|  |  |
| --- | --- |
| **A blue and red logo  Description automatically generated** | **technical documentation**    **COBIT19 CONSULTANTS** |

Contents

[1. Introduction 2](#_Toc150422002)

[2. Technologies Used 3](#_Toc150422003)

[2.1 Microsoft Visual Studio 2022 3](#_Toc150422004)

[2.2 Blazor Framework 4](#_Toc150422005)

[2.3 Bootstrap 5](#_Toc150422006)

[2.4 Entity Framework Core 6](#_Toc150422007)

[2.5 Model-View-Controller (MVC) Pattern 7](#_Toc150422008)

[2.6 SQL Server Management Studio (SMSS) 8](#_Toc150422009)

[2.7 Syncfusion Essential Studio 9](#_Toc150422010)

[2.8 Docker 10](#_Toc150422011)

[2.9 GitHub 11](#_Toc150422012)

[2.10 Hosting 12](#_Toc150422013)

[3. Setup 12](#_Toc150422014)

[3.1 Visual Studio 12](#_Toc150422015)

[3.2 SQL Server 13](#_Toc150422016)

[3.3 SQL Server Management Studio (SSMS) 13](#_Toc150422017)

[4. Migration the COBIT19 Software Solution 13](#_Toc150422018)

[5. Migrations, Data Seeding and Data Scaffolding 15](#_Toc150422019)

[5.1 Migrations 15](#_Toc150422020)

[5.2 Data Seeding 16](#_Toc150422021)

[5.3 Data Scaffolding 17](#_Toc150422022)

[6. Syncfusion Essential Studio 18](#_Toc150422023)

[7. Dependencies and Version 22](#_Toc150422024)

[8. Front End Functionality 23](#_Toc150422025)

[8.1 Recommendations for future development 24](#_Toc150422026)

[9. Back End Functionality 25](#_Toc150422027)

[9.1 Recommendations for future development 28](#_Toc150422028)

# Introduction

This document provides a comprehensive analysis and clarification of the technical features of COBIT Consultants software solution. It includes a thorough description of the frameworks, programming languages, and coding processes used in the creation of this solution. COBIT Consultants created the software solution as a totally web-based application, with security and user ease as top priorities.

The COBIT 19 framework software is a solution by Phenyo Modisane. The client’s idea is based on the simplification and automation of audits within organizations. Information Technology (IT) audits are essential in the governance and management of an organization, they make it possible to manage and allocate tasks, resources and people within the organization. The overall effectiveness of an IT audit determines the productivity and functioning of an organization. At the end of the day an effective organization is equal to a productive organization which in turn equals to profitability. However, organizations are still using the legacy system, which can be time consuming, error prone and costly. Moreover, they may lead to inconsistencies and inefficiencies in the audit process.

The scale at which digital transformation takes place within organizations driving them to transform their business models and processes, which requires new approaches to governance and management of information technology; the mitigation and management of cyber security threats and the growing number of regulatory requirements related to information technology i.e. ISO and ITIL, all of these aforementioned factors are important when auditing an organization but the auditing task becomes daunting and prone to errors and inefficiencies when using the traditional manual auditing processes. Thankfully, these issues are addressed by the COBIT19 Framework software proposed by the client.

The client has commissioned the project team to develop software to address the mentioned factors, to simplify and automate audits with the objective of improving the performance of an organization. The COBIT19 framework will provide organizations with the opportunity to improve their IT governance and management practices, to give an organization an overview perspective of its efficiency and areas that need improvement.

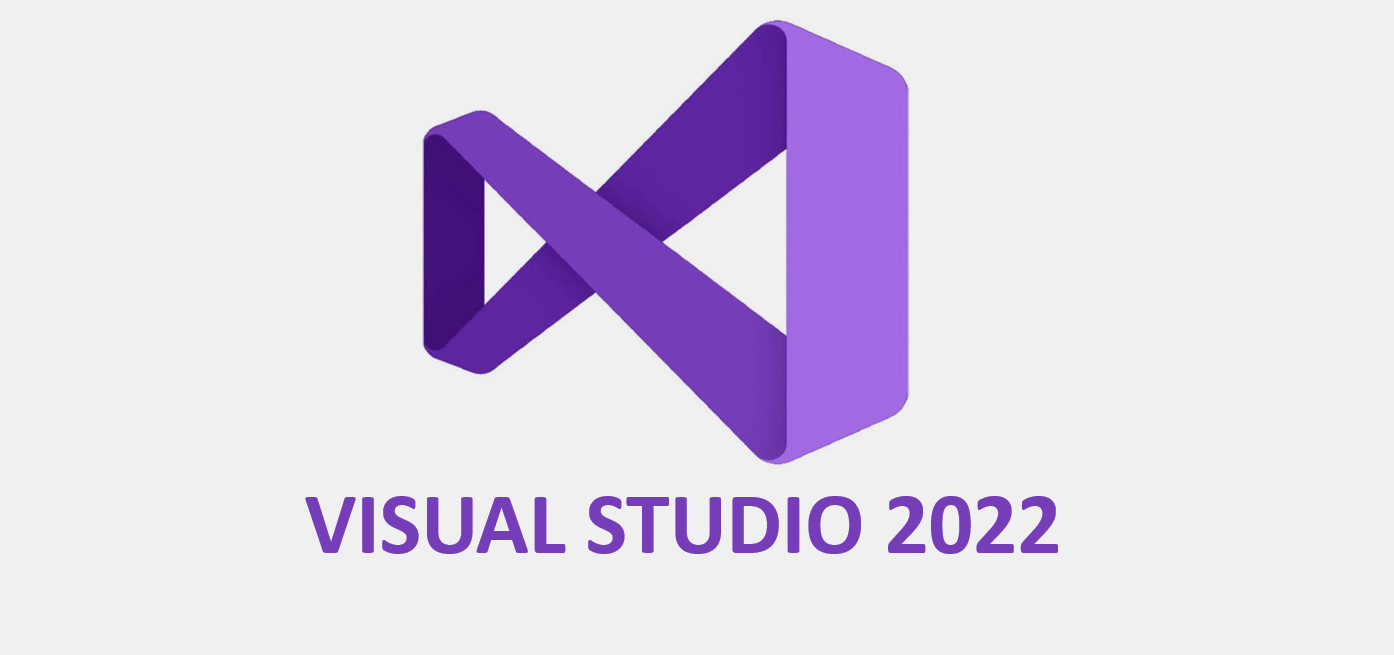
This document serves the following objectives:

* Comprehensive overview of the requisite development tools for the creation of this software solution.
* Provides guidance on how to configure the tools and environment so that it can be maintained in the future.
* Possible features to be added in the COBIT19 Software solution.

# Technologies Used

The following technologies utilized to develop this solution:

## Microsoft Visual Studio 2022



Microsoft Visual Studio[[1]](#footnote-1) is an integrated development environment (IDE). It is utilized in the creation of software like as websites, web applications, online services, and mobile apps. Visual Studio supports several computer languages and allows the code editor and debugger to handle (to varied degrees) practically any programming language as long as a language-specific service is available. It includes several built-in languages such as HTML, XML, Typescript, C++, JavaScript, CSS and many more. It was selected because of the following reasons:

* Simplicity: offers user-friendly experience, making it easy to learn for everyone and allowing users to write high quality applications.
* Copilot: AI-powered software for code completion, it seamlessly integrates well with Visual Studio and auto generates snippets of code.
* Intelligent code editing: includes syntax highlighting which helps users understand their errors better and write their code efficiently.
* Comprehensive debugging tools: provides tools that developers can utilize to analyse their errors and fix them, tools such as active debugging, breakpoints.
* Team collaboration: it has built in version control that allows users to collaborate in one project, review code, and track changes.
* Cross-platform development: allows developers to develop applications for cross platform systems, such as Linux, MacOS and Windows.
* Open source: has comprehensive documentation to help new developers and allows them to communicate in a wider community, this allows collaboration in the development environment.

## Blazor Framework



Blazor[[2]](#footnote-2) is a web framework that allows you to create web UI components (Razor components) that may be hosted in a variety of ways. Razor components may be executed server-side in ASP.NET Core (Blazor Server) or client-side in a browser using a WebAssembly-based.NET runtime (Blazor WebAssembly, Blazor WASM). In this project the Blazor Server was utilized. Razor components may also be hosted in native mobile and desktop apps that render to an embedded Web View control (Blazor Hybrid). The method you develop Razor components is the same regardless of the hosting model. The identical Razor components may be used with any of the hosting types. It was selected because of the following reasons:

* **Integration with .NET Ecosystem:** interacts smoothly with the larger.NET environment, allowing developers to take advantage of their current knowledge and abilities in C# and .NET packages. This implies that while creating interactive web apps, developers may use common tools, frameworks, and procedures.
* **Cross-platform compatibility:** allows you to create web apps that operate on a variety of operating platforms, including Microsoft Windows, macOS, and Ubuntu. This cross-platform interoperability guarantees that the programs reach a wide audience and are usable in a variety of user scenarios.
* **Full-stack development with C#:** enables developers to use C# to execute full-stack development. This implies that developers may write clients-side and server-side programs in the same language (C#), easing the creation process and providing for seamless interaction between both the client and the server.
* **Authorization capabilities:** includes built-in functionality for authorizing and authenticating web apps. This enables developers to simply incorporate authentication for users, access control based on roles, and other safety features, assuring the security and protection of applications.
* **Password Hashing:** supports secure hashing of passwords techniques, which are critical for user credential protection. Passwords are therefore securely saved and cannot be readily hacked in the case of a security attack.

## Bootstrap

**A purple square with a white b in the middle

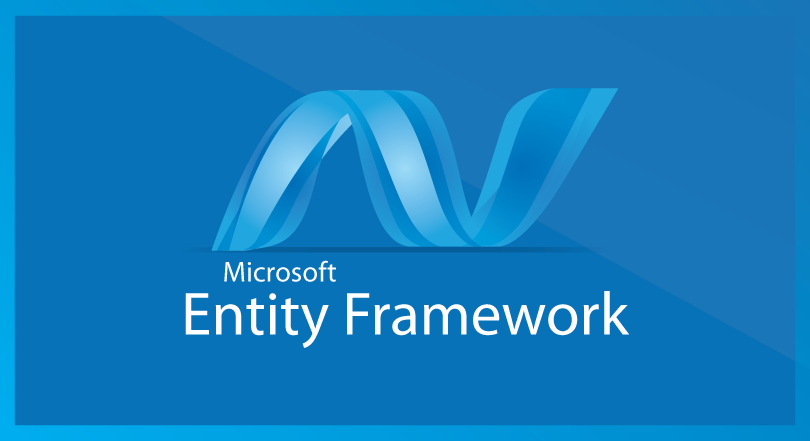
Description automatically generated**

Bootstrap[[3]](#footnote-3) is a versatile web development framework designed for creating fast, mobile-responsive front-end interfaces. It comes packed with pre-designed templates for essential interface components like buttons, forms, navigation bars, and fonts. Additionally, Bootstrap not only emphasizes reusable CSS code but also offers support for JavaScript programming, making it compatible with all front-end languages. The framework's remarkable efficiency and time-saving capabilities have made it an essential tool for professional developers.

Even without its code-simplifying benefits, Bootstrap remains a cornerstone in web development. Upon integration into a project, Bootstrap immediately establishes a unified set of style definitions across all HTML code, resulting in a seamless, mobile-optimized foundation. Developers can then effortlessly customize this baseline structure to align with their specific design preferences. It dedicated to crafting responsive and mobile-friendly front-end web applications. It equips developers with design templates encompassing various elements such as typography, forms, buttons, navigation, and more. It was selected because of the following reasons:

* **Consistent design:** Developers may produce a uniform and professional design across multiple pages and areas of a website by utilizing Bootstrap, which makes the user experience (UX) integrated and professional.
* **Cross-browser compatibility**: Bootstrap is meant to function seamlessly across several browsers, eliminating the need for rigorous testing and debugging for compatibility concerns.
* **Responsive grid system:** The grid structure of Bootstrap provides for easy content organization while also guaranteeing that it adjusts to different screen widths.
* **Extensive documentation:** Bootstrap has comprehensive documentation, tutorials, and examples, making it suitable for developers of all ability levels. This shortens the learning curve and speeds up progress.
* **Pre-styled components:** Bootstrap has a variety of pre-styled elements such as navigation bars, buttons, forms, and more. These are easily customizable to meet the exact design needs of a project.

## Entity Framework Core

****

Entity framework[[4]](#footnote-4) (EF) core is a lightweight, extensible open, source and cross-platform version of a popular entity framework data access technology. From that facilitates the interaction between applications and data by service as an Object/Relational mapping (ORM) framework. For .NET developers, it alleviates the need to constantly adapt to database modifications.

It establishes a link between the structure of a relational database and corresponding objects, offering a level of abstraction. Within EF, you have the ability to create Entity classes that are not tied to a specific database layout, and then connect them to the tables and relationships within the database. Each of these entities has their owned defined structure. It was selected because of the following reasons:

* **Simplicity:** Core streamlines data access by abstracting the database at a high level, allowing programmers to interact with familiar object-oriented structures instead of writing raw SQL queries.
* **Cross-platform support:** EF Core is intended for use on a variety of platforms, include macOS, Windows, and Linux. This makes it appropriate for projects aimed at many operating systems.
* **Fast and lightweight:** compact and efficient. It has a reduced amount of memory than the previous version, making it appropriate for resource-constrained situations.
* **Extensibility:** EFCore is intended to be extensible, allowing programmers to adapt and expand its functionality to match the needs of unique projects. This is important for dealing with complex circumstances or interacting with specialist databases.
* **Support for Multiple Database providers:** Microsoft SQL Server, PostgreSQL, MySQL, SQLite, and other database providers are supported by EF Core. This allows developers to operate with many database systems without requiring large code modifications.

## Model-View-Controller (MVC) Pattern

A diagram of a diagram

Description automatically generated

The MVC[[5]](#footnote-5) architectural pattern, which consists of "service providers," is a design pattern used to decouple the user interface (view), data (model), and application logic (controller). This pattern aids in achieving a separation of concerns. The Model-View-Controller (MVC) pattern is widely employed in software development, particularly in web applications. Each component has a defined role and responsibility, contributing to a clean and organized codebase.

* Model – represents the data and business logic of the software, it is responsible for managing the state, and interacting with the database. In web-based applications it represents tables or documents which handle data retrieval and manipulation.
* View – is responsible for presenting the data to the user. It defines how the information is displayed and provides the user interface.
* Controller – acts as an intermediary between the Model and the View. It receives user input from the View, processes it (often involving interaction with the Model), and updates the View accordingly.

The MVC design pattern was selected because of the following:

* MVC responsibilities are divided between the client & server, compatible with web application architecture.
* MVC is helpful design pattern when planning development.
* Separation of Concerns: that code is divided based on function to either the model, view, or controller bucket.
* Loosely Coupled and code reusable.
* Removes unnecessary dependencies.
* Code reuse

A reference of the MVC architectural pattern can be found it in this link: [Blogifier architecture](https://github.com/blogifierdotnet/Blogifier).

## SQL Server Management Studio (SMSS)

A logo of a yellow cylinder and a screwdriver

Description automatically generated

Microsoft SQL Server Management Studio[[6]](#footnote-6) is a powerful Microsoft software tool used for establishing, managing, and managing all Microsoft components. SMSS enables the building of sophisticated databases with referential integrity maintained by relationships. It was selected because of the following reasons:

* **Integration with source control:** interacts smoothly with source control systems, enabling database developers to version-manage their applications. This guarantees that changes are tracked, allowing for cooperation, version history, and the simplification of deployment operations.
* **Comprehensive database management:** provides an extensive feature set for managing all aspects of the Microsoft SQL Server databases. Database development, creation, management, recovery and backup, security, performance optimization, and monitoring are examples of such jobs. Because of its broad variety of functions, it is a flexible and necessary tool for database experts.
* **Multi-language support:** supports a wide variety of programming languages, allowing developers to work in multiple tongues as long as a language-specific API is provided.
* **Intuitive user interface:** provides an easy and structured environment that is suitable to both seasoned and inexperienced database experts.

## Syncfusion Essential Studio

A blue text on a black background

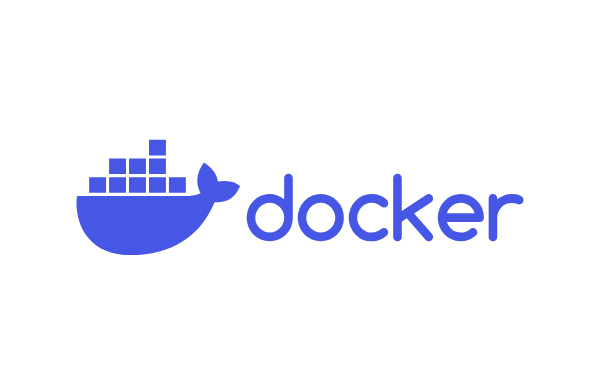
Description automatically generated

Syncfusion Essential Studio[[7]](#footnote-7) is a renowned source of enterprise-class development components and libraries that enable a wide variety of UI, reporting, and business intelligence features on all major Windows platforms. Each product boasts a distinctive and invaluable feature set, becoming an indispensable tool in every developer's toolkit. By utilizing Syncfusion’s components, you can streamline the development process, allowing you to focus on the core business logic of your applications.

Complementing Syncfusion's offerings is Bootstrap, a dynamic web development framework designed for swift and mobile-responsive front-end programming. It provides pre-designed templates for essential interface elements like buttons, forms, navigation bars, and fonts. Bootstrap's primary focus is on identifying and sharing reusable CSS code, while also extending support for JavaScript programming and seamless integration with all front-end languages.

* **Comprehensive component library:** Syncfusion provides an extensive array of over 1,600 UI components for a variety of platforms, including Windows Forms, ASP.NET, ASP.NET MVC, Back Office, Silverlight, Business Intelligence, and WPF applications, empowering you to seamlessly integrate innovation.
* **Data visualisation:** Syncfusion provides comprehensive data visualization features such as charts, graphs, and mappings. These components allow developers to generate visually appealing and meaningful data visualizations.
* **Learning resources and extensive documentation:** offers rich documentation, tutorials, and learning materials. This enables developers to get started fast with the platform while making the most of its features.
* **Integration with Visual Studio**: Syncfusion components interface smoothly with popular programming environments such as Visual Studio, providing programmers with an effortless and familiar working environment.

## Docker



Docker[[8]](#footnote-8) is a software platform that enables rapid development, testing, and deployment of programs. Docker organizes software into standardized units called containers, which include everything the software requires to execute, such as libraries, system tools, code, and runtime. Docker allows you to swiftly deploy and grow apps in any environment while ensuring the code will execute.

Docker works by offering a standardized method for running your programs. Docker is a container operating system. Containers virtualize a server's operating system in the same way as virtual machines virtualize (remove the need to physically manage) server hardware. Docker is installed on each server and provides easy commands for building, starting, and stopping containers. It was selected because of the following reasons:

* **Consistent development environment:** keeps developers working in a consistent environment across their development lifecycle.
* **Portability:** Docker containers may run on any Docker-enabled system, independent of the underlying infrastructure. This allows programs to be readily relocated and deployed across several environments, whether on-premises, in the cloud, or on developer workstations, providing consistent behaviour across multiple settings.
* **Version control:** supports container image versioning and storage in registries. This guarantees that particular versions of apps may be duplicated and deployed with confidence at any moment.
* **Scalability:** Docker's container-based design enables compact and effective application scaling. Containers may be launched and terminated quickly and controlled using technologies such as Docker Swarm or Kubernetes for optimal resource use and high availability, making it a good choice for large-scale deployments.
* **Easier testing and continuous integration**

## GitHub



GitHub[[9]](#footnote-9) is a platform for hosting code that allows for version control and collaboration. It allows you and others to collaborate on projects from anywhere. GitHub supports collaborative coding by offering a hosting service and online interface for the Git code repository, as well as collaboration management tools. Think of the developer platform as a social networking site for software engineers. Members may follow one another, rate one another's work, get updates on specific open-source projects, and chat openly or privately. It was selected because of the following reasons:

* **Collaborative development**: GitHub is a website that allows developers from all around the world to work together on projects. It encourages teamwork by providing tools for version control, pull requests, and team management.
* **Version control:** GitHub uses Git, a strong version control system, to monitor code changes. Developers may maintain several versions of their projects, evaluate changes, and revert to prior states as needed.
* **Issue tracking and management:** GitHub has an issue tracker built in that allows teams to report, prioritize, and manage tasks, issues, and requests for features. This feature simplifies project management by ensuring that activities are properly structured, monitored, and addressed.
* **Code reviews and feedback:** Before integrating changes into the main branch, GitHub's pull request mechanism allows team members to offer input, suggest improvements, and guarantee code quality.
* **Integration with Visual Studio:** GitHub works perfectly with Microsoft Visual Studio, creating a streamlined workflow for developers who use this popular integrated programming environment (IDE). This connection boosts productivity by allowing developers to use their favourite tools while still taking use of GitHub's version control and collaboration capabilities.

## Hosting



Currently the system is hosted on dedicated servers provided by the NWU and maintained by the Masters students. Currently no integrated deployment pipeline is in place so deployment needs to be done manually. To interact with the server PuTTY is used to provide terminal access to the server. The following steps is currently used to upload new versions of the software to the server.

* Login to server (from within the NWU intranet) using the following credentials
  + IP : 196.252.135.196
  + http://196.252.135.195:8001/Identity/Account/Login?ReturnUrl=%2F
  + Login : cobitdev
  + Password : iWillNeverBeEnough
* Redirect to Cobit-19 file using the following command : cd cobit19/Cobit-19
* Stop the current Docker container using : docker-compose down
* Delete the current docker image using: docker image rm cobit19
* Pull latest version of code from Github using: git pull origin
* Restart the docker compose container with: docker-compose up -d
* The image should then rebuild entirely.

# Setup

## Visual Studio

To install Visual Studio, follow the following steps:

1. Navigate to the Visual Studio official website [Visual Studio](https://visualstudio.microsoft.com/), Click the “Download Visual Studio Community 2022”, since it is free and contains all the necessary tools.
2. Once it is downloaded run the installer.
3. Select the workloads, they are development tools and components, in this case it will be:
   1. ASP.NET and web development
   2. .NET Multi-platform App UI development
   3. .NET desktop development
   4. Desktop development with C++
4. Choose a location where you want to install Visual Studio in your machine, usually the default location is utilized.
5. Click on the install button, it might take some time depending on the internet connection.
6. Launch Visual Studio.

## SQL Server

To install SQL Server, follow the following steps:

1. Navigate to the SQL Server official website [SQL Server](https://www.microsoft.com/en-us/sql-server/sql-server-downloads).
2. Choose the Developer Edition and download it.
3. Once it is downloaded run the installer.
4. Choose the Basic installation type and accept the license terms.
5. Install it in the default location and click Install when it is done press Install SSMS, the steps for downloading it will be below.

## SQL Server Management Studio (SSMS)

To install SSMS, follow the following steps:

1. Navigate to the SSMS official website [SSMS](https://learn.microsoft.com/en-us/sql/ssms/download-sql-server-management-studio-ssms?view=sql-server-ver16) v19.1.
2. Once it is downloaded run the installer.
3. Choose the default installation options and accept the license terms.
4. Click on the install button, it might take some time depending on the internet connection.
5. Once it is downloaded, you can launch SQL Server Management Studio
6. Select the workloads, they are development tools and components, in this case it will be:

# Migration the COBIT19 Software Solution

The first step into utilizing the Software Solution is to clone the repository from the GitHub account [Cobit19NWU](https://github.com/Cobit19NWU/Cobit-19).

This project can be cloned either using GitHub for desktop, Git terminal or Visual Studio.

**GitHub for desktop**

1. Download the GitHub Desktop from the official side [GitHub Desktop](https://desktop.github.com/).
2. Lauch it and sign in with your GitHub account.
3. Click on “File” and select “Clone repository” and select the URL tab.
4. Choose a path where you want to clone your repository and click the button clone.

**Git Terminal**

1. Open the preferred terminal with Git support.
2. Use the following prompt “git clone (link)”

**Microsoft SQL Server Management Studio**

After cloning the project make sure you make the sure that the Server name is “localhost” since so that it will run in your local machine, and you can make the data seeding local then connect.

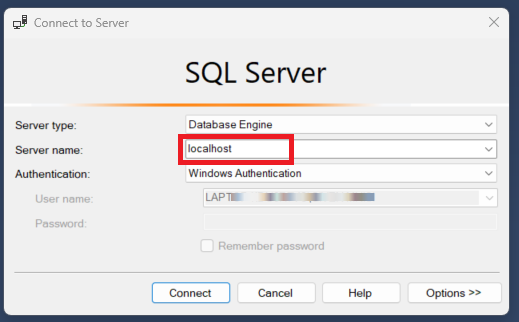


Figure Database configuration

**Visual Studio**

1. Open visual studio, in the first window you will see the following picture:

A screenshot of a computer

Description automatically generated

Figure Cloning a repository.

1. Click the Clone a repository button and enter your link.
2. Click the “Clone” button.

**Restoring NuGet Packages**

After cloning the project, open it in Visual Studio and select the Solution and on the right-side and restore the NuGet Packages which will be listed in this document. This will download and restore all the required dependencies for the project.

**Database Migration**

* Open the Package Manager Console in Visual Studio (Tools -> NuGet Package Manager -> Package Manager Console).
* In the console window, run the following command to apply database migrations: “**Update -Database”**. This command triggers the Entity Framework migration system which is a way to manage your database schema over time, it checks for any pending migrations (changes to the schema) and applies them to your database, which makes sure that your codebase aligns with your database structure.
* This database can be found in the SSMS, as
* With regards to database, there is a concept of **migrations, data seeding and data scaffolding** which is critical for this project which will be discussed for future development.

**Run the Project**

After the database is created, you can run the project.

# Migrations, Data Seeding and Data Scaffolding

## Migrations

When developing web applications, the application needs to interact with the database to store and retrieve data. Migrations help you manage these changes in a structured and controlled manner. They provide a way to define and apply incremental updates to the database schema while preserving existing data.

* Migration is offered by a certain framework that is utilized in this case it is Entity Framework, a migration is a code file that describes the changes you want to make to the database schema. It contains instructions on how to update the database.
* A migration can be created with this command ‘Add-Migration (nameofMigration)’
* Open the generated migration file (located in the Migrations folder below). This file contains C# code that represents the changes to be made in the database schema.
* Once you're satisfied with the changes in the migration file, apply the migration to your database using the ‘Update-Database’

A screenshot of a computer

Description automatically generated

Figure Migrations

## Data Seeding

* **Data seeding** is the process of populating a database with certain data. This is useful when there are pre-defined data that need to be recorded. In this case it is the COBIT Core Model since it consists of several questions that needs to be asked to an organisation.
* These questions are stored in a JSON file since it is easier to read, and simple to understand.
* Below is an attached image of the Data seeding that need to be seeded for future development for other focus areas.

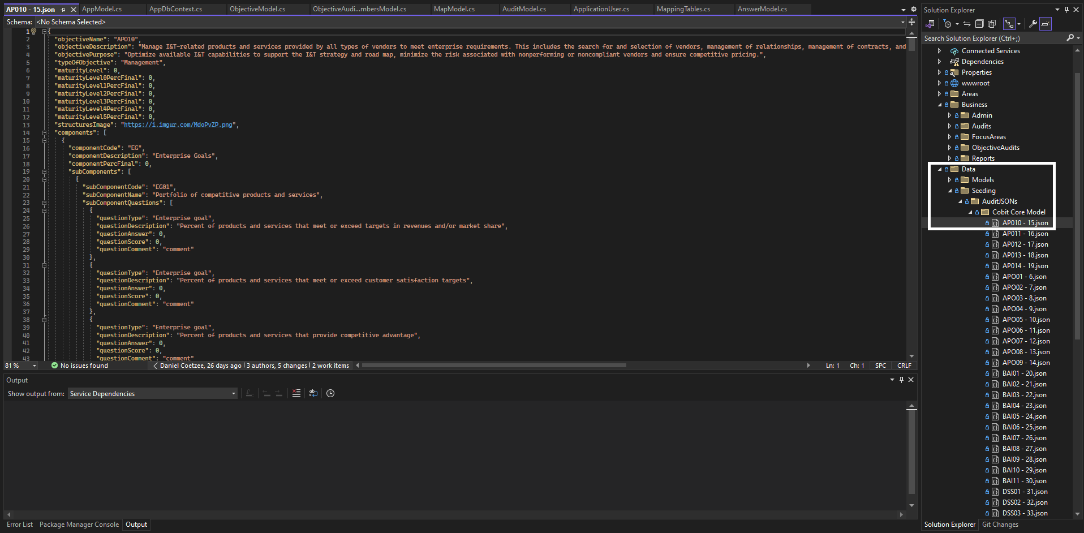


Figure Data seeding for COBIT Core Model

Below is a picture of the COBIT Core Model data Seeding and its template, additional Focus areas such as Information Security, Small and Medium Enterprises etc. need to be also data seeded.

A screenshot of a computer

Description automatically generated

Figure Data Seeding

## Data Scaffolding

* Scaffolding in software development refers to the automated generation of code, typically based on existing templates or models. In the context of databases, scaffolding is about automatically generating the code needed to interact with a database based on its structure.
* This includes generating models, controllers, views, and other components that are necessary for CRUD (Create, Read, Update, Delete) operations. Scaffolding saves developers time and effort by automating the boilerplate code.

Essentially after running ‘Update -Database’, Entity Framework not only applies any pending migrations but also has the capability to automatically execute data seeding scripts and generate necessary code (scaffolding) for interacting with the database. This combined process ensures that your database is up-to-date and populated with initial data, making your application ready for use upon deployment. Below is a picture of the Cobit database and its respective tables in SSMS:

A screenshot of a computer

Description automatically generated

Figure CobitDb in SSMS

# Syncfusion Essential Studio

When utilizing the Software solution especially for displaying the charts, you will come across a message ‘This application was built using a trial version of Syncfusion Essential Studio. To remove the license validation message permanently, a valid license key must be included. [Claim your free account](https://www.syncfusion.com/account/login)’

This is specific to Syncfusion, as it requires a license. The license is available for free, but to obtain one, you'll need to create an account.

* The first step is to create an account in the Syncfusion website.
* After signing up, go to License & downloads and click claim license key as in the image below:

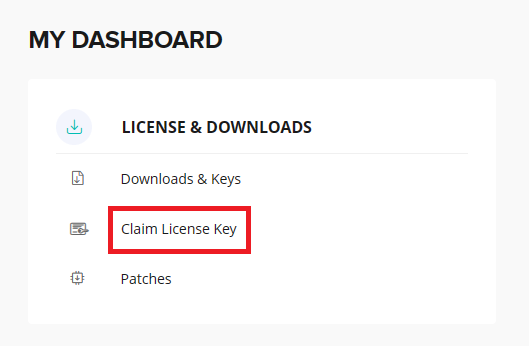


Figure Claim license key

* You will be presented with 3 options, choose the ‘Free Community License’ as in the image below:

A screenshot of a computer

Description automatically generated

Figure Community License

* After filling in the form you will receive a message saying ‘Your form was submitted successfully Thank you for submitting your request for a free community license. We have received your application and created a ticket with the reference number #5175639. We will validate your request within the next 48 business hours and will get back to you with the status of your application as soon as possible. In the meantime, we have sent you an email containing a 7-day trial key for Essential Studio. Please check your inbox for further instructions and the key’
* The license key will be sent via email.
* To incorporate the Syncfusion community license into your development environment, ‘search environment variables’ in your machine and press this button.

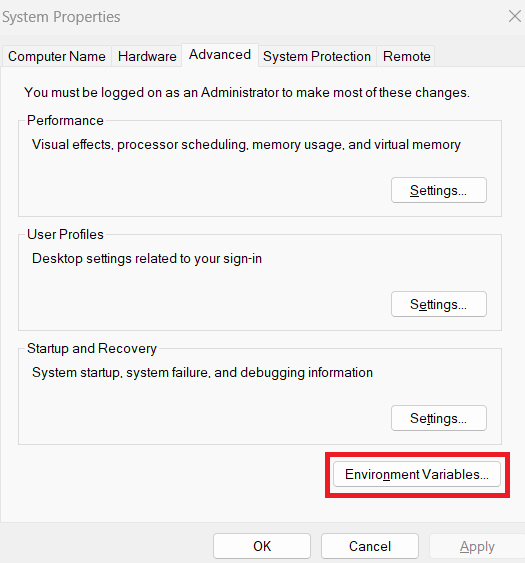


Figure System environment

* Then click ‘New’ to add a new system variables.

A screenshot of a computer program

Description automatically generated

Figure System variables

* The variable name should be ‘SYNCFUSION\_KEY’ and the variable value should be the key itself.

A screenshot of a computer

Description automatically generated

Figure New system variable

# Dependencies and Version

Table List of all the dependencies needed to run the project.

|  |  |
| --- | --- |
| Dependencies | Version |
| AutoMapper | 12.0.1 |
| Automapper.Extensions.Microsoft.DependencyInjection | 12.0.1 |
| Blazored.Toast | 4.1.0 |
| FontAwesome | 4.7.0 |
| Microsoft.AspNetCore.Diagnostics.EntityFrameworkCore | 6.0.19 |
| Microsoft.AspNetCore.Identity.EntityFrameworkCore | 7.0.5 |
| Microsoft.AspNetCore.Identity.UI | 6.0.18 |
| Microsoft.EntityFrameworkCore.SqlServer | 7.0.5 |
| Microsoft.EntityFrameworkCore.Tools | 7.0.5 |
| Microsoft.VisualStudio.Web.CodeGeneration.Design | 6.0.14 |
| Microsoft.VisualStudio.Azure.Containers.Tools.Targets | 1.17.0 |
| Microsoft.VisualStudio.Web.CodeGeneration.Design | 6.0.14 |
| Newtonsoft.Json | 13.0.3 |
| Syncfusion.Blazor.Charts | 22.1.34 |
| Syncfusion.Blazor.Circulargauge | 22.1.34 |
| Syncfusion.Blazor.Lineargauge | 22.1.34 |
| Syncfusion.Blazor.Notifications | 22.1.34 |
| Syncfusion.Blazor.ProgressBar | 22.1.34 |
| Syncfusion.Pdf.Net.Core | 22.1.34 |

# Front End Functionality

The front-end functionality pages can be located in the Solution Explorer tab, specifically under the "Pages" folder.

Table Front-end functionality

|  |  |
| --- | --- |
| COMPONENTS | Description |
| Login.cshtml | The web page is a user-friendly login form where individuals can input their email and password. It allows users to choose a "Remember Me" option and, upon clicking the "Login" button, sends the provided data to the server for authentication. The page displays validation messages for any input errors and offers a link for password recovery. It's part of a larger web application, serving as the interface for user login and interaction with the server to access their account. |
| Register.cshtml | This page implements validation techniques to ensure that new users enter only valid data for their profile to be saved. |
| *Administrator* | |
| UserManagement.razor | This page functions as the principal centre for user-related processes. Account creation, update, and removal are all possible. This module provides administrators with effective access control, including role assignment and permission customisation. |
| *Audit* | |
| Audit.razor | This page captures the name of the new audit. |
| AuditSelect.razor | Where the authorized user can choose between the audits that they have created or participated in to be displayed on the UserHome.razor |
| AuditSelectTable.razor | This is page presents a table displaying Active and Inactive audits, as well as those available for selection. This page offers a clear overview of the audit status, facilitating easy selection and management. |
| ManageAudits.razor | These pages comprise functionalities for creating, deleting, and assigning audits to specific users. This component serves as a comprehensive tool for administrators to oversee the entire audit lifecycle, ensuring seamless management and allocation of responsibilities. |
| *DesignFactor* | |
| DesignFactor.razor | This page encompasses of the ten Design Factors that need to be rated between a scale in terms of importance |
| DesignFactorChart.razor | This displays the several charts using Syncfusion of the different objectives. |
| DesignFactorReport.razor | This page consists of a comprehensive report in PDF form of the Design Factors, showcasing the results of all the input captured from the design factors pages. |
| *Home* | |
| FocusAreaSelectTable.razor | This displays the Table of the name of the focus area, and status of it whether it is complete or not. |
| UserHome.razor | This is the home page for higher level users of the software application. It consists of a built in training of what the software is about, it is a dashboard of the software application. |
| *ObjectiveAudit* | |
| Component.razor | This page consists of the different components of an Objective. |
| ComponentSelectBar.razor | This page is where the user selects the different components which consists of the Alignment goals, processes, Information Flow and many more. |
| ObjectiveAuditMember.razor | This page assigns an auditor who is responsible for administering an audit on an objective |
| ObjectiveAuditScope.razor | This page presents the high, medium, and low importance objectives that Audits should look out for. |
| ObjectiveAuditSelect.razor | This page assesses the maturity of an Audit, when an audit is selected it will be assessed based on the Enterprise goals, Alignment goals, processes and many more |
| QuestionTable.razor | The questions of the specific knowledge area. |
| SubComponent.razor | Displays the relevant input areas of each of the components of the selected knowledge area. There are multiple pages for each of the knowledge areas. |
| SubComponentTable.razor | The table where the user can select which component that the user wants to audit of the selected knowledge area. |
| *Reporting* | |
| Reports | Where reports are generated and can be downloaded. |

## Recommendations for future development

Future development teams have some recommended developments that they can contribute to the COBIT19 Software:

* **Generate reports using GPT technology:** implement a reporting feature leveraging GPT (Generative Pre-trained Transformer) technology which can generate more insightful and detailed reports.
* **Setup Collaboration Tools:** Integrate essential collaboration tools such as real-time chat, notifications, and task assignment features. This will streamline team communication and cooperation within the software, enhancing productivity and efficiency.
* **Enhance User Experience and Flow:** Address the current user experience concerns by focusing on the software's flow and look. Conduct a comprehensive UI (User Interface) overhaul to improve navigation, optimize layouts, and ensure a seamless and intuitive user interaction. This will significantly enhance the overall user satisfaction and usability of the software.

# Back End Functionality

Table :Back End

|  |  |
| --- | --- |
| Component Name | Description |
| UserManagementProvider.cs | This class is a backend logic for user and role management utilizing the Microsoft Identity framework and AutoMapper for data mapping. It relies on various dependencies, including AutoMapper and Microsoft Identity components, to facilitate tasks such as user retrieval, creation, updating, and deletion. Additionally, it manages user roles and provides methods for adding users to specific roles and retrieving role information. The class also offers functionalities for obtaining user IDs and real names based on a ClaimsPrincipal object. It is a fundamental component for handling user management, authorization, and role-based access within the application's backend logic. |
| AuditorProvider.cs | The "AuditProvider" class is a central component of the web application's audit management system. It manages and facilitates various operations related to audits, design factors, and objectives. This class interacts with a database, allowing users to create, retrieve, update, and delete audits, design factors, and associated data. It also provides methods for checking user membership in audits and managing audit members. Additionally, the class relies on dependencies like AutoMapper for data mapping, the AppDbContext for database access, and other providers for objective audits and user management. |
| MappingService.cs | The "MappingService" class performs calculations and data mapping for objectives in the application. It consists of two main methods: "Calculate" and "CalculateSummary." The "Calculate" method computes objective values by analyzing design factors and their associated questions, taking into account weights, answers, baselines, and odds. It then calculates the relative importance of each objective based on these factors. The "CalculateSummary" method, on the other hand, aggregates objective values across multiple design factors to provide a summarized view of the relative importance of objectives. This class facilitates data analysis and supporting decision-making within the application's audit management system by evaluating the significance of objectives in relation to design factors. |
| FocusAreaProvider.cs | The "FocusAreaProvider" class is responsible for managing focus areas, audits, and user subscriptions. It offers features like retrieving focus areas based on user subscriptions, accessing audits within specific focus areas, and determining the completion and activity status of focus areas. Additionally, it facilitates the retrieval of the last audit associated with a focus area and enables navigation between design factors. The class also allows users to be added to focus areas by creating subscription entries. Overall, "FocusAreaProvider" plays a vital role in controlling user access to application data and audits, monitoring the status of focus areas, and facilitating seamless navigation within the audit management system. |
| AudiJSONParsingService.cs | The "AuditJSONParsingService" class facilitates the conversion of objective audit templates between JSON and structured data representations, making it a valuable component for working with these templates in the application's backend. |
| ObjectiveAuditProvider.cs | The "ObjectiveAuditProvider" is responsible for managing objective audits, handling user membership, and performing various calculations and assessments related to the auditing process. |
| MappingTables.cs | The “MappingTable” class automates the process of generating and mapping data for various design factors. It iterates through 40 objectives and multiple questions for each design factor, assigning and storing weights for these combinations in a database using Entity Framework. It is part of the application aimed at associating weights with objectives and questions to evaluate different design factors effectively, facilitating data population for decision-making processes. |
| GoalsCascadeReport.cs | The a "GoalsCascadeReport" class constructs detailed reports. The class allows users to input information about assessment data and design factors, and it creates a structured PDF document with multiple pages. Each page includes elements like text, tables, and images. It's a valuable tool allowing for automatic generation of reports that can be stored, or delivered to users. |
| ObjectiveAuditReport.cs | The "ObjectiveAuditReport" class is responsible for generating PDF reports related to objective audits. It initializes a PDF document, creates report pages, and adds information about the assessment, objectives, and audit components. The generated PDF reports include details about the organization, assessment, lead assessor, auditor, and audit information. The report also covers the maturity assessment and specific processes. |
| ReportProvider.cs | The ReportProvider class serves as a backend component responsible for generating PDF reports in the application. It interfaces with the AppDbContext to access and process data, primarily producing reports for objective audits and goals cascade analysis. These reports play a crucial role in data analysis and documentation. The class encapsulates methods for creating these reports, such as createObjectiveAuditReport and createGoalsCascadeReport, enabling seamless report generation and management within the application's backend functionality. |
| AppDBContext.cs | The AppDBContext.cs file defines a database context for the application, configured for Entity Framework. It handles database interactions and includes entity configurations for tables, data seeding for various entities, and relationships between them. The code seeds data like weights for different design factors, audit templates from JSON files, and audit instances associated with members. This code serves to set up and initialize the application's database, establish schema, and populate it with sample data, facilitating the management of objectives, questions, audit templates, audits, and their members. To use the “AppDBContext”, ensure proper Entity Framework setup and execution during database initialization. |
| Program.cs | “Program.cs” configures the Microsoft ASP.NET Core application. It sets up services, dependencies, and configurations for the web application. Notable aspects include configuring a database context (with options for development and production environments), enabling user authentication with Identity, adding SignalR for real-time functionality, and integrating libraries like Syncfusion.Blazor and Blazored.Toast. The code initializes the application, handles migrations, and defines request pipelines, allowing the application to run as a web server. It maps routes, sets up authentication and authorization, and runs the application. Furthermore, it registers licenses for Syncfusion components and sets up various providers for user management, audits, focus areas, objective audits, and reports. Finally, it distinguishes between development and production environments and provides error handling. |

## 9.1 Recommendations for future development

Future development teams have some recommended developments that they can contribute to the COBIT19 Software:

* **Setup CI/CD Pipeline:** Implement a Continuous Integration/Continuous Deployment (CI/CD) pipeline to automate the process of building, testing, and deploying code changes. This will ensure that new updates are seamlessly integrated into the software and deployed to production without manual intervention, leading to faster and more reliable releases.
* **Decouple Frontend and Backend:** Adopt a decoupled architecture, separating the frontend and backend components. This design approach allows for greater flexibility, scalability, and maintainability of the software. It enables independent development and deployment of front-end and backend services, facilitating easier updates and enhancements.
* **Host Server on Cloud Platform:** choose a reputable cloud platform (e.g., AWS, Azure, Google Cloud) to host the server. Utilize the benefits of cloud infrastructure, such as scalability, high availability, and security. Deploying the server on the cloud ensures optimal performance, reliability, and accessibility for users, regardless of their location. It also allows for easy scaling as the user base grows.

1. <https://visualstudio.microsoft.com/vs/> [↑](#footnote-ref-1)
2. <https://dotnet.microsoft.com/en-us/apps/aspnet/web-apps/blazor> [↑](#footnote-ref-2)
3. [Bootstrap · The most popular HTML, CSS, and JS library in the world. (getbootstrap.com)](https://getbootstrap.com/) [↑](#footnote-ref-3)
4. [Overview of Entity Framework Core - EF Core | Microsoft Learn](https://learn.microsoft.com/en-us/ef/core/) [↑](#footnote-ref-4)
5. [ASP.NET MVC Pattern | .NET (microsoft.com)](https://dotnet.microsoft.com/en-us/apps/aspnet/mvc) [↑](#footnote-ref-5)
6. <https://learn.microsoft.com/en-us/sql/ssms/download-sql-server-management-studio-ssms?view=sql-server-ver16> [↑](#footnote-ref-6)
7. [.NET, Xamarin, JavaScript, Angular UI components | Syncfusion](https://www.syncfusion.com/) [↑](#footnote-ref-7)
8. <https://www.docker.com/> [↑](#footnote-ref-8)
9. <https://github.com/> [↑](#footnote-ref-9)