Jonathan Baker

Full Stack Web Developer

jdbaker.net

jon.david.baker@gmail.com

Agency Employee Training and Education

A Learning Management System

May 3rd, 2019

About Me

For as long as I can remember, I've always had an interest in technology and computers. When my family's computer was no longer able to run the games that I wanted to play, I saved up my money and built myself a gaming PC when I was in high school. It was also in high school that I was first exposed to coding. I was fortunate enough to be able to take classes in BASIC, Pascal, and C++ as electives in high school. I was even a member of the school's competitive coding team for a time.

I planned to pursue development as a career over 15 years ago, entering college as a Computer Science major. However, traditional college was not a good fit for me at the time and I never completed a degree. I began working as Direct Support Staff for individuals with developmental disabilities over 10 years ago. I loved that job and found it very rewarding. It required a lot of creative problem solving and problem sensitivity. However, after over a decade, it was time for me to move on.

I enrolled in Centriq Training's full-stack web development program and I loved it immediately. The fast pace and focused material kept me interested and it was very challenging without being overwhelming. I really enjoyed learning about all of the different technologies and how to use them together with ASP.NET MVC. My favorite part of the track has been the middle-tier and back-end technologies. Working with C# to solve a problem often feels like a fun logic puzzle to me.

Because of that, I chose the Learning Management System for my final project. It seemed natural to use my former career in Direct Support as inspiration for the theming of the project. I also felt that the LMS would offer me the most opportunities to do some creative work with C#. This is what I focused on the most during development, utilizing custom ViewModels, ExpandoObjects, and plenty of LINQ method syntax.

Project Brief:

Agency is an online Learning Management System aimed at agencies providing support for individuals with developmental disabilities. Staff working in this field require many trainings which need to be renewed regularly. Not only does this application allow staff to work through their trainings and monitor their own progress, but managers are also able to monitor their staff to make sure that everyone working is fully qualified.

Each course has at least one lesson. The lessons may contain a video and/or a PDF file with additional information. Once an employee has viewed every lesson in a course, it is recorded as a completed course. The employee's manager will be sent an email automatically each time one of their employees completes a course.

Roles and Use-Cases

Anonymous Users:

-Can view courses and lessons, but no progress is recorded

Employees:

-Can view courses and lessons, with LessonView and CourseCompletion tables updating automatically

-Can view “My Progress” view, detailing which courses are complete and incomplete. Incomplete courses display the number of remaining lessons and percent complete. Completed courses display the expiration date of the training, highlighting it in red if it will expire soon.

Managers:

-Can view all courses and lessons

-Can view “My Staff” view, displaying a list of all of the staff members they supervise, along with the number of incomplete courses and the number of courses that are expired or will expire soon.

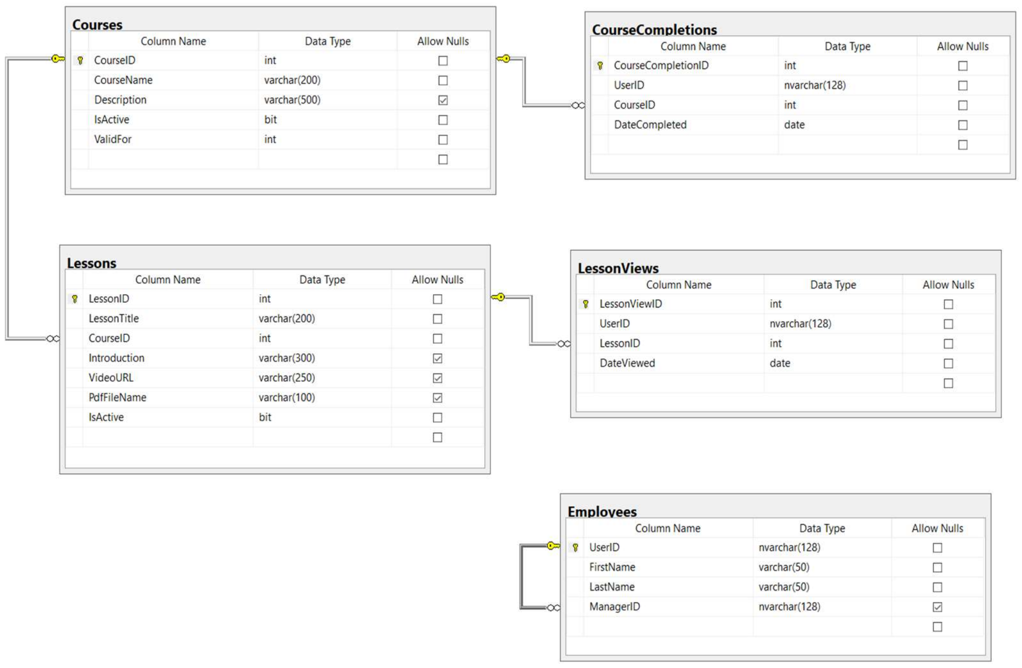
-Can view each staff member’s individual progress, similar to the Employee’s “My Progress” view, but with some additional contact information for the staff member.

Admin:

-Full Create, Read, Update, and Delete capabilities for employees, lessons and courses.

-Can view and edit courses before they are active, ensuring that a course is not published before it is complete.

Database Diagram



Courses

Column Name	Data Type	Allow Nulls
CourseID	int	☐
CourseName	varchar(200)	☐
Description	varchar(500)	☑
IsActive	bit	☐
ValidFor	int	☐
		☐

CourseCompletions

Column Name	Data Type	Allow Nulls
CourseCompletionID	int	☐
UserID	nvarchar(128)	☐
CourseID	int	☐
DateCompleted	date	☐
		☐

Lessons

Column Name	Data Type	Allow Nulls
LessonID	int	☐
LessonTitle	varchar(200)	☐
CourseID	int	☐
Introduction	varchar(300)	☑
VideoURL	varchar(250)	☑
PdfFileName	varchar(100)	☑
IsActive	bit	☐
		☐

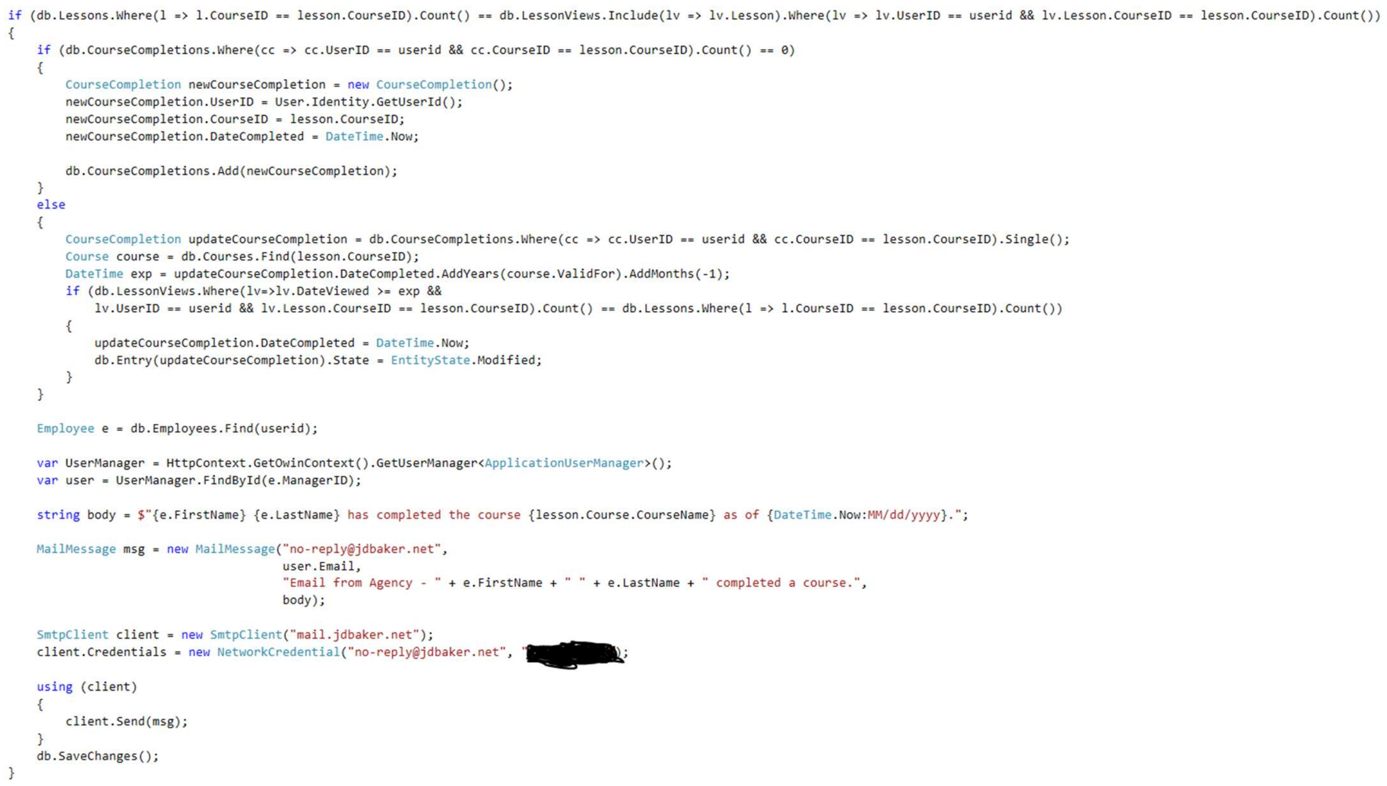
LessonViews

Column Name	Data Type	Allow Nulls
LessonViewID	int	☐
UserID	nvarchar(128)	☐
LessonID	int	☐
DateViewed	date	☐
		☐

Employees

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

Code Snippets and Output Examples

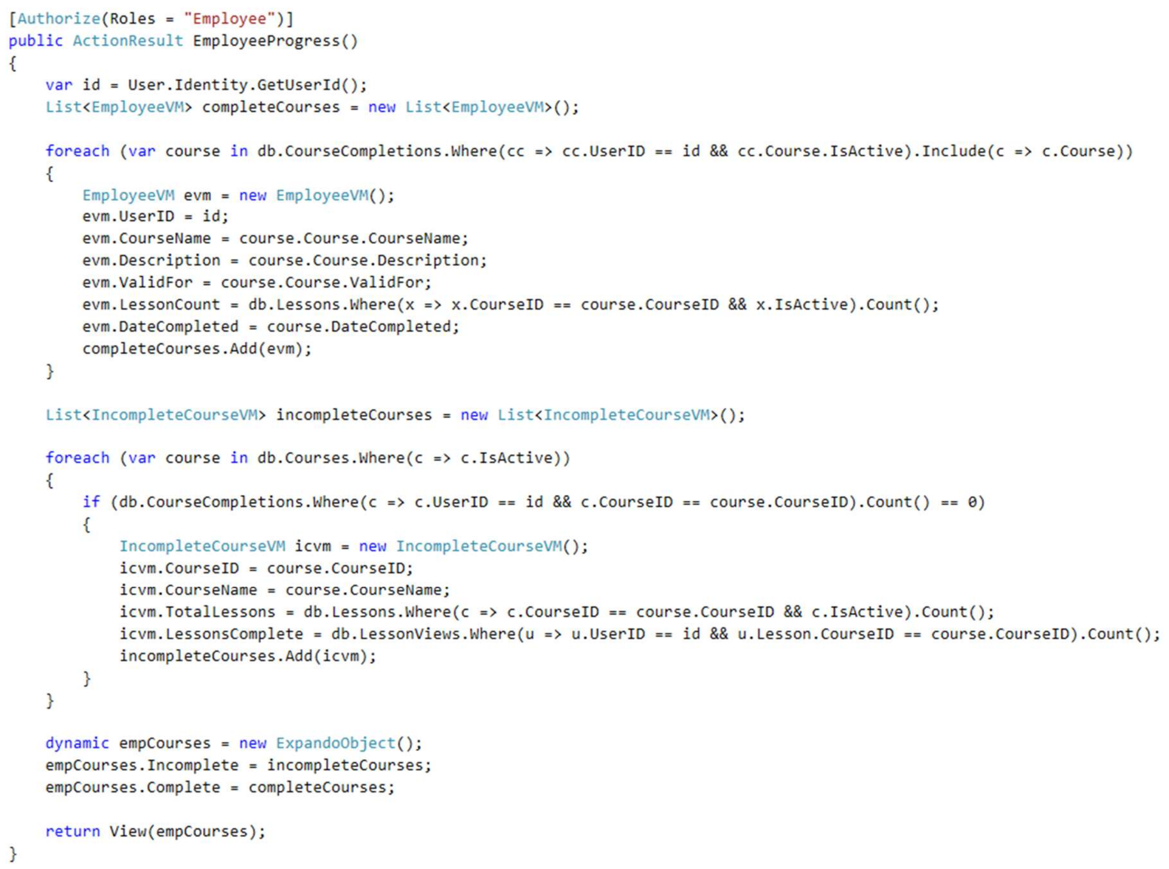


```
if (db.Lessons.Where(l => l.CourseID == lesson.CourseID).Count() == db.LessonViews.Include(lv => lv.Lesson).Where(lv => lv.UserID == userid && lv.Lesson.CourseID == lesson.CourseID).Count())
{
    if (db.CourseCompletions.Where(cc => cc.UserID == userid && cc.CourseID == lesson.CourseID).Count() == 0)
    {
        CourseCompletion newCourseCompletion = new CourseCompletion();
        newCourseCompletion.UserID = User.Identity.GetUserId();
        newCourseCompletion.CourseID = lesson.CourseID;
        newCourseCompletion.DateCompleted = DateTime.Now;

        db.CourseCompletions.Add(newCourseCompletion);
    }
    else
    {
        CourseCompletion updateCourseCompletion = db.CourseCompletions.Where(cc => cc.UserID == userid && cc.CourseID == lesson.CourseID).Single();
        Course course = db.Courses.Find(lesson.CourseID);
        DateTime exp = updateCourseCompletion.DateCompleted.AddYears(course.ValidFor).AddMonths(-1);
        if (db.LessonViews.Where(lv=>lv.DateViewed >= exp &&
            lv.UserID == userid && lv.Lesson.CourseID == lesson.CourseID).Count() == db.Lessons.Where(l => l.CourseID == lesson.CourseID).Count())
        {
            updateCourseCompletion.DateCompleted = DateTime.Now;
            db.Entry(updateCourseCompletion).State = EntityState.Modified;
        }
    }

    Employee e = db.Employees.Find(userid);

    var UserManager = HttpContext.GetOwinContext().GetUserManager<ApplicationUserManager>();
    var user = UserManager.FindById(e.ManagerID);

    string body = $"{e.FirstName} {e.LastName} has completed the course {lesson.Course.CourseName} as of {DateTime.Now:MM/dd/yyyy}.";

    MailMessage msg = new MailMessage("no-reply@jdbaker.net",
                                      user.Email,
                                      "Email from Agency - " + e.FirstName + " " + e.LastName + " completed a course.",
                                      body);

    SmtpClient client = new SmtpClient("mail.jdbaker.net");
    client.Credentials = new NetworkCredential("no-reply@jdbaker.net", [redacted]);

    using (client)
    {
        client.Send(msg);
    }
    db.SaveChanges();
}
```

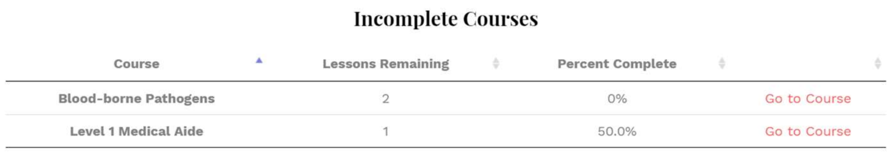
The code snippet above is from the Lessons controller, specifically the Details action. This is where CourseCompletion records are created. If the number of lessons in a course equals the number of lessons the user has viewed from that course, the course has been completed. If there is no record of that user completing that course, a new CourseCompletion record is made. If the user had already taken the course, the existing record is updated with the current date instead.

That strategy presented another problem. If a user is re-taking a course that has more than one lesson, I need to make sure they have viewed each lesson again before the CourseCompletion record is updated. As a workaround for this, I assume that the employee would not be re-taking the course unless they were within one month of the training expiring. The CourseCompletion record is only updated if all of the LessonViews are more recent than one month before the expiration date of the course.

As soon as the course is completed, an email is sent to the user's manager to let them know. This presented an interesting challenge because none of my tables link directly the ASP.NET Identity Samples tables (the Employees table populates with the Identity Samples UserID automatically, but there is no actual relationship.) I had to do some research to figure out how to access the Identity Samples tables to be able to find the email address of the current user's manager since the Identity Samples tables were not part of my database entities context and Email is not a field in my Employees table.

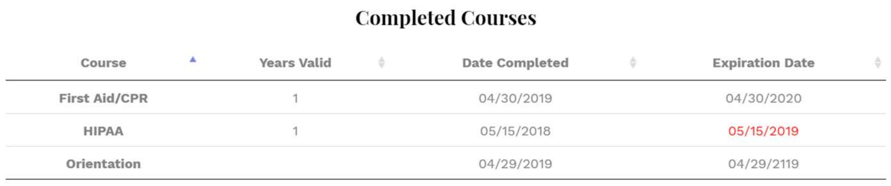


```
[Authorize(Roles = "Employee")]
public ActionResult EmployeeProgress()
{
    var id = User.Identity.GetUserId();
    List<EmployeeVM> completeCourses = new List<EmployeeVM>();

    foreach (var course in db.CourseCompletions.Where(cc => cc.UserID == id && cc.Course.IsActive).Include(c => c.Course))
    {
        EmployeeVM evm = new EmployeeVM();
        evm.UserID = id;
        evm.CourseName = course.Course.CourseName;
        evm.Description = course.Course.Description;
        evm.ValidFor = course.Course.ValidFor;
        evm.LessonCount = db.Lessons.Where(x => x.CourseID == course.CourseID && x.IsActive).Count();
        evm.DateCompleted = course.DateCompleted;
        completeCourses.Add(evm);
    }

    List<IncompleteCourseVM> incompleteCourses = new List<IncompleteCourseVM>();

    foreach (var course in db.Courses.Where(c => c.IsActive))
    {
        if (db.CourseCompletions.Where(c => c.UserID == id && c.CourseID == course.CourseID).Count() == 0)
        {
            IncompleteCourseVM icvm = new IncompleteCourseVM();
            icvm.CourseID = course.CourseID;
            icvm.CourseName = course.CourseName;
            icvm.TotalLessons = db.Lessons.Where(c => c.CourseID == course.CourseID && c.IsActive).Count();
            icvm.LessonsComplete = db.LessonViews.Where(u => u.UserID == id && u.Lesson.CourseID == course.CourseID).Count();
            incompleteCourses.Add(icvm);
        }
    }

    dynamic empCourses = new ExpandoObject();
    empCourses.Incomplete = incompleteCourses;
    empCourses.Complete = completeCourses;

    return View(empCourses);
}
```



Incomplete Courses

Course	Lessons Remaining	Percent Complete	
Blood-borne Pathogens	2	0%	Go to Course
Level 1 Medical Aide	1	50.0%	Go to Course

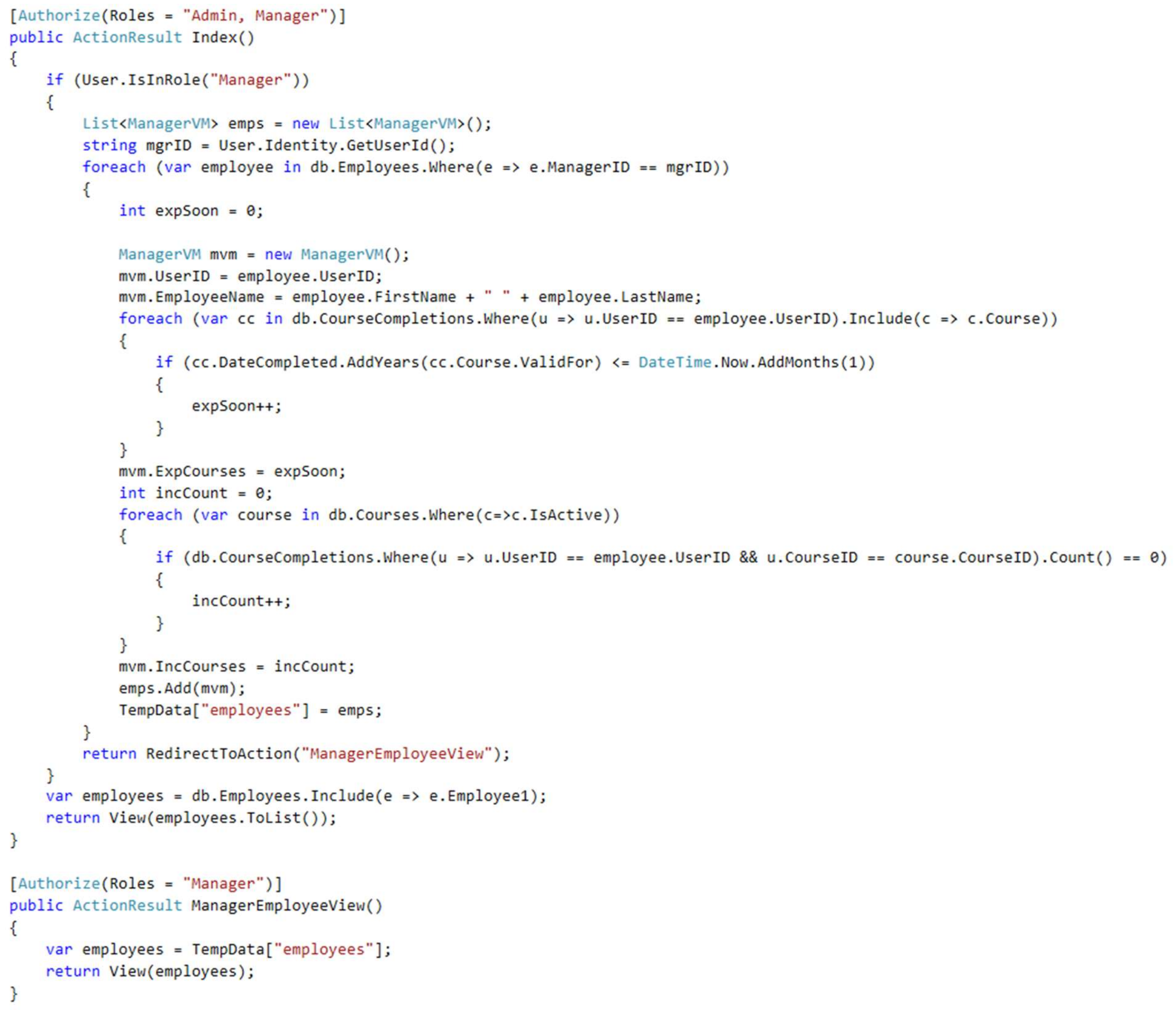


Completed Courses

Course	Years Valid	Date Completed	Expiration Date
First Aid/CPR	1	04/30/2019	04/30/2020
HIPAA	1	05/15/2018	05/15/2019
Orientation		04/29/2019	04/29/2119

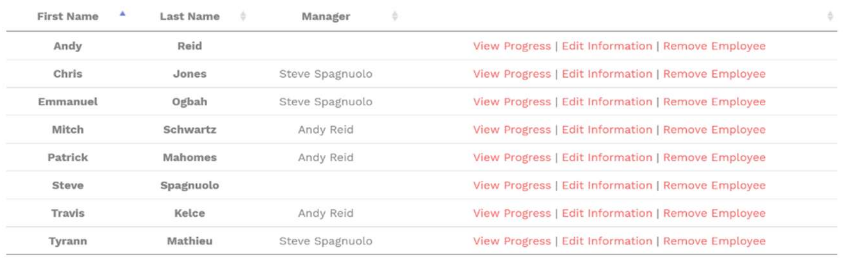
Here you see the code behind the Employee's "My Progress" page and the resulting output. This was another challenge because I wanted to display two different ViewModels on one strongly-typed view. I did some research and I learned about dynamic models and ExpandoObject, which I used here.

Also of note, some columns shown here, like "Expiration Date" and "Percent Complete" are read-only properties that are calculated in the ViewModel.



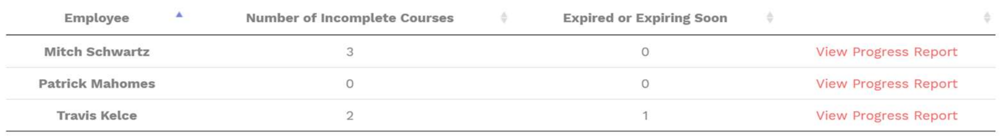
```
[Authorize(Roles = "Admin, Manager")]
public ActionResult Index()
{
    if (User.IsInRole("Manager"))
    {
        List<ManagerVM> emps = new List<ManagerVM>();
        string mgrID = User.Identity.GetUserId();
        foreach (var employee in db.Employees.Where(e => e.ManagerID == mgrID))
        {
            int expSoon = 0;

            ManagerVM mvm = new ManagerVM();
            mvm.UserID = employee.UserID;
            mvm.EmployeeName = employee.FirstName + " " + employee.LastName;
            foreach (var cc in db.CourseCompletions.Where(u => u.UserID == employee.UserID).Include(c => c.Course))
            {
                if (cc.DateCompleted.AddYears(cc.Course.ValidFor) <= DateTime.Now.AddMonths(1))
                {
                    expSoon++;
                }
            }
            mvm.ExpCourses = expSoon;
            int incCount = 0;
            foreach (var course in db.Courses.Where(c=>c.IsActive))
            {
                if (db.CourseCompletions.Where(u => u.UserID == employee.UserID && u.CourseID == course.CourseID).Count() == 0)
                {
                    incCount++;
                }
            }
            mvm.IncCourses = incCount;
            emps.Add(mvm);
            TempData["employees"] = emps;
        }
        return RedirectToAction("ManagerEmployeeView");
    }
    var employees = db.Employees.Include(e => e.Employee1);
    return View(employees.ToList());
}

[Authorize(Roles = "Manager")]
public ActionResult ManagerEmployeeView()
{
    var employees = TempData["employees"];
    return View(employees);
}
```

The snippet above is from the Employees controller. Below, you can see the output when an Admin is signed in. From that view, they can edit or delete Employees. However, that view is not very useful for Managers.



First Name	Last Name	Manager	
Andy	Reid		View Progress | Edit Information | Remove Employee
Chris	Jones	Steve Spagnuolo	View Progress | Edit Information | Remove Employee
Emmanuel	Ogbah	Steve Spagnuolo	View Progress | Edit Information | Remove Employee
Mitch	Schwartz	Andy Reid	View Progress | Edit Information | Remove Employee
Patrick	Mahomes	Andy Reid	View Progress | Edit Information | Remove Employee
Steve	Spagnuolo		View Progress | Edit Information | Remove Employee
Travis	Kelce	Andy Reid	View Progress | Edit Information | Remove Employee
Tyrann	Mathieu	Steve Spagnuolo	View Progress | Edit Information | Remove Employee

Below is the Manager's view of the Employees Index. Not only does it only display the employees who are under that Manager, but it displays information from a different ViewModel to better fit the needs of a Manager.



Employee	Number of Incomplete Courses	Expired or Expiring Soon	
Mitch Schwartz	3	0	View Progress Report
Patrick Mahomes	0	0	View Progress Report
Travis Kelce	2	1	View Progress Report

Jonathan Baker

FULL-STACK DEVELOPER

816-668-8654
jon.david.baker@gmail.com
www.jdbaker.net

Qualifications

- Solid foundational knowledge of designing and developing full-stack web applications using .NET framework.
- Ability to successfully sustain new challenges and stressful situations.
- Function as a team player, as well as demonstrate a proven ability to work well independently.
- Goal-oriented individual with strong leadership capabilities.

Technical Skills

Front End:	HTML5, JavaScript, jQuery, jQueryUI, CSS3, Responsive/Mobile Web Development, Bootstrap
Middle Tier:	Visual Studio, C#.NET, ASP.NET, LINQ, MVC, EF
Back End:	ADO.NET, SQL, SQL Server, SSMSE

Independent Development Projects

- **Personal Site**: www.jdbaker.net
- **U Store**: Created a secure application for managing product data. Application is built to simulate an online store front with a shopping cart. Administrators have the ability to manage product, category and vendor data.
- **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 have the ability to manage employee, department data and all details relating to assigned hardware and software.

Technical Training

CENTRIQ TRAINING | KANSAS CITY, MO **JANUARY 2019 – PRESENT**

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

Professional Experience

COMMUNITY CHOICE, INC. | KANSAS CITY, MO **2018 –2019**

DIRECT SUPPORT PROFESSIONAL

- Assisted individuals with developmental and intellectual disabilities in their homes and communities, helping them live as independently as possible.
- Ensured that medications were properly dosed and administered.
- Transported individuals in adaptive van to facilitate community integration

COMFORT OF HOME HEALTHCARE | KANSAS CITY, MO **2017 – 2018**

LEAD STAFF

- Worked full-time assisting developmentally disabled individuals in their home and the community.
- Managed staff in the home to ensure that individuals were receiving appropriate quality care and assistance.
- Scheduled staff to work shifts and remained on-call to fill shifts as needed.

EASTERSEALS MIDWEST | KANSAS CITY, MO **2014 – 2017**

MEDICAL SUPPORT INSTRUCTOR

- Assisted individuals with developmental disabilities and extensive medical needs.
- Maintained appropriate medical certifications and delegations.

COMMUNITY CHOICE, INC. | KANSAS CITY, MO **2013 – 2014**

COMMUNITY INTEGRATION SPECIALIST

- Assisted individuals with developmental and intellectual disabilities in the community.
- Transported individuals throughout the city to attend events and visit attractions.
- Encouraged appropriate social skills and interactions.

ALTERNATIVE OPPORTUNITIES | KANSAS CITY, MO **2010 – 2013**

LEAD STAFF

- Worked full-time assisting developmentally disabled individuals in their home and in the community.
- Managed staff in the home to ensure that individuals were receiving appropriate quality care and assistance.
- Scheduled staff to work shifts and remained on-call at all times to assist staff or fill shifts as needed.