Here's an example ASP.NET Core MVC application that manages Employee and Admin data with CRUD operations and implements basic authentication:

1. Project Setup:

- Create a new ASP.NET Core MVC project in Visual Studio or your preferred IDE.
- Install required NuGet packages:
  - Microsoft.EntityFrameworkCore
  - Microsoft.EntityFrameworkCore.SqlServer

2. Database Model:

- Define the following classes representing the Employee and Admin tables:

C#
```
public class Employee
{
    public int Id { get; set; }
    public string FirstName { get; set; }
    public string LastName { get; set; }
    public string Position { get; set; }
    public decimal Salary { get; set; }
}

public class Admin
{
    public int AdminId { get; set; }
    public string Username { get; set; }
    public string Password { get; set; }
    public string Email { get; set; }
}
```

3. Data Context:

- Create a DbContext class to interact with the database:

C#
```
public class AppDbContext : DbContext
{
    public AppDbContext(DbContextOptions<AppDbContext> options) :
base(options) { }

    public DbSet<Employee> Employees { get; set; }
    public DbSet<Admin> Admins { get; set; }

    protected override void OnModelCreating(ModelBuilder modelBuilder)
    {
        modelBuilder.Entity<Employee>(entity =>
        {
            entity.HasKey(e => e.Id);
        });
```

```
        modelBuilder.Entity<Admin>(entity =>
        {
            entity.HasKey(e => e.AdminId);
        });
    }
}
```

4. Controllers:

- Create separate controllers for Employee and Admin management:

EmployeeController.cs:

C#

```
public class EmployeeController : Controller
{
    private readonly AppDbContext _context;

    public EmployeeController(AppDbContext context)
    {
        _context = context;
    }

    // GET: Employee/List
    public IActionResult List()
    {
        var employees = _context.Employees.ToList();
        return View(employees);
    }

    // GET: Employee/Create
    public IActionResult Create()
    {
        return View();
    }

    // POST: Employee/Create
    [HttpPost]
    [ValidateAntiForgeryToken]
    public IActionResult Create(Employee employee)
    {
        if (ModelState.IsValid)
        {
            _context.Employees.Add(employee);
            _context.SaveChanges();
            return RedirectToAction("List");
        }
        return View(employee);
    }

    // GET: Employee/Edit/5
    public IActionResult Edit(int id)
    {
        var employee = _context.Employees.Find(id);
        if (employee == null)
```

```
        {
            return NotFound();
        }
        return View(employee);
    }

    // POST: Employee/Edit/5
    [HttpPost]
    [ValidateAntiForgeryToken]
    public IActionResult Edit(int id, Employee employee)
    {
        if (id != employee.Id)
        {
            return NotFound();
        }

        if (ModelState.IsValid)
        {
            _context.Update(employee);
            _context.SaveChanges();
            return RedirectToAction("List");
        }
        return View(employee);
    }

    // GET: Employee/Delete/5
    public IActionResult Delete(int id)
    {
        var employee = _context.Employees.Find(id);
        if (employee == null)
        {
            return NotFound();
        }

        return View(employee);
    }

    // POST: Employee/Delete/5
    [HttpPost, ActionName("Delete")]
    [ValidateAntiForgeryToken]
    public IActionResult DeleteConfirmed(int id)
    {
        var employee = _context.Employees.Find(id);
        _context.Employees.Remove(employee);
        _context.SaveChanges();
        return RedirectToAction("List");
    }
}
```

AdminController.cs:

C#

```
public class AdminController : Controller
{
    private readonly AppDbContext _context;
```

```
public AdminController(AppDbContext context)
{
    _context = context;
}

// GET: Admin/Login
public IActionResult Login()
{
    return View();
}

// POST: Admin/Login
[HttpPost]
[ValidateAntiForgeryToken]
public IActionResult Login(Admin admin)
{
    var foundAdmin
```