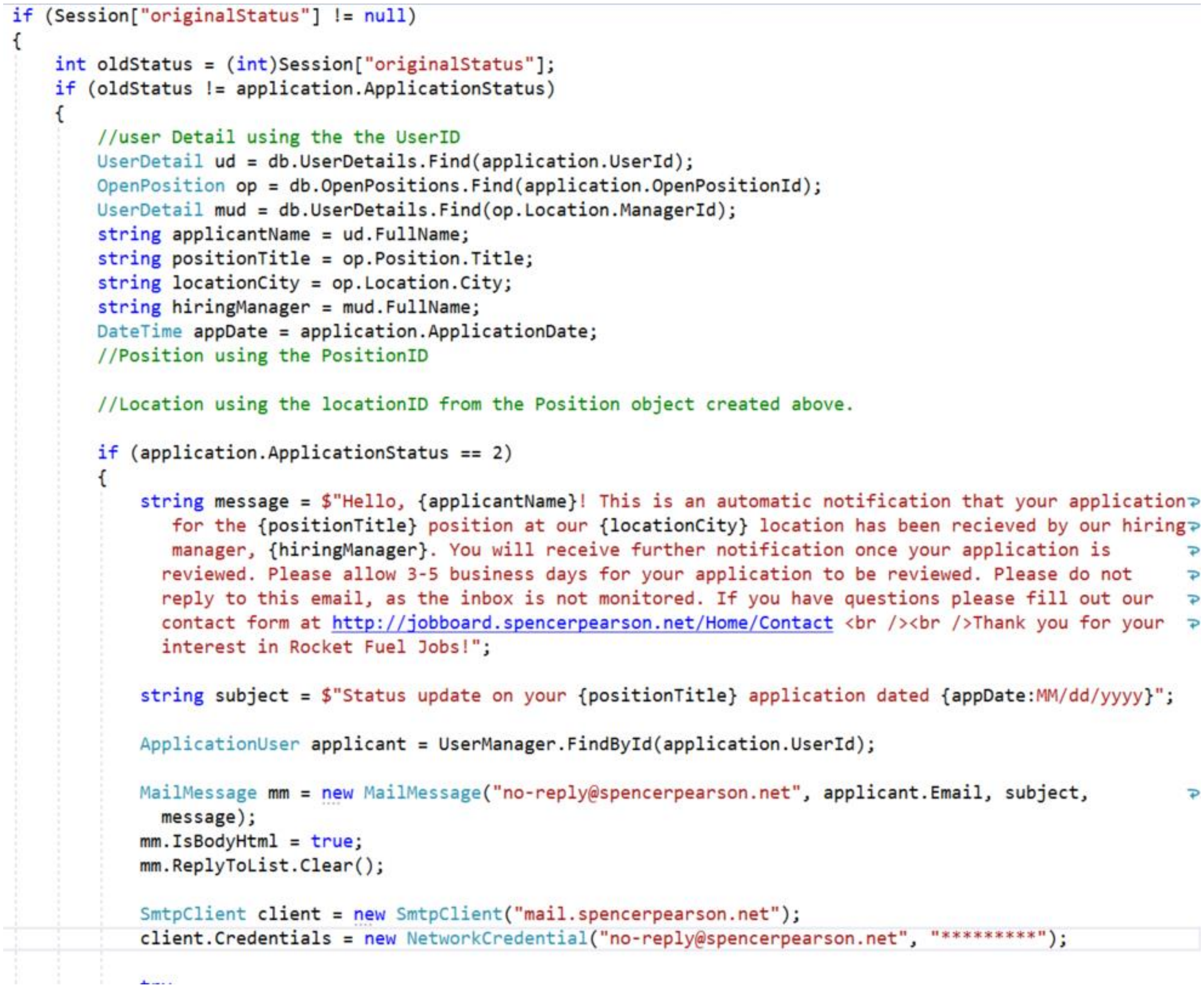
This is the controller logic that checks the application date against the current date and sends a warning message if it is less than 6 months from the application date.



```
<div class="form-actions no-color">
    @if (DateTime.Now >= Model.ApplicationDate.AddMonths(6) || User.IsInRole("Admin"))
    {
        <input type="submit" value="Delete" class="btn btn-custom-danger" />
    }
    &ensp;
    @Html.ActionLink("Cancel", "Index", null, new { @class = "btn btn-orange" })
</div>
```

This is the view logic that hides the delete button from managers for 6 months after the application date.

Finally, I want to highlight the custom automatic email notification functionality I added in the Applications controller. This section of code will automatically generate an email and send it to the applicant when a Manager (or Admin) edits the status of any of their applications:



```
if (Session["originalStatus"] != null)
{
    int oldStatus = (int)Session["originalStatus"];
    if (oldStatus != application.ApplicationStatus)
    {
        //user Detail using the the UserID
        UserDetail ud = db.UserDetails.Find(application.UserId);
        OpenPosition op = db.OpenPositions.Find(application.OpenPositionId);
        UserDetail mud = db.UserDetails.Find(op.Location.ManagerId);
        string applicantName = ud.FullName;
        string positionTitle = op.Position.Title;
        string locationCity = op.Location.City;
        string hiringManager = mud.FullName;
        DateTime appDate = application.ApplicationDate;
        //Position using the PositionID

        //Location using the locationID from the Position object created above.

        if (application.ApplicationStatus == 2)
        {
            string message = $"Hello, {applicantName}! This is an automatic notification that your application
               for the {positionTitle} position at our {locationCity} location has been recieved by our hiring
               manager, {hiringManager}. You will receive further notification once your application is
              reviewed. Please allow 3-5 business days for your application to be reviewed. Please do not
              reply to this email, as the inbox is not monitored. If you have questions please fill out our
              contact form at http://jobboard.spencerpearson.net/Home/Contact <br /><br />Thank you for your
              interest in Rocket Fuel Jobs!";

            string subject = $"Status update on your {positionTitle} application dated {appDate:MM/dd/yyyy}";

            ApplicationUser applicant = UserManager.FindById(application.UserId);

            MailMessage mm = new MailMessage("no-reply@spencerpearson.net", applicant.Email, subject,
              message);
            mm.IsBodyHtml = true;
            mm.ReplyToList.Clear();

            SmtpClient client = new SmtpClient("mail.spencerpearson.net");
            client.Credentials = new NetworkCredential("no-reply@spencerpearson.net", "*********");
```

This code uses a session variable to store the "old" status, then checks to see if the status has been updated. If it has been, a unique message will be generated based upon which status the application has been updated to. For brevity, the logic for the "received" status has been shown above.

This application was a lot of fun to create, and I am proud of the way it turned out. I challenged myself to include some unique functionality and I was happy with my ability to meet the requirements specified in the short time frame given to complete the project.

-Spencer Pearson
Full Stack Web Developer
http://spencerpearson.net

[Spencer Pearson]

<FULL-STACK WEB DEVELOPER />

T: 816.729.6963
E: spencer@spencerpearson.net

- linkedin.com/in/spencer-pearson
- github.com/SpencerPearson
- http://spencerpearson.net

QUALIFICATIONS

- Solid foundational knowledge of designing and developing full-stack web applications using .NET framework.
- Years of experience in design, deployment, and maintenance of Front-End applications as an HTML and WordPress hobbyist.
- Extensive background in digital multimedia design, editing, and production.
- Various management experiences in Beverage, Hospitality and Retail industries including hiring and training employees.
- Adaptive communicator with an instinct for matching diverse communication styles across any team.
- Entrepreneur with experience starting and managing a business.
- Experience developing and maintaining client and vendor relationships.

TECHNICAL SKILLS

Front End	• HTML5 • JavaScript • jQuery • jQueryUI • CSS3	• Mobile Web Development • Bootstrap • ReactJS • WordPress
Middle Tier	• Visual Studio • ASP.NET • C# .NET	• LINQ • MVC Framework • Entity Framework
Back End	• ADO.NET • SQL Server	• SQL • SSMSE

INDEPENDENT DEVELOPMENT PROJECTS

- **Personal Site:** spencerpearson.net
- **StoreFront:** A secure product data management application that an ecommerce store with a shopping cart. Administrators can change product, category & vendor data.
- **S.A.T. Scheduling Tool:** This product data management application simulates an online class scheduling system. Admin users can manage students, courses, scheduled classes, and enrollments.
- **Final project:** Designed and deployed a secure data-driven ASP.NET MVC application that tracks and organizes hardware and software within a company. Administrators can manage employee, department data, and all details related to assigned hardware and software.

TECHNICAL TRAINING & EDUCATION

2020 - 2021
Full-Stack Developer Program
CENTRIQ TRAINING
Kansas City, MO

- MVC Framework
- Troubleshooting & debugging
- Agile/Scrum (Team Project)
- Website Deployment
- Pair programming
- Code review
- Professionalism & Teamwork
- Problem solving
- Effective communication

2008 - 2011
Journalism, Psychology
UNIVERSITY OF MISSOURI
Columbia, MO

WORK EXPERIENCE

2019 - Present
LUPINE MEDIA
Founder & Owner

- Provide beautiful fully-customized media production for professional clients as well as personal projects.
- Photography, audio/video production and digital design are the specialties.

2016 - 2019
THE ROASTERIE
Tour Guide, Retail Sales Manager

- Started as a tour guide and worked up to Retail Sales Manager.
- Responsible for hiring and training tour guides, tracking sales data, and managing inventory.

2011 - 2016
THE BLUE FUGUE | GUMBY’S | JAZZ
Bartender, General Manager

- Worked from bartender to management at three local bars in Columbia.
- Learned the importance of quality, communication, and problem solving.