*Please note this document is a rough draft intended to be fleshed out when included as a part of an actual website. It is by no means a complete, production-ready document.*

# Introduction

Dot Net Admin (DNA) strives to standardize the application development process. It consists of a parent website, dashboard, and collection of optional class libraries to provide the functionality needed by your application.

Provided as Assembly Parts contained in their own Area, each class library provides the Models, Views, and Controllers needed to provide a specific unit of functionality. For example: the AR library provides Accounts Receivable, and the SMO library provides SQL Server Management Objects. In this fashion, complete solutions can be constructed by starting with a parent website and including as many Assembly Parts as needed to complete the application. Since assemblies are constructed to work with any parent website, the same assemblies can be used in multiple solutions, greatly reducing the amount of code maintenance required.

## DNA (Parent Website)

The parent website serves as the target for Content Management features and provides the configuration and settings for the entire solution. If your application doesn’t need a parent website, simply redirect to /Dashboard.

## DNA3 (Dashboard Assembly Part)

The Dashboard Assembly Part provides Client, User, and User Identity Management, Authorization and Authentication Services, Content Management, and a single Migration file to create the database tables. Due to limitations in Migration Scripting capability, tables are created for every known module but are configured only when the system detects that a module actually exists. Based upon its presence or absence, each module will be added or removed from the dashboard menu automatically.

## SMO (SQL Server Management Objects)

The SQL Server Management Objects module provides Upload, Download, Backup, Restore, and Database Scripting capabilities to facilitate migration of data from one version of SQL Server to another. Automated backups can be scheduled and completed backups can be downloaded to your desktop on demand.

## Utilities Class Library

As its name suggests, the Utilities Class Library provides functionality common to all modules ranging from simple string manipulation to functionality specific to ADO.NET and Entity Framework. Classes intended for system wide use should be included in this project.

# Installation

Configuration settings are broken down into two parts: Common settings contained within the source code tree and application secrets defined in a separate file in a location of your choosing. There are only two secrets required for basic operation: A Database Connection String, and a random string for JWT encryption.

The system will run on Windows, Linux, and Docker based servers equipped with the .NET 8 Runtime. The following procedure will take only a few minutes to complete:

* Create C:\DNASettings.json and replace its content with the following:

{

"ConnectionStrings": {

"MainContext": "Server=your\_domainn\_ame.com;Database=DNA;User=dna;Password=random\_string;TrustServerCertificate=True;"`

},

}

* If you change the location of your secrets file, update the path in Program.cs
* Edit appsettings.json as needed, the settings are all self-explanatory and commented.
* Create a SQL Server database, user, and password for the application and update DNASettings.json accordingly.
* Run the CreateTables Migration Script located at DNA/DNA3/DNA3/Migrations
* Create a suitable SSL configured website to host your application.
* Publish the source code and navigate to <https://your_domain_name.com>

The database tables will be seeded and the dashboard menus adjusted automatically the first time you run the application and on each subsequest restart. If you add or remove modules, the dashboard menus will be adjusted accordingly, but all existing data will remain intact.

At this point you can sign in to the default administrator account using [admin@companyone.com](mailto:admin@companyone.com) for the user name and P@ssw0rd (note the zero) as the password.

# Customization

The parent website and dashboard modules can be themed separately. Themes consist of a Layout file, associated views, and the CSS and Javascript required for your template. Just about any standard template can be used with the only caveat being that your must break down pages into MVC compatible format. To function with the Content Management Features, each view will require a trivial amount of code to be inserted.