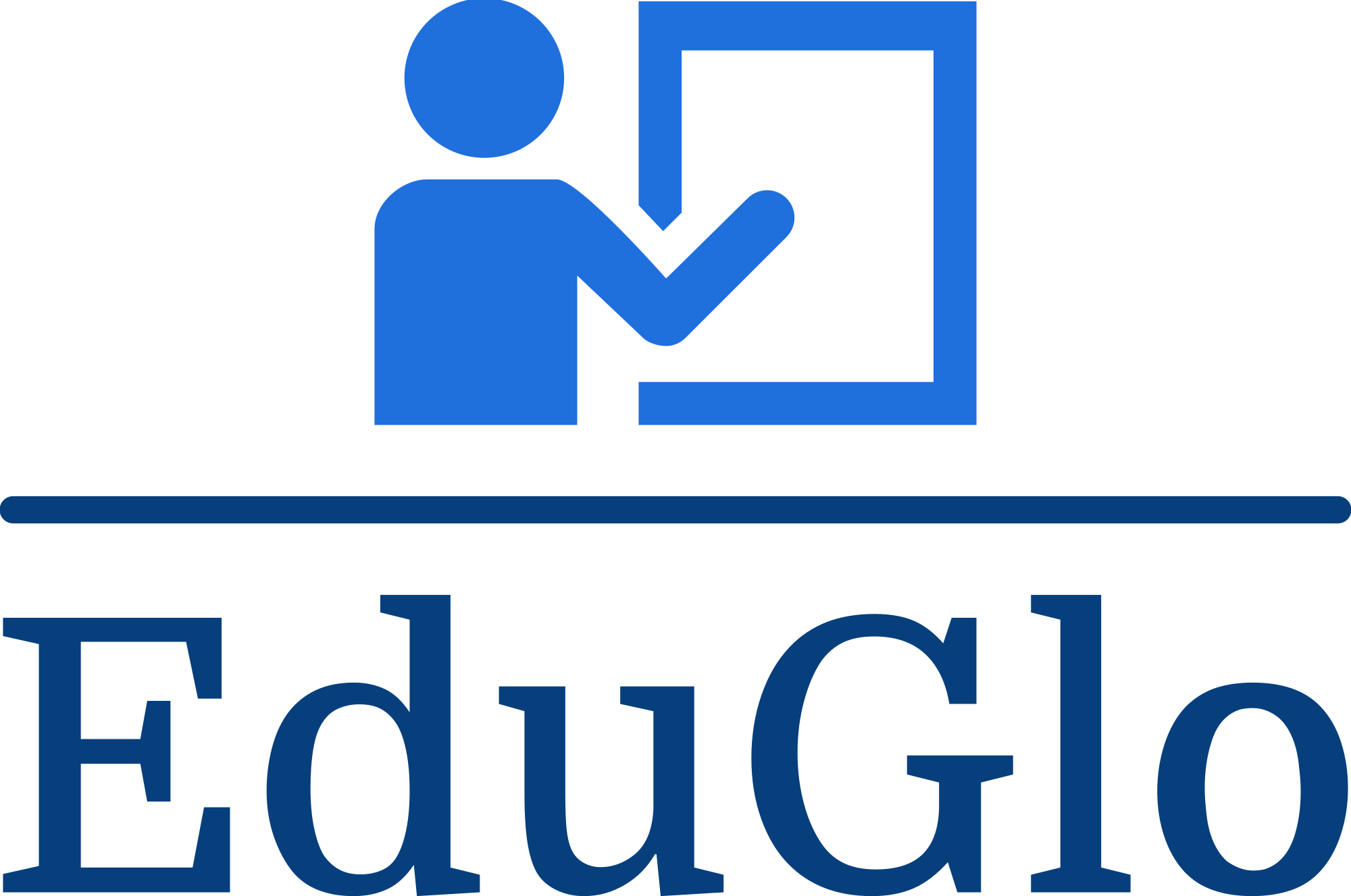
****

**Final Project**

**EduGlo Student Management System Project Report**

Prepared by G. V. Nihara Nethmini

DIIT 01 – 00202392



Diploma in Information Technology

ESOFT Metro Campus

Nittabuwa

1/27/2024

|  |  |
| --- | --- |
| **Final Project – Assessment Sheet** | |
| **Programme** | Diploma in Information Technology |
| **Assessor/Lecturer** | Mr. Chalitha Fernando / Miss Ruwanthika Hettiarachchi |
| **Unit** | Final Project |
| **Project Title** | EduGlo Student Management System |
| **Student’s Name** | G. V. Nihara Nethmini |
| **Student’s Registration No** | 00202392 |
| **Submission Date** |  |
| **Marks Achieved (%)** |  |
| **Assessor/Lecturer Feedback** |  |
|  |  |
|  |  |

# **Acknowledgement**

I would like to express my sincere gratitude to all those who have contributed to the successful completion of my Final Project for the Diploma program. This endeavor, the development of EduGlo, a Student Management System, would not have been possible without the unwavering support and guidance of two key individuals.

I extend my heartfelt gratitude to my supervisors, Mr. Chalitha Fernando and Miss Ruwanthika Hettiarachchi, for their unwavering support, guidance, and encouragement throughout the development of my final project. Their expertise and constructive feedback have played a pivotal role in shaping the direction and outcome of this project.

Furthermore, a special mention goes to the developers' community for their open-source contributions and online forums that proved to be invaluable resources during challenging moments in the development process.

Finally, I want to express my gratitude to my family for their unwavering support and understanding during the intense periods of project development.

This project has been a journey of learning and growth, and I am sincerely thankful to all those who played a role, big or small, in making EduGlo a reality.

Nihara Nethmini

1/27/2024

# **Abstract**

The completion of my Diploma program brings forth the fruition of an innovative project, the EduGlo Student Management System. Developed using C# and Visual Studio 2022 with the .NET Framework, this offline Windows Forms Application facilitates efficient administration of educational entities. With the exclusive user role being the administrator, the system encompasses five essential forms: Admin Login, Dashboard, Student Details, Lecturer Details, and Course Details. The utilization of MS SQL Server Management Studio for database development ensures a robust and secure foundation for data management. The system's fundamental functionalities include seamless admin login and logout capabilities, alongside comprehensive CRUD operations for managing Student Details, Lecturer Details, and Course Details. In terms of database architecture, the project incorporates three key tables: Students, Lecturers, and Courses. The students table maintains a Many-to-One relationship with the Courses table, while the relationship between Lecturers and Courses is established as One-to-One. This relational structure enhances the system's flexibility and scalability, aligning it with the dynamic needs of educational institutions. A distinctive feature of EduGlo is its offline nature, catering to the specific requirements of Windows Forms Applications. The database operates locally, ensuring data integrity and accessibility without dependence on external network connectivity. The EduGlo Student Management System stands as a testament to the successful amalgamation of technology and education. This project not only demonstrates proficiency in software development but also addresses the practical needs of educational administrators. The inherent functionalities and relational database design make EduGlo an asset for educational institutions seeking an efficient and user-friendly solution.

**Keywords:** Student Management System, C#, Windows Forms Application, Database Management, Offline System

# **Declaration of Originality**

I, Nihara Nethmini, solemnly declare that the work presented in this project, entitled "EduGlo Student Management System," is entirely my own. I certify that I have diligently developed and executed this project as part of my academic studies and research.

Every external reference, whether from scholarly articles, documentation, or other sources, is duly acknowledged in the appropriate sections of this document. Contributions from collaborators or third-party sources are explicitly recognized and attributed.

I accept full responsibility for the accuracy and validity of all aspects of this project, including the design, coding, and methodologies employed. The work presented herein is a result of my independent efforts, and any assistance received is appropriately acknowledged.

With a profound understanding of the significance of academic and professional integrity, I affirm that this work is free from illegal help, collusion, or any unethical actions. I am fully aware of the potential consequences associated with deviating from the principles of originality and academic honesty, such as academic sanctions and harm to credibility.

By submitting this Declaration of Originality, I commit to upholding the highest standards of honesty in academic and professional settings. I acknowledge the importance of maintaining the integrity of my work and understand the gravity of any departure from the norms of originality. This declaration attests to my unwavering dedication to academic honesty and the ethical conduct expected in the scholarly community.

# **Table of Contents**

[**Acknowledgement** 3](#_Toc157365174)

[**Abstract** 4](#_Toc157365175)

[**Declaration of Originality** 5](#_Toc157365176)

[**Table of Contents** 6](#_Toc157365177)

[**List of Figures** 7](#_Toc157365178)

[**Chapter 1 – Introduction** 8](#_Toc157365179)

[**1.1 Problem Definition** 8](#_Toc157365180)

[**1.2 Project Scope** 9](#_Toc157365181)

[**1.3 Project Objectives** 9](#_Toc157365182)

[**Chapter 2 – Requirements** 10](#_Toc157365183)

[**2.1 Functional Requirements** 10](#_Toc157365184)

[**2.2 Non -Functional Requirements** 11](#_Toc157365185)

[**Chapter 3 – Development Tools and Technologies** 12](#_Toc157365186)

[**3.1 Development Methodology** 12](#_Toc157365187)

[**3.2 Programming Languages and Tools** 14](#_Toc157365188)

[**Chapter 4 – Diagrams** 19](#_Toc157365189)

[**4.1 Use Case Diagram** 19](#_Toc157365190)

[**4.2 Class Diagram** 20](#_Toc157365191)

[**4.3 ER Diagram** 21](#_Toc157365192)

[**Chapter 5 – User Interface** 22](#_Toc157365193)

[**5.1 Admin Login** 22](#_Toc157365194)

[**5.2 Dashboard** 23](#_Toc157365195)

[**5.3 Student Details** 24](#_Toc157365196)

[**5.4 Lecturer Details** 25](#_Toc157365197)

[**5.5 Course Details** 27](#_Toc157365198)

[**Chapter 6 – Discussion** 28](#_Toc157365199)

[**6.1 Future Implementation** 28](#_Toc157365200)

[**6.2 Conclusion** 29](#_Toc157365201)

[**References** 30](#_Toc157365202)

[**Appendix** 31](#_Toc157365203)

# **List of Figures**

[Figure 3.1‑01 Agile vs. Waterfall 12](#_Toc157340814)

[Figure 3.1‑02 Scrum Process 13](#_Toc157340815)

[Figure 4.1‑01 Use Case Diagram 19](#_Toc157340816)

[Figure 4.2‑02 Class Diagram 20](#_Toc157340817)

[Figure 4.3-03 ER Diagram 21](#_Toc157340818)

[Figure 5.1-01 Admin Login: Interface 22](#_Toc157340819)

[Figure 5.2-02 Dashboard: Interface 23](#_Toc157340820)

[Figure 5.3-03 Student Details: Interface 24](#_Toc157340821)

[Figure 5.4-04 Lecturer Details: Interface 25](#_Toc157340822)

[Figure 5.5-05 Course Details: Interface 27](#_Toc157340823)

# **Chapter 1 – Introduction**

In the realm of contemporary education, the seamless management of student information, lecturer details, and course particulars is integral to the efficient functioning of academic institutions. In response to this imperative, the "EduGlo Student Management System" emerges as a comprehensive solution, tailored for the streamlined administration of educational entities. This project represents the culmination of my diligent efforts in the field of software development, undertaken as the final endeavor in my Diploma program. Leveraging C# and Visual Studio 2022 with the .NET Framework, EduGlo is a Windows Forms Application designed exclusively for administrators, offering essential functionalities through five distinct forms: Admin Login, Dashboard, Student Details, Lecturer Details, and Course Details. The incorporation of MS SQL Server Management Studio ensures a robust foundation for database management, facilitating secure and efficient data handling. This introduction sets the stage for an in-depth exploration of EduGlo, underscoring its significance in addressing the evolving needs of educational institutions and contributing to the intersection of technology and education.

## **1.1 Problem Definition**

Traditional methods of managing student information, lecturer details, and course data within educational institutions often suffer from inefficiencies and manual processes. The absence of a dedicated system can lead to challenges in data organization, accessibility, and overall administrative effectiveness. Recognizing these issues, the need for a tailored solution arises. The "EduGlo Student Management System" seeks to address these shortcomings by providing a user-friendly, offline application that streamlines administrative tasks, enhances data management, and contributes to the overall efficiency of educational administration. This project aims to bridge the gap between traditional practices and contemporary technological solutions, offering a reliable platform for educational institutions to manage their resources seamlessly.

## **1.2 Project Scope**

The scope of the "EduGlo Student Management System" is centered around the development of a robust offline Windows Forms Application, exclusively designed for administrators. The system will encompass essential functionalities, including Admin Login, Dashboard, Student Details, Lecturer Details, and Course Details forms. Leveraging C# and Visual Studio 2022 with the .NET Framework, the project aims to facilitate efficient data management through MS SQL Server Management Studio. Key features within the project scope include seamless admin login and logout capabilities, along with comprehensive CRUD operations for managing student, lecturer, and course details. The database architecture will consist of three primary tables – Students, Lecturers, and Courses – establishing crucial relationships to enhance flexibility and scalability. Notably, the application will operate offline, addressing the specific requirements of Windows Forms Applications and ensuring data integrity and accessibility locally. The scope of this project is tailored to meet the practical needs of educational administrators, providing a user-friendly solution for effective student and course management.

## **1.3 Project Objectives**

* Develop an offline Windows Forms Application, "EduGlo," using C# and Visual Studio 2022, aimed exclusively at administrators.
* Implement key functionalities, including Admin Login, Dashboard, Student Details, Lecturer Details, and Course Details, with a focus on seamless CRUD operations.

# **Chapter 2 – Requirements**

## **2.1 Functional Requirements**

The functional requirements of the "EduGlo Student Management System" delineate the essential capabilities and features vital for the system's intended purpose. Key functionalities include:

1. **User Authentication and Authorization**

* Admin can securely login using a user-friendly Admin Login mechanism, requiring a valid username and password.

1. **Dashboard Overview**

* A comprehensive dashboard provides admins with key metrics and summaries related to student enrollment, lecturer assignments, and course offerings.
* Quick navigation buttons allow admins to access detailed information in the Student Details, Lecturer Details, and Course Details forms effortlessly.

1. **Student Details Management**

* Admin can perform CRUD operations for managing student information, including creating, reading, updating, and deleting student records.

1. **Lecturer Details Management**

* CRUD operations empower admins to create, read, update, and delete lecturer information as needed.

1. **Course Details Management**

* Admin can execute CRUD operations for course information, encompassing creation, reading, modification, and deletion.

## **2.2 Non -Functional Requirements**

Non-functional requirements focus on aspects such as performance, security, and usability, enhancing the overall quality and effectiveness of the EduGlo system:

1. **Performance**

* The system ensures responsive performance, guaranteeing swift loading times for forms and dashboards.
* Efficient handling of concurrent user interactions is a priority, maintaining data accuracy and integrity.

1. **Security**

* Access control measures restrict unauthorized users from manipulating or accessing confidential data.

1. **Usability**

* The user interface is designed for intuitiveness, featuring clear navigation and a minimalistic design for ease of use.
* Adequate tooltips and guidance features assist admins in utilizing the system seamlessly.

1. **Reliability**

* The system maintains data consistency and reliability even during network unavailability.

1. **Scalability**

* The architecture accommodates potential expansions, allowing for the addition of more students, lecturers, and courses without compromising performance.

# **Chapter 3 – Development Tools and Technologies**

## **3.1 Development Methodology**

The chosen development methodology for this project is the agile approach, an iterative and collaborative strategy aimed at delivering functional and valuable software through short development cycles. A prominent framework within Agile development is the Scrum process, widely recognized for its effectiveness. This methodology involves the formation of small, cross-functional teams collaboratively working to deliver increments of functionality within defined timeframes known as sprints, typically lasting four weeks each.

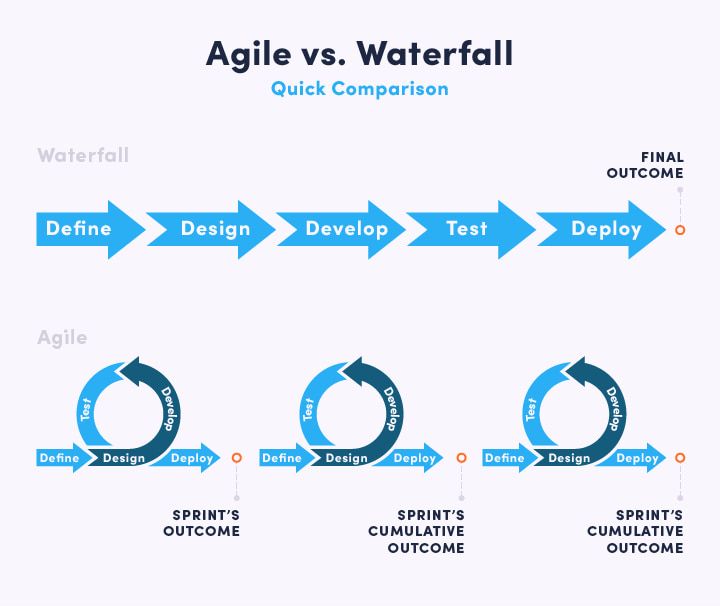


Figure 3.1‑ Agile vs. Waterfall

Agile development excels in flexibility, allowing teams to promptly respond to changes in customer needs and evolving requirements. In contrast to methodologies like Waterfall, which prioritize detailed planning and sequential execution, Agile development emphasizes collaboration, communication, and iterative delivery. This approach proves crucial in today's dynamic business environment, where customer needs and requirements undergo constant evolution. (Pawlicka, 2023)

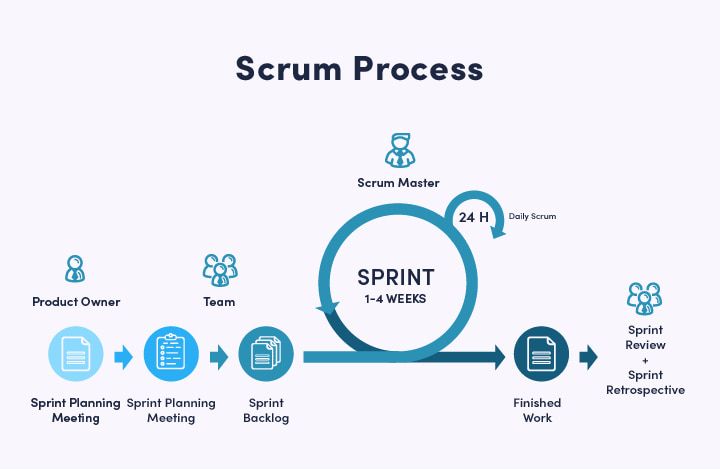


Figure 3.1‑ Scrum Process

While often associated with team-based projects, Agile development proves highly advantageous for individual projects as well. Its adaptability extends to projects of any size, enabling developers to break down tasks into manageable components, prioritize them, and address them in short sprints. This approach facilitates swift responses to changing requirements, minimizes risks, and accelerates the delivery of high-quality software.

Agile development emerges as a fitting strategy for projects seeking to efficiently deliver high-quality software while embracing the principles of flexibility, collaboration, and continuous improvement. The development of this project is anchored in the Agile methodology, ensuring a dynamic and responsive approach to meet evolving project requirements.

## **3.2 Programming Languages and Tools**

**C# Programming Language**

C# is a powerful and versatile programming language developed by Microsoft within the .NET framework. It combines the robustness of C++ with the simplicity of Visual Basic, making it an ideal choice for developing a wide range of applications, from desktop software to web applications and games.

Key features of C# include:

* **Object-Oriented Programming (OOP):** C# is built around the principles of object-oriented programming, allowing developers to create modular, reusable code through classes and objects. This facilitates code organization, maintenance, and scalability.
* **Syntax Simplicity:** C# boasts an intuitive and straightforward syntax, making it accessible to developers of various skill levels. Its syntax is similar to that of other C-style languages like C++ and Java, simplifying the learning curve for those familiar with these languages.
* **Memory Management:** C# employs automatic memory management through garbage collection, relieving developers of the burden of manual memory allocation and deallocation. This enhances code reliability and reduces the risk of memory-related errors.
* **Platform Independence:** While initially developed for Windows-based systems, C# has evolved to become platform-independent through the .NET framework. With the introduction of .NET Core and later .NET 5, C# applications can now run on various operating systems, including Windows, macOS, and Linux.
* **Rich Standard Library:** C# benefits from a comprehensive standard library that provides a wide range of pre-built classes and functions for common programming tasks. This library simplifies development by offering ready-to-use solutions for tasks such as file I/O, networking, and database access.
* **Integration with Visual Studio:** C# development is seamlessly integrated with Microsoft’s Visual Studio IDE, offering powerful tools for code editing, debugging, and project management. Visual Studio enhances developer productivity through features like IntelliSense, code refactoring, and built-in version control integration.
* **Strong Community Support:** C# enjoys a vibrant and active community of developers, contributing to its extensive documentation, tutorials, and third-party libraries. This community-driven ecosystem fosters collaboration and knowledge sharing, facilitating the resolution of coding challenges and promoting best practices.

**Visual Studio**

Visual Studio is an integrated development environment (IDE) developed by Microsoft, providing a comprehensive set of tools for software development. Widely regarded as one of the most robust and feature-rich IDEs, Visual Studio supports multiple programming languages, including C#, making it a preferred choice for developers across various domains.

Key features of Visual Studio include:

* **Code Editing and Debugging:** Visual Studio offers a powerful code editor with features like IntelliSense, which provides intelligent code completion suggestions, and robust debugging tools that streamline the development and debugging process.
* **Integrated Testing Tools:** The IDE includes built-in testing tools for unit testing and test case management, enabling developers to ensure the reliability and correctness of their code.
* **Version Control Integration:** Visual Studio seamlessly integrates with popular version control systems such as Git and Team Foundation Server (TFS), facilitating collaborative development and code management.
* **Extensibility:** Visual Studio supports a vast ecosystem of extensions and plugins, allowing developers to customize their development environment based on their specific needs and preferences.
* **Cross-Platform Development:** With the advent of .NET Core and later .NET 5, Visual Studio has expanded its support for cross-platform development, enabling developers to build applications that run on Windows, macOS, and Linux.

**Windows Forms**

Windows Forms is a graphical user interface (GUI) framework within the Microsoft .NET framework, designed for creating desktop applications for Windows operating systems. It provides a set of classes and libraries that simplify the development of interactive and visually appealing desktop applications.

Key features of Windows Forms include:

* **Rapid Application Development (RAD):** Windows Forms facilitates rapid application development by offering a drag-and-drop interface for designing user interfaces. This speeds up the development process and allows developers to focus on application logic.
* **Event-Driven Programming Model:** Windows Forms utilizes an event-driven programming model, allowing developers to respond to user actions and system events easily. This results in responsive and interactive user interfaces.
* **Rich Control Library:** Windows Forms includes a rich set of pre-built controls, such as buttons, textboxes, and datagrids, making it easy for developers to create a variety of UI elements without starting from scratch.
* **Integration with Visual Studio:** Windows Forms development is seamlessly integrated with Visual Studio, providing a cohesive environment for designing, coding, and debugging desktop applications. The combination of Visual Studio and Windows Forms streamlines the development workflow.
* **Data Binding and Connectivity:** Windows Forms supports data binding, simplifying the connection between the application's user interface and data sources. This enhances the efficiency of working with databases and other data-centric components.

**Version Controller - Git**

Git, a pivotal component in modern software development, serves as a robust version control system, facilitating effective collaboration and codebase management. It is indispensable for tracking code changes, enabling seamless collaboration among developers, and managing different iterations of the codebase.

Key Features:

* **Committing:** Git allows developers to record changes to the codebase through commits, providing a detailed and organized history of modifications. This aids in tracing the evolution of the code and understanding the rationale behind each change.
* **Branching:** The branching feature in Git enables developers to create isolated environments for new features or bug fixes. This ensures that changes can be developed independently, promoting parallel work and reducing conflicts.
* **Merging:** Git simplifies the process of integrating changes from one branch into another through merging. This ensures a smooth and cohesive codebase, consolidating developments from various contributors.
* **Staging:** The staging area in Git allows developers to selectively include changes in a commit, providing granularity in version control. This ensures that only intended modifications are included in a particular commit.
* **Conflict Resolution:** In collaborative projects, conflicts may arise when multiple contributors modify the same code. Git facilitates efficient conflict resolution, ensuring that changes are harmoniously integrated into the codebase.
* **Distribution:** Git's distributed nature enables developers to work independently and later synchronize their changes. This decentralization enhances flexibility, allowing contributors to work offline and share their contributions seamlessly.

Role in Application Development:

In the context of this individual project, Git played a pivotal role in streamlining the development process. Its features significantly contributed to the project's success:

* **Codebase Stability:** Using Git commits, the development process maintained a stable and well-documented codebase. Each commit represented a logical unit of change, aiding in the identification of specific modifications and their impact.
* **Feature Isolation:** Branching in Git allowed for the isolation of new features or improvements, preventing interference with the main codebase until changes were thoroughly tested and ready for integration.
* **Version Tracking:** Git's version tracking capabilities ensured that every iteration of the application was meticulously recorded. This proved invaluable for troubleshooting, auditing changes, and understanding the evolution of the software.
* **Efficient Conflict Management:** Git's conflict resolution mechanisms facilitated the seamless integration of code modifications. This streamlined the development process, minimizing disruptions even in the absence of collaborative efforts.

While Git's collaborative features were not extensively utilized in this individual project, its other functionalities proved instrumental in maintaining code stability, isolating features, tracking versions, and efficiently managing conflicts. Git's adaptability allows it to enhance the development workflow, whether in collaborative or individual settings.

# **Chapter 4 – Diagrams**

## **4.1 Use Case Diagram**

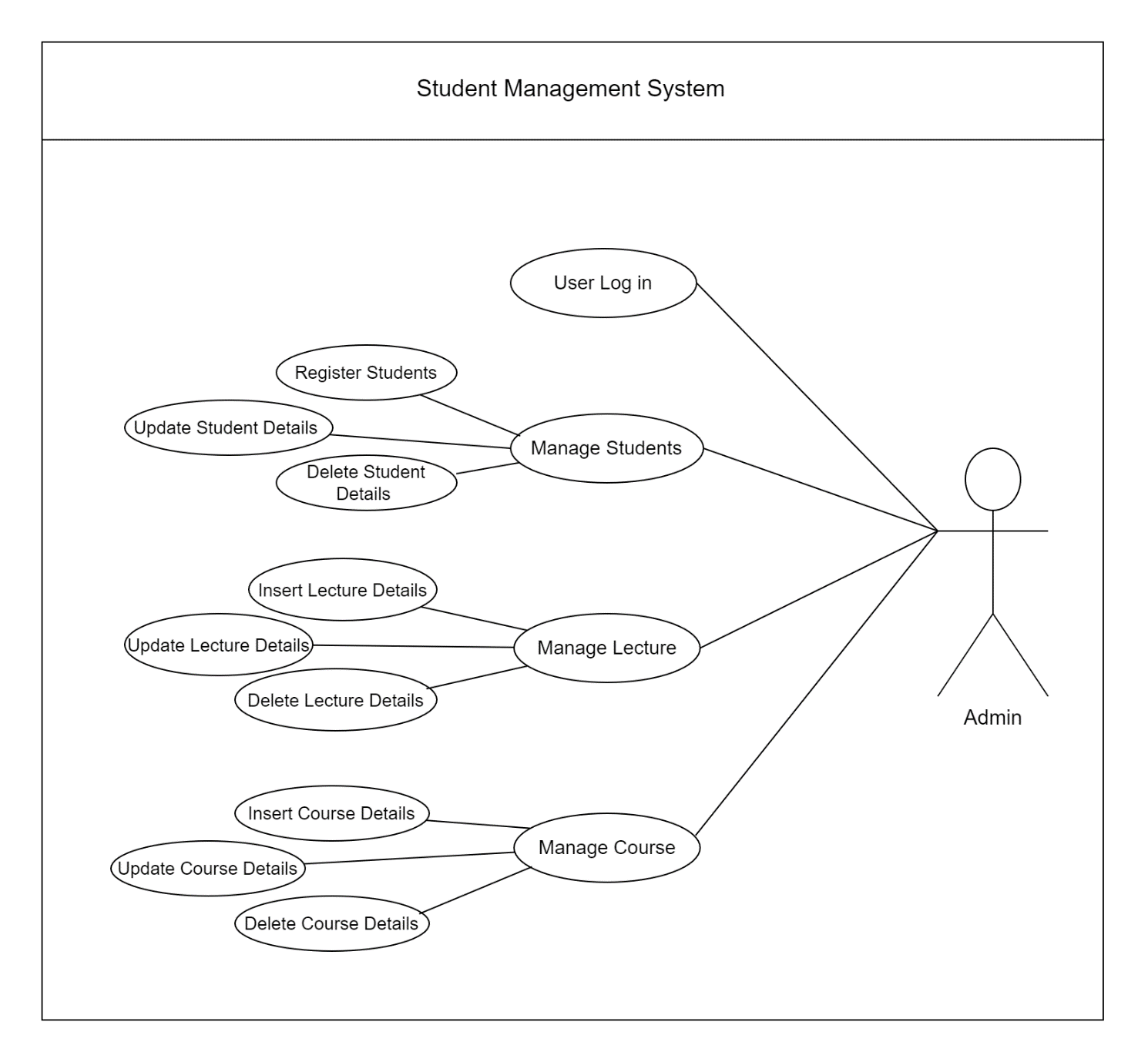


Figure 4.1‑ Use Case Diagram

## **4.2 Class Diagram**

A diagram of a computer

Description automatically generated with medium confidence

Figure 4.2‑ Class Diagram

## **4.3 ER Diagram**

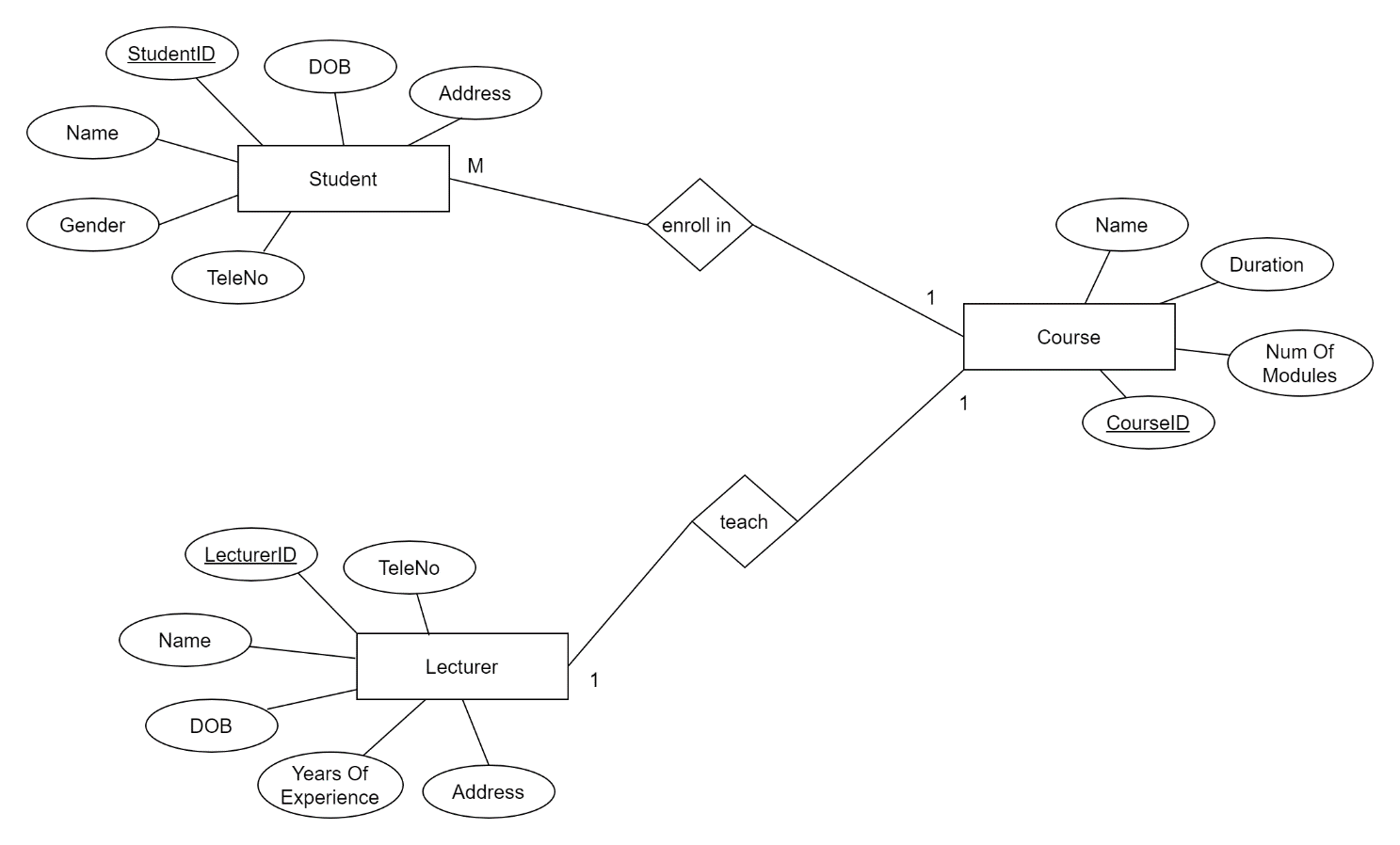


Figure 4.3-0 ER Diagram

# **Chapter 5 – User Interface**

## **5.1 Admin Login**

This Admin Login form sets the foundation for secure access to the application, incorporating intuitive elements and robust error handling to enhance user interaction.

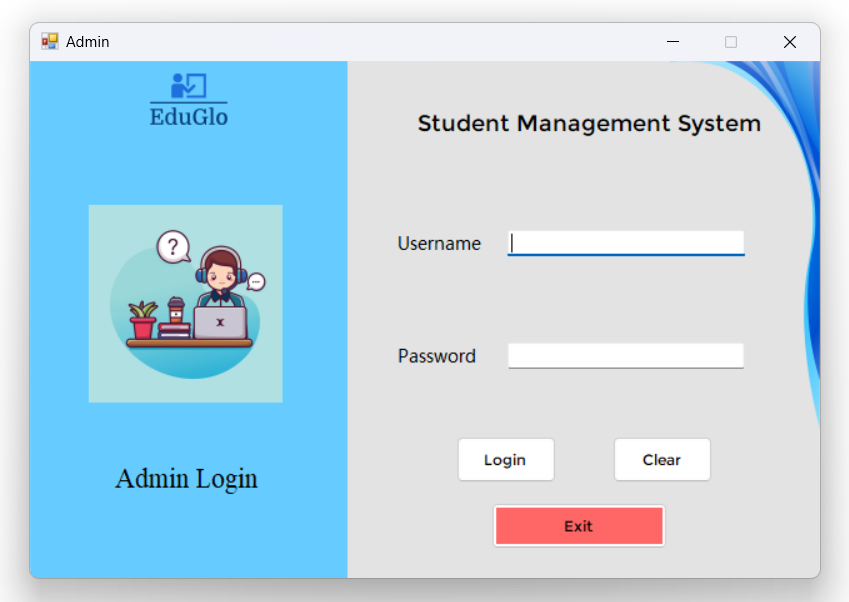


Figure 5.1- Admin Login: Interface

**Initialization:**

* The Admin Login form is the initial interface loaded upon application startup.

**User Credentials:**

* Only one user, the administrator (admin), can access the form.
* Admin provides a correct username and password for authentication.

**Interaction Elements:**

* Text Boxes: Allows input for username and password.
* Clear Button: A single button clears both username and password fields, offering a convenient reset option.
* Exit Button: Provides an option to close the application.

**Error Handling:**

* Exception and error handling are implemented through message boxes.
* Ensures a user-friendly experience by notifying users of any incorrect inputs or issues during the login process.

## **5.2 Dashboard**

This Dashboard Form acts as a command center, providing easy access to essential functionalities and ensuring seamless navigation between different sections of the application.

A screenshot of a computer

Description automatically generated

Figure 5.2- Dashboard: Interface

**Redirection:**

* Upon successful login, the application redirects to the Dashboard Form, serving as the central hub for navigation.

**Functional Buttons:**

* Student View: Button enabling access to the Student Details management form.
* Lecturer View: Button facilitating access to the Lecturer Details management form.
* Course View: Button directing to the Course Details management form.
* Logout: Button to log out from the system, redirecting the admin back to the Login page.

## **5.3 Student Details**

This Student Details Form offers a comprehensive interface for managing student information, providing functionalities for searching, displaying, inserting, updating, and deleting records with ease.

A screenshot of a computer

Description automatically generated

Figure 5.3- Student Details: Interface

**Form Elements:**

* Student ID: Text field for entering the unique identifier of the student.
* Name: Text field for inputting the student's name.
* Date of Birth: Text field, internally converted to yyyy-mm-dd format.
* Gender: ComboBox for selecting the student's gender.
* Address: Text field for entering the student's address.
* TeleNo: Text field for inputting the student's telephone number.
* Course ID: ComboBox connected to the Course Table's Course\_ID in the database.

**Buttons:**

* Search: Allows the admin to search for student details by entering the student ID, with results displayed on the DataGridView.
* Show All: Displays all student details in the DataGridView and refreshes the view.
* Insert: Adds a new student record based on the entered details.
* Update: Modifies the selected student's details.
* Delete: Deletes the selected student record.
* Clear: Resets all fields for a new entry or action.
* Home: Navigates back to the Dashboard Form.

## **5.4 Lecturer Details**

This Lecturer Details Form provides a user-friendly interface for managing lecturer information, offering functionalities for searching, displaying, inserting, updating, and deleting records seamlessly.

A screenshot of a computer

Description automatically generated

Figure 5.4- Lecturer Details: Interface

**Form Elements:**

* Lecturer ID: Text field for entering the unique identifier of the lecturer.
* Name: Text field for inputting the lecturer's name.
* Date of Birth: Text field, internally converted to yyyy-mm-dd format.
* Address: Text field for entering the lecturer's address.
* TeleNo: Text field for inputting the lecturer's telephone number.
* Years of Experience: Text field for specifying the lecturer's years of experience.
* Course ID: ComboBox connected to the Course Table's Course\_ID in the database.

**Buttons:**

* Search: Allows the admin to search for lecturer details by entering the lecturer ID, with results displayed on the DataGridView.
* Show All: Displays all lecturer details in the DataGridView and refreshes the view.
* Insert: Adds a new lecturer record based on the entered details.
* Update: Modifies the selected lecturer's details.
* Delete: Deletes the selected lecturer record.
* Clear: Resets all fields for a new entry or action.
* Home: Navigates back to the Dashboard Form.

## **5.5 Course Details**

This Course Details Form facilitates efficient management of course information, offering functionalities for searching, displaying, inserting, updating, and deleting records seamlessly.

A screenshot of a computer

Description automatically generated

Figure 5.5- Course Details: Interface

**Form Elements:**

* Course ID: Text field for entering the unique identifier of the course.
* Name: Text field for inputting the course name.
* Duration: Text field for specifying the duration of the course.
* Num of Modules: Text field indicating the number of modules in the course.

**Buttons:**

* Search: Allows the admin to search for course details by entering the course ID, with results displayed on the DataGridView.
* Show All: Displays all course details in the DataGridView and refreshes the view.
* Insert: Adds a new course record based on the entered details.
* Update: Modifies the selected course's details.
* Delete: Deletes the selected course record.
* Clear: Resets all fields for a new entry or action.
* Home: Navigates back to the Dashboard Form.

# **Chapter 6 – Discussion**

## **6.1 Future Implementation**

In envisioning the future development of this application, several enhancements and expansions can be considered to further enrich its functionality and user experience.

* **Additional Forms for Academic Management:** Integrate modules for recording marks, grading, and performance analytics to comprehensively manage students' academic data.
* **Financial Management Features:** Expand the application with forms for student payments and fee tracking, enhancing administrative efficiency.
* **User Roles Expansion:** Introduce diverse user roles, allowing students access to specific functionalities such as viewing academic results.
* **Online Database Implementation:** Transition from a local to an online database for enhanced accessibility and real-time updates.
* **Reporting and Analytics Advancements:** Develop advanced features for detailed reporting, offering insights into student performance and overall system metrics.
* **Security Enhancement Measures:** Strengthen the application's security with multi-factor authentication and encryption for safeguarding sensitive data.
* **Data Backup and Recovery Protocols:** Implement robust protocols to ensure data resilience against unforeseen loss.

## **6.2 Conclusion**

In conclusion, the development of the EduGlo Student Management System marks a significant achievement in addressing the administrative complexities of educational institutions. This project, realized through the utilization of C# and Visual Studio, demonstrates a commitment to leveraging modern technologies for efficient and streamlined processes. The system's core functionalities, including Admin Login, Dashboard, Student Details, Lecturer Details, and Course Details, lay a solid foundation for comprehensive student and other management. The incorporation of Git as the version control system ensures codebase stability, even in the context of an individual project, emphasizing the importance of systematic version tracking. As the EduGlo Student Management System progresses, it exemplifies the synergy between technology and education, showcasing a commitment to innovation and efficiency. The successful execution of this project underscores the potential for ongoing growth and refinement, positioning the system as an asset for educational institutions seeking modern and effective management solutions.

# **References**

*Creating a database with tables and relationships (MS SQL)* (2016) *ProgrammingKnowledge*. Available at: <https://www.youtube.com/watch?v=H7uGGg1BQ2Y> (Accessed: 23 January 2024).

*SQL Server database creation, tables and relationships | SQL server management studio* (2020) *TECHINSPEC*. Available at: <https://www.youtube.com/watch?v=qBMo-AORiFQ> (Accessed: 23 January 2024).

*Stackoverflow*, (1956) *How to show a custom error or warning message box in .net winforms?* Available at: <https://stackoverflow.com/questions/2109441/how-to-show-a-custom-error-or-warning-message-box-in-net-winforms> (Accessed: 24 January 2024).

Pawlicka, A. (2023) *Agile Software Development Process - Guide*. Available at: <https://selleo.com/blog/agile-software-development-process-everything-you-need-to-know> (Accessed: 26 January 2024).

# **Appendix**