ASSIGNMENT 1 FRONT SHEET

Qualification	**BTEC Level 5 HND Diploma in Computing**		
Unit number and title	Unit 30: Application Development		
Submission date	05/03/2024	**Date Received 1st submission**	
Re-submission Date		**Date Received 2nd submission**	
Student Name	Bui Nguyen Ngoc Han	**Student ID**	BH00150
Class	IT0501	**Assessor name**	Nguyen Thanh Trieu

Student declaration

I certify that the assignment submission is entirely my own work and I fully understand the consequences of plagiarism. I understand that making a false declaration is a form of malpractice.

Student's signature	Bui Nguyen Ngoc Han

Grading grid

P1	P2	P3	M1	M2	D1

❒ Summative Feedback:		❒ Resubmission Feedback:
Grade:	Assessor Signature:	Date:
Lecturer Signature:		

Table of Contents

I. INTRODUCTION 6

II. CONTENTS 7

P3 Research the use of software development tools and techniques and identify any that have been selected for the development of this application 7

1. UML 7

1.1. UML definition 7

1.2. Some popular UML Diagrams 8

1.2.1. Class diagram 8

1.2.2. Use case diagram 9

1.2.3. Activity diagram 10

1.2.4. Component diagram 12

1.3. Use the UML tool 13

1.3.1. Diagrams.net 13

1.3.2. Lucidchart 14

1.3.3. Gleek.io 16

2. Chosen design tools 17

3. Development Tools and Techniques 17

3.1. C# 17

3.2. ASP.NET Core MVC 18

3.3. Visual Studio 2022 20

3.4. SQL Server 22

M2 Compare the differences between the various software development tools and techniques researched and justify your preferred selection as well as your preferred software development methodology 23

1. Compare and justify the preferred selection of development tools and techniques 23

1.1. Intergrated development environment 23

1.1.1. Visual Studio 2022 23

1.1.2. Sublime Text 25

1.1.3. Visual Studio Code (VS Code) 26

1.1.4. Compare 28

1.2. Programming language 29
1.2.1. C# 29
1.2.2. Java 31
1.2.3. Python 33
1.2.4. Compare 35
1.3. Web framework 36
1.3.1. ASP.NET Core MVC 36
1.3.2. Spring MVC 38
1.3.3. Django 41
1.3.4. Compare 43
1.4. Database management system 44
1.4.1. SQL Server 44
1.4.2. MySQL 46
1.4.3. MongoDB 48
1.4.4. Compare 50
2. Preferred selection justification 51
3. Preferred software development methodology 52
3.1. Agile 52
3.2. Waterfall 54
3.3. V-Model 55
3.4. Preferred software development methodology: Agile 57
III. CONCLUSION 58
IV. REFERENCES 58

List of Tables

Table 1: Compare IDE 28
Table 2: Compare Programming language 35
Table 3: Compare Web framework 43
Table 4: Compare Database management server 51

List of Figures

Figure 1: UML....7
Figure 2: Class diagram....8
Figure 3: Usecase diagram....9
Figure 4: Activity diagram....10
Figure 5: Component diagram....12
Figure 6: Diagrams.net....13
Figure 7: Lucidchart....14
Figure 8: Gleek.io....16
Figure 9: C#....17
Figure 10: ASP.NET Core MVC....18
Figure 11: Visual Studio 2022....20
Figure 12: SQL Server....22
Figure 13: Visual Studio 2022....23
Figure 14: Sublime Text....25
Figure 15: Visual Studio Code....26
Figure 16: C#....29
Figure 17: Java....31
Figure 18: Python....33
Figure 19: ASP.NET Core MVC....36
Figure 20: Spring MVC....38
Figure 21: Django....41
Figure 22: SQL Server....44
Figure 23: MySQL....46
Figure 24: MôngDB....48
Figure 25: Agile....52
Figure 26: Waterfall....54
Figure 27: V-Model....55

I. INTRODUCTION

In the context of increasingly developing technology, FPT Company wishes to build a continuous learning environment throughout the corporation. They wanted to develop a "Training" activity management system for the company's internal training program. This system is used by the human resources department and is used for student account management, instructor management, course catalog management, course management, topic management, topic assignment for courses , assign instructors to topics, assign students to courses.

Our team will have to prepare a software design document that includes the following sections:

- Requirements specification helps explore the problem in terms of a set of user and system requirements, as well as identify any risks involved in successfully completing your application. For this task we will provide an SRS template that completes the task.
- Evaluation section: in this section we will study the use of software development tools and techniques as well as identify any tools and techniques that were chosen to develop this application.
- Design part: in this part we will use the tools selected from the previous step to create a design diagram for our solution based on the requirements specification.

In addition to the SRS documentation we have provided, in this report I will conduct research on the use of software development tools and techniques as well as identify any such tools and techniques was chosen to develop the application for this application. Additionally, I will also compare the differences between the various software development tools and techniques studied and justify the preferred choice and preferred software development method.

II. CONTENTS

P3 Research the use of software development tools and techniques and identify any that have been selected for the development of this application

1. UML

1.1. UML definition

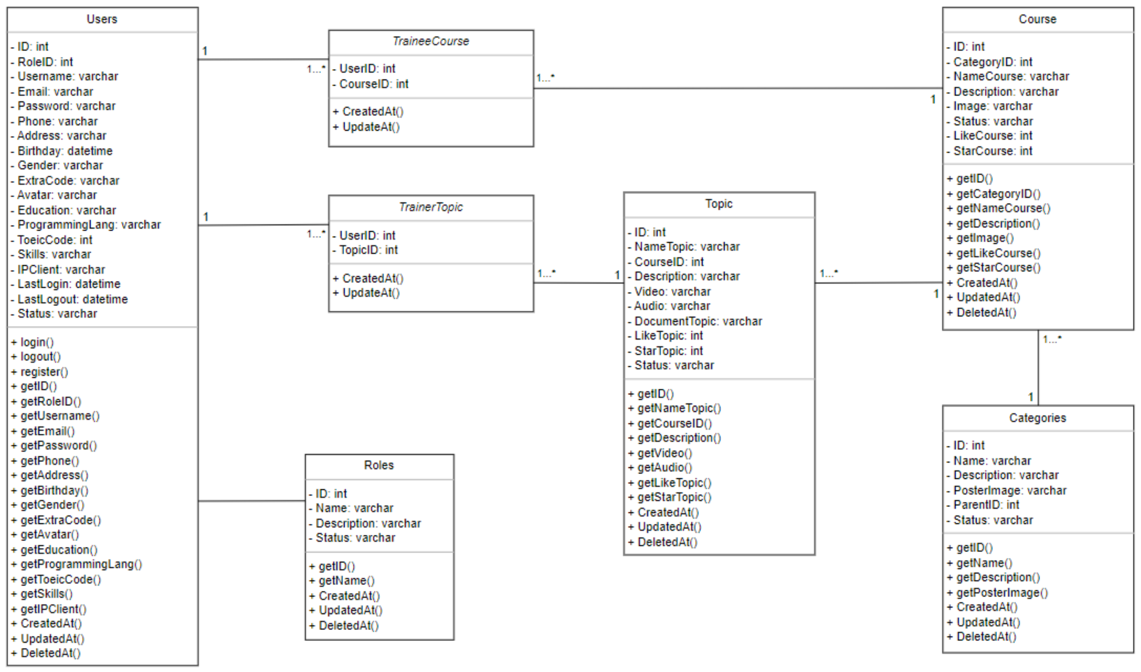


Figure 1: UML

The Unified Modeling Language (UML) is a standardized modeling language that consists of an integrated set of diagrams. It was developed to help system and software developers for specifying, visualizing, constructing, and documenting the artifacts of software systems, as well as for business modeling and other non-software systems. UML uses mostly graphical notations to express the design of software projects.

UML provides a standard notation for many types of diagrams which can be roughly divided into three main groups: behavior diagrams, interaction diagrams, and structure diagrams. The creation of UML was originally motivated by the desire to standardize the disparate notational systems and approaches to software design.

1.2. Some popular UML Diagrams

1.2.1. Class diagram

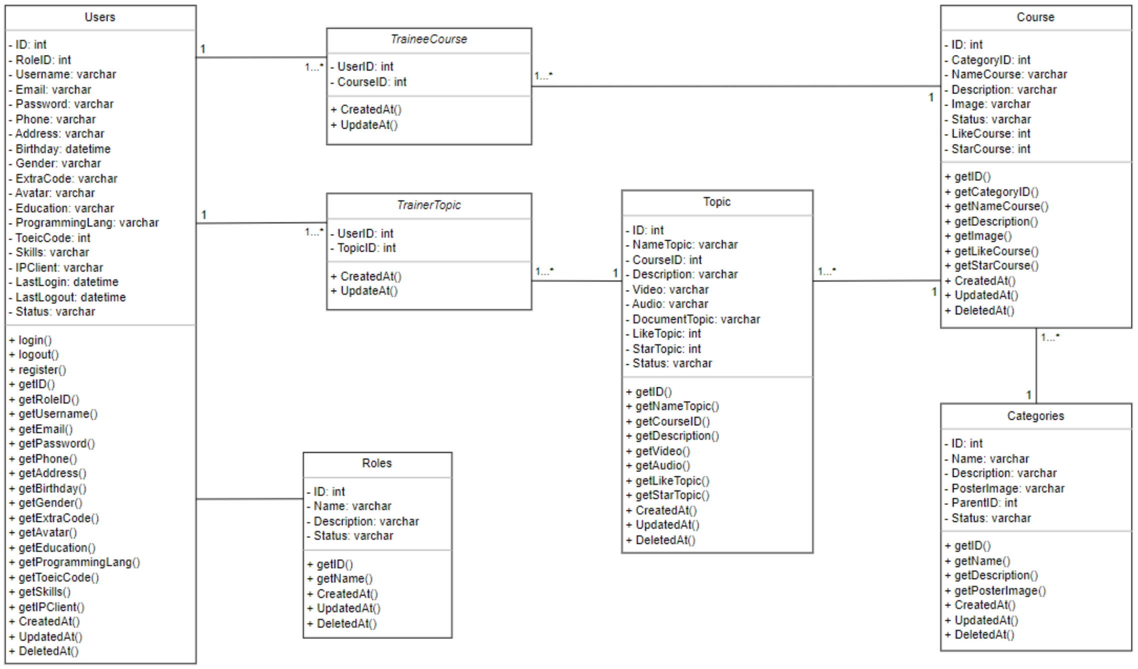


Figure 2: Class diagram

Class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system. It shows the system's classes, their attributes, operations (or methods), and the relationships among objects.

The main components of a class diagram:

- Classes: These are represented with boxes that contain three compartments: the class name, attributes, and operations.
- Attributes: These are shown in the second partition of the class box. They define what objects of the class "know" and represent the state of an object of the class.
- Operations: These are shown in the third partition of the class box. They define what objects of the class "can do" and represent the way in which objects may interact.
- Relationships: A class may be involved in one or more relationships with other classes. There are 4 types of relationships: Inheritance, Association, Composition, Agreegation.

Access Modifier in class diagram is used to specify the scope of access for Attributes and Operations of a class. It includes:

- Private (-): Specifies that objects created from this class can be used by themselves.
- Public (+): All objects can use it.
- Protected (#): Only objects created from this class and classes that inherit from this class can be used.
- Package/Default: Objects created from classes in the same package class can be used.

1.2.2. Use case diagram

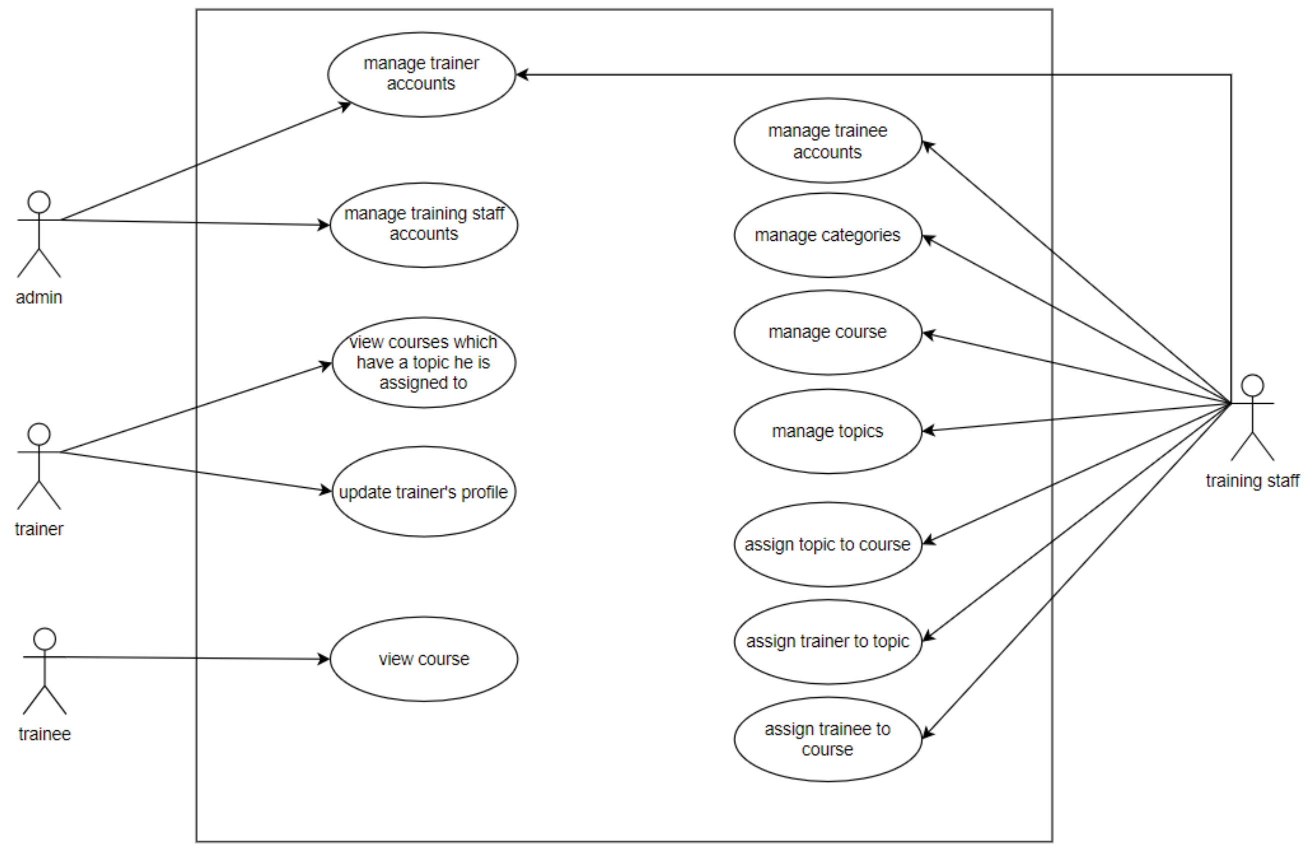


Figure 3: Usecase diagram

Use Case Diagram is a type of diagram used in the Unified Modeling Language (UML) to represent the interactions between a system and its users (known as actors). Use case diagram is the primary form of system/software requirements for a new software program underdeveloped. It's a high-level representation that doesn't go into a lot of detail, but instead provides an overview of the relationship between use cases, actors, and systems.

Here are the main components of a Use Case Diagram:

- Actors: These are the users that interact with a system. An actor can be a person, an organization, or an outside system that interacts with your application or system.
- System: This represents a specific sequence of actions and interactions between actors and the system.
- Goals: These are the end results of most use cases.

Use Case Diagrams are typically developed in the early stage of development and are used for representing the goals of system-user interactions, defining and organizing functional requirements in a system, specifying the context and requirements of a system, and modeling the basic flow of events in a use case. They are an effective technique for communicating system behavior in the user's terms by specifying all externally visible system behavior.

1.2.3. Activity diagram

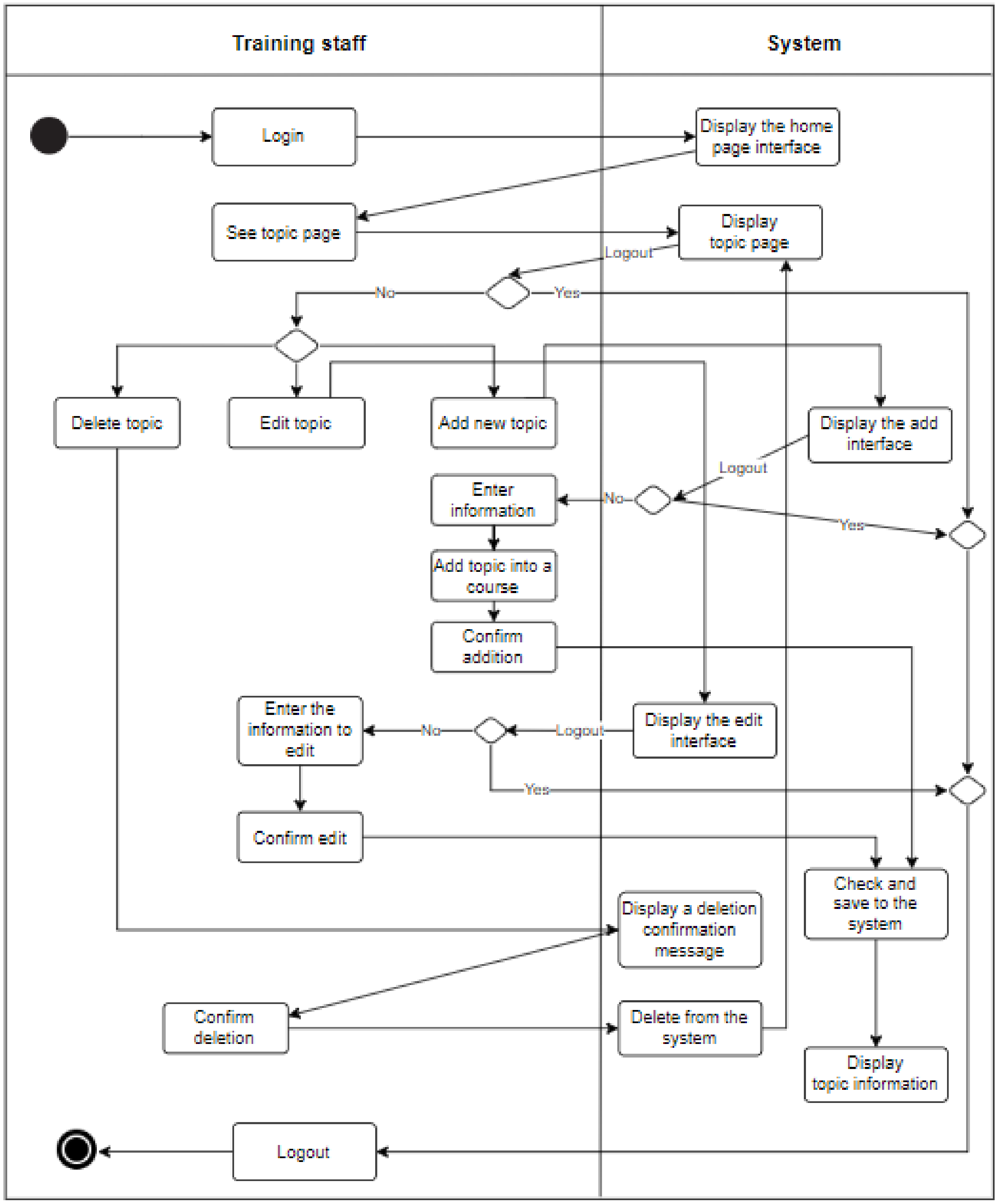


Figure 4: Activity diagram

An Activity Diagram is a behavioral diagram in UML (Unified Modeling Language) that describes the dynamic aspects of a system. Activity diagram is another important behavioral diagram in UML diagram to describe dynamic aspects of the system. It's essentially an advanced version of a flowchart, modeling the flow from one activity to another.

Here are the main components of an Activity Diagram:

- Activities: These are the operations of a system.
- Transitions: These are the relationships between activities.
- Decision Points: These are points in the process where the outcome of a decision dictates the next step.
- Merge Points: These are points in the process where divergent paths re-converge.
- Start and End Points: Every activity diagram has a start point and an end point.

Activity Diagrams are used to describe how activities are coordinated to provide a service, which can be at different levels of abstraction. They are particularly useful for modeling complex workflows in operations on objects, identifying candidate use cases through the examination of business workflows, identifying pre- and post-conditions for use cases, and modeling workflows between/within use cases. They can also be used to model in detail complex activities in a high-level activity diagram.

1.2.4. Component diagram

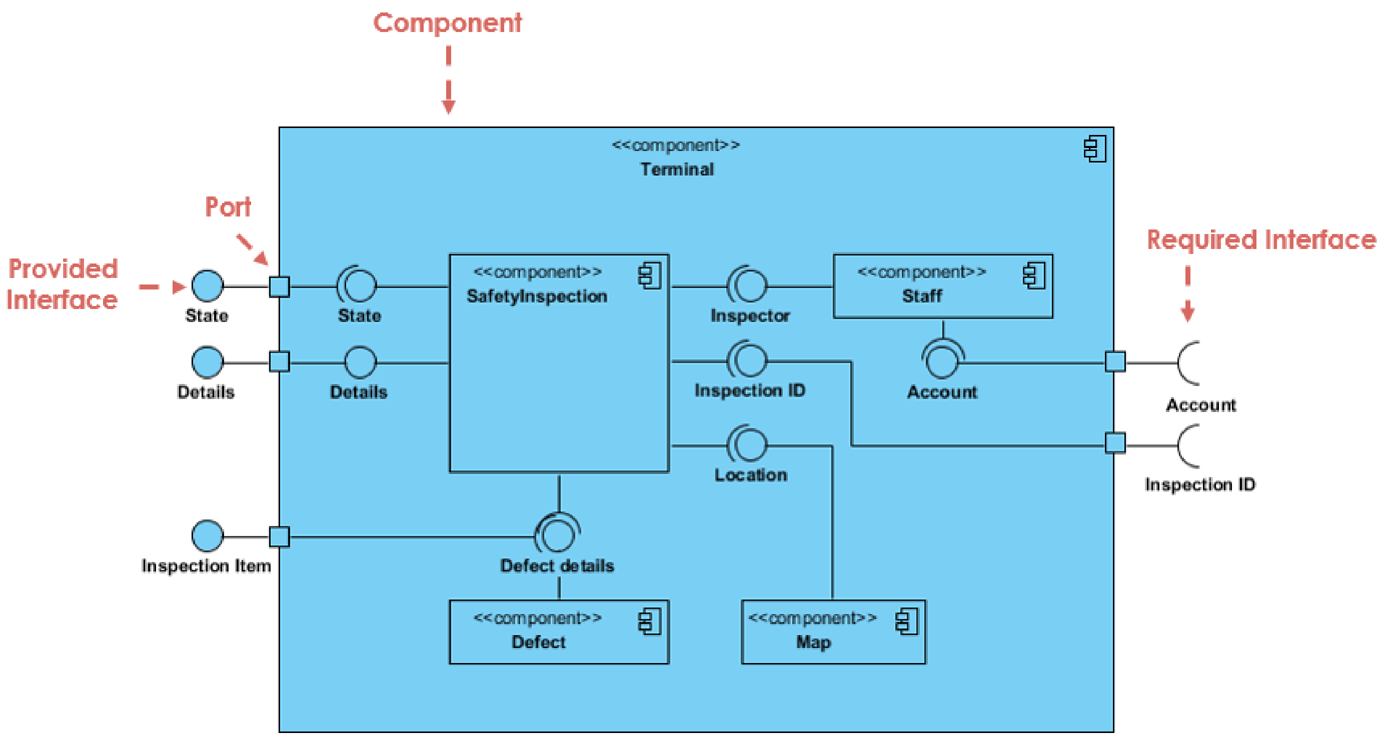


Figure 5: Component diagram

A Component Diagram is a type of UML (Unified Modeling Language) diagram that models the physical aspects of object-oriented systems. It's used for visualizing, specifying, and documenting component-based systems and also for constructing executable systems through forward and reverse engineering.

Here are the main components of a Component Diagram:

- Components: These are modular parts of a system that encapsulate its contents and whose manifestation is replaceable within its environment.
- Interfaces: These are the points of interaction between components. There are two types of interfaces: provided and required. Provided Interface represented with a complete circle at their end, it represents an interface that the component provides. Required Interface represented with only a half circle at their end, it represents an interface that the component requires.

A Component Diagram breaks down the actual system under development into various high levels of functionality. Each component is responsible for one clear aim within the entire system and only interacts with other essential elements on a need-to-know basis.

1.3. Use the UML tool

1.3.1. Diagrams.net

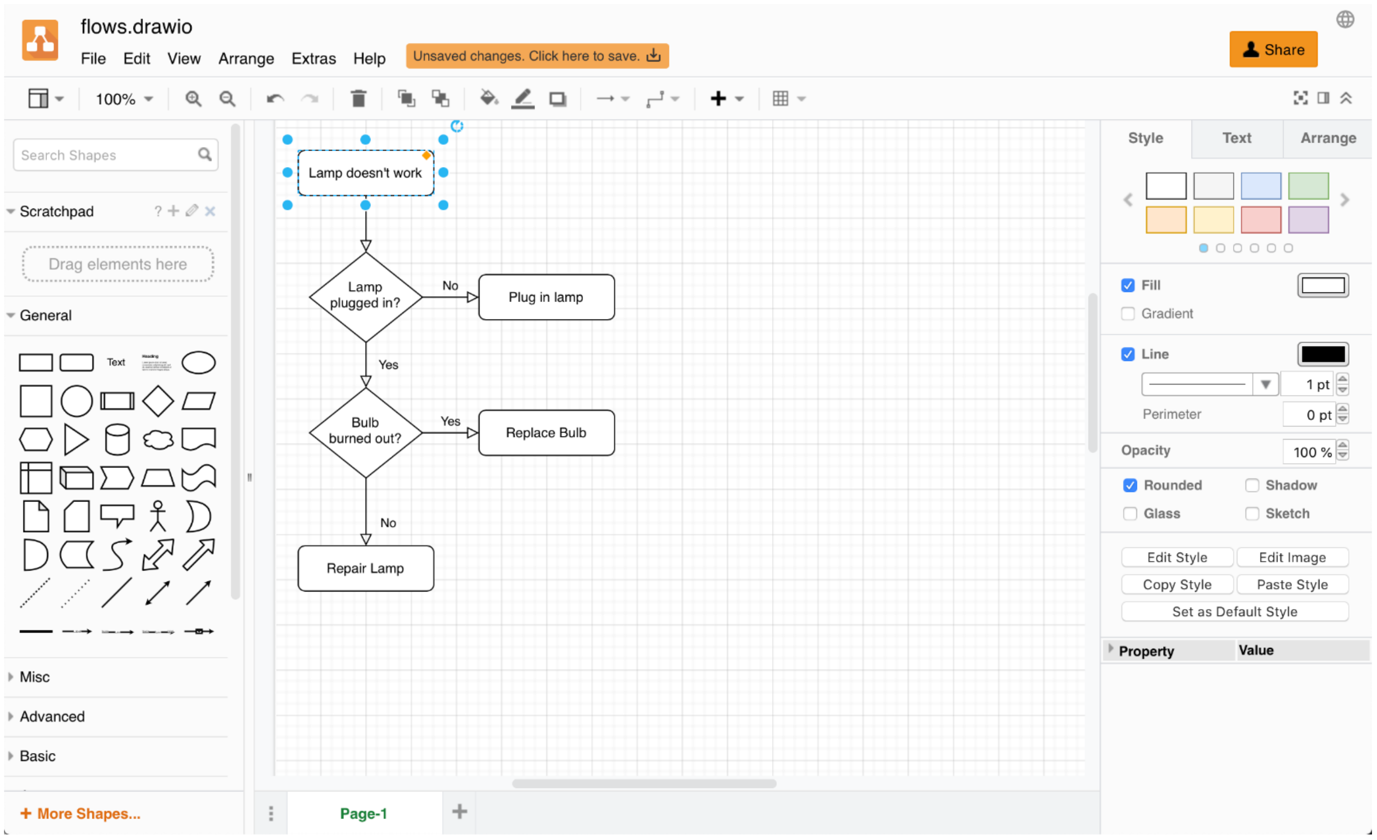


Figure 6: Diagrams.net

Diagram.net, previously known as draw.io, is a cross-platform graph drawing software developed in HTML5 and JavaScript. It's a free online diagram editor that integrates tightly with Google Drive. User can use its interface to create various types of diagrams such as flowcharts, wireframes, UML diagrams, organizational charts, and network diagrams.

Some of the key features of Diagram.net include:

- Security-first diagramming: Your data is stored wherever you want to, and Diagram.net cannot access your data.
- Powerful features: You can collaborate with shared cursors in real-time.
- Cross-platform: Diagram.net is available free of charge as an online web app, and as an offline desktop application for Linux, macOS, and Windows.
- Integration with cloud services: It integrates with cloud services for storage including Dropbox, OneDrive, Google Drive, GitHub, and GitLab.com.
- Embedding in platforms: It is also available as a plugin to embed the web app in platforms such as NextCloud, MediaWiki, Notion, Atlassian Confluence, and Jira.

Advantage:

- Rich functionality: For a web-based application, Diagrams.net offers a wide range of features.
- Integration with cloud storage services: Diagrams.net can save files directly to Google Drive, OneDrive, Dropbox, and more.
- Real-time collaboration: You can work with anyone and share your work easily.
- Great flexibility customizing text labels: This allows for more personalized and detailed diagrams.
- Completely free: Unlike many other diagramming tools, Diagrams.net is completely free.
- Desktop version available: In addition to the web version, there is also a desktop version of Diagrams.net.
- Good list of different output formats: Diagrams.net supports export for PNG, GIF, JPEG, PDF, SVG, HTML, and XML.

Disadvantage:

- Dependent on Google Drive: If there is an error with Google Drive, then Diagrams.net is out of reach, which can be inconvenient if you need access to your charts.
- Saving drafts: If you did not save and download all of your drafts in several formats before turning your device off, you might experience problems with recovering your drafts.
- Only web version available: Desktop or other platforms are not supported.

1.3.2. Lucidchart

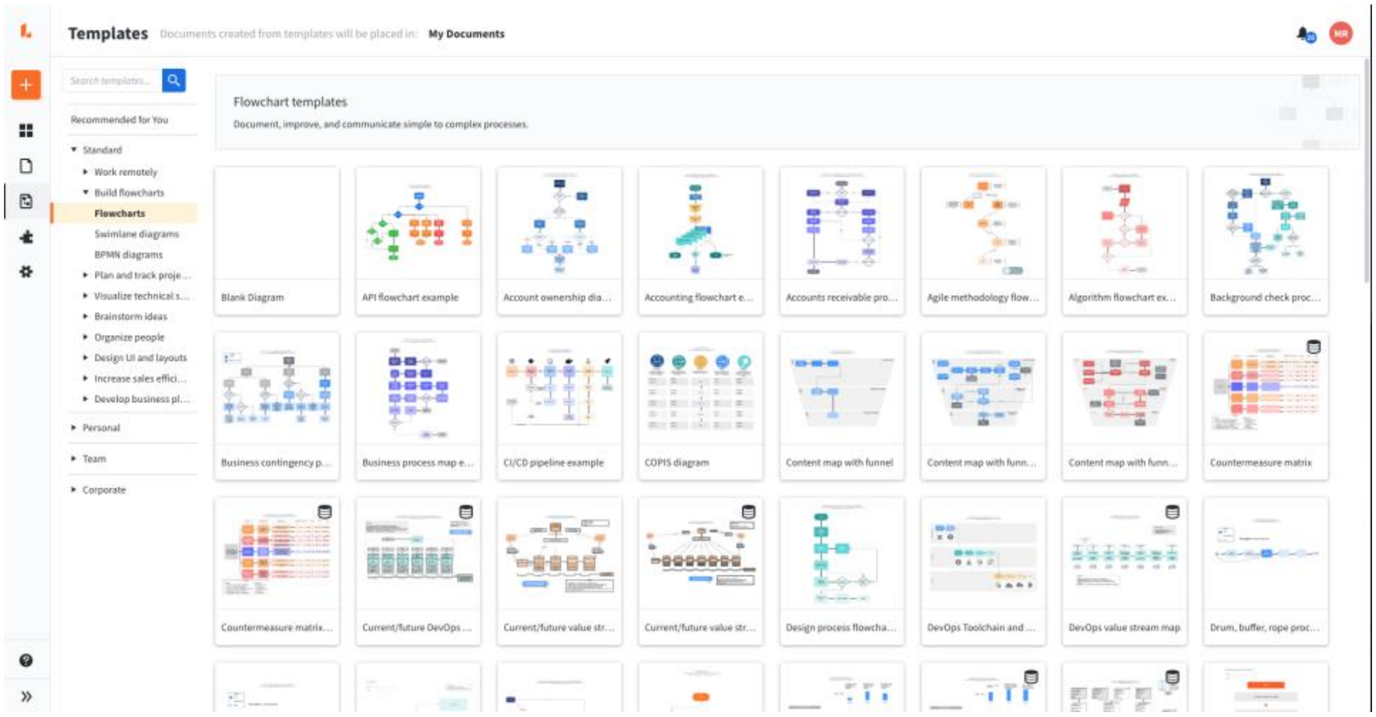


Figure 7: Lucidchart

Lucidchart is an intelligent diagramming application that allows individuals and teams to create and share diagrams. It's used by individuals and teams to clarify complexity, align insights, and build the future faster. Lucidchart is a cloud-based solution where everyone can work visually and collaborate in real time while building flowcharts, mockups, UML diagrams, and more.

Here are some of the key features of Lucidchart:

- Intelligent Diagramming: Lucidchart offers a wide range of features for creating various types of diagrams such as flowcharts, wireframes, UML diagrams, organizational charts, and network diagrams.
- Real-Time Collaboration: Teams can collaborate in real time from anywhere in the world.
- Integration with Industry-Leading Apps: Lucidchart integrates with popular apps like Google Workspace, Atlassian, Slack, and more.
- Security: Lucidchart maintains compliance certifications such as PCI, Privacy Shield, and SOC2.

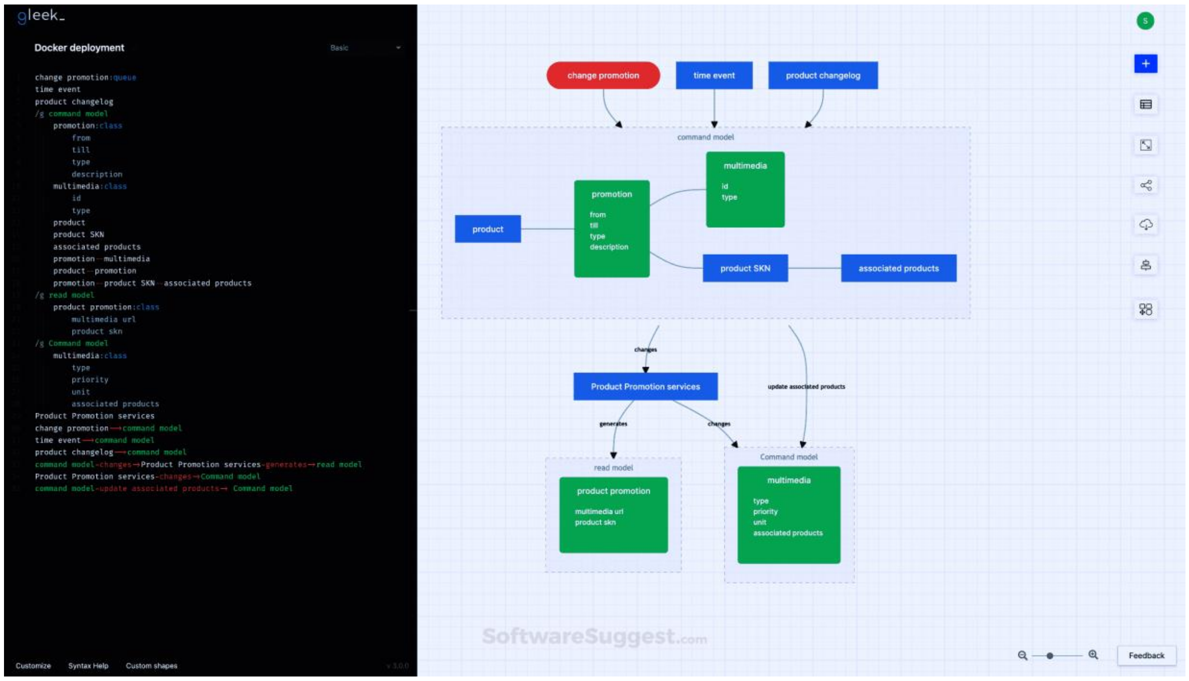
Advantage:

- Extensive collection of libraries and templates: Lucidchart offers a wide range of features for creating various types of diagrams.
- Real-time collaboration: Teams can collaborate in real time from anywhere in the world.
- Integration with services and productivity apps: Lucidchart integrates with popular apps like Google Workspace, Atlassian, Slack, and more.
- Highly-intuitive editing panel: The editing panel is user-friendly and easy to navigate.
- Works with various online web browsers: Lucidchart is compatible with multiple web browsers, providing flexibility for users.

Disadvantage:

- Lack of some essential template categories: Some users have noted that Lucidchart could benefit from a wider variety of template categories.
- Inability to select multiple objects and drag around: Some users have reported difficulty in selecting and dragging multiple objects.
- Auto-resize issues when inserting into documents and updating: Users have reported issues with auto-resizing when inserting diagrams into documents.
- Object limit on the free version: The free version of Lucidchart has a limit on the number of objects you can use.

1.3.3. Gleek.io



```
change promotion:queue
time event
product changelog
/g command model
    promotion:class
        from
        till
        type
        description
    multimedia:class
        id
        type
    product
    product SKN
    associated products
    promotion--multimedia
    product--promotion
    promotion--product SKN--associated products
/g read model
    product promotion:class
        multimedia url
        product skn
/g Command model
    multimedia:class
        type
        priority
        unit
        associated products
Product Promotion services
change promotion-->command model
time event-->command model
product changelog-->command model
command model-changes->Product Promotion services-generates->read model
Product Promotion services-changes->Command model
command model-update associated products-> Command model
```

Figure 8: Gleek.io

Gleek.io is a diagram maker designed specifically for developers. It allows you to create diagrams without using your mouse, instead generating them from descriptions written in its own unique syntax. Here are some of the key features of Gleek.io:

- Visualize Ideas: You can create various types of diagrams such as informal, class, sequence, state, gantt, user journey, or entity-relationship diagrams.
- Rapid Diagramming: Developers can focus on ideas without touching a mouse.
- Version Control: Gleek offers meaningful version control.
- Live Collaboration: Experience working with others. Plan, conceptualize and finalize your project ideas.
- Diagram Export: Export and share diagrams in different formats that suit your requirements.
- Design Templates: Use pre-designed templates to guide you through common use cases.
- Customization: Choose the right look for your diagrams to reflect their nature.

Advantage:

- Efficient Diagramming: Gleek.io saves time in creating complex diagrams.
- Enhanced Clarity: It provides clear visual representations of systems and processes.
- Versatility: Gleek.io is suitable for a range of professional needs[3].
- Ease of Use: It is accessible to users with varying levels of technical expertise.
- Automatic Drawing: The software draws diagrams automatically when you write code.
- Variety of Elements: Gleek.io offers different types of elements like oval or database.

Disadvantage: Learning Curve: It takes some time to learn how to use the tool.

2. Chosen design tools

To carry out this project, our team will use the Diagrams.net design tool (draw.io) because it is a familiar tool, easy to use and suitable for our project team. Diagrams.net will be applied to design diagrams such as use case diagram, UML, activity diagram,... for this project.

3. Development Tools and Techniques

3.1. C#

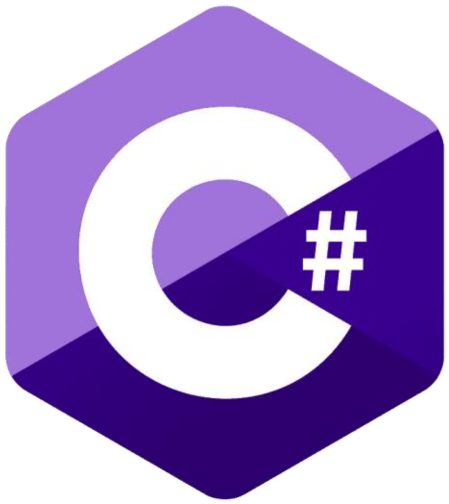


Figure 9: C#

C# (pronounced C Sharp) is a modern, innovative, open-source, cross-platform object-oriented programming language developed by Microsoft. It runs on the .NET Framework and is used to develop a variety of applications, including web apps, desktop apps, mobile apps, and games.

Advantages:

- Object-oriented programming: C# is a completely object-oriented language.
- Cross-platform: Thanks to the wide features of .NET, C# becomes a flexible language that can be used on cross-platform.
- Inbuilt garbage collector: C# has an inbuilt garbage collector. It is a memory manager that keeps track of unused items and automatically releases memory.
- Version control: Version control is efficiently addressed via the assembly approach.

Disadvantages:

- Windows-based platforms: Since C# is a part of the .NET framework, almost all of its applications are for Windows-based platforms. Some of the C# features might not function or be available if you decide to work with a different OS.
- Relies on .NET resources: To run on various operating systems or platforms, C# largely relies on .NET resources. If you're not using .NET as your primary technological stack, it's not all that adaptable on its own.

3.2. ASP.NET Core MVC

Figure 10: ASP.NET Core MVC

ASP.NET Core MVC is a rich framework for building web apps and APIs using the Model-View-Controller (MVC) design pattern. The MVC architectural pattern separates an application into three main groups of components: Models, Views, and Controllers. This pattern helps to achieve separation of concerns:

- Models: The Model in an MVC application represents the state of the application and any business logic or operations that should be performed by it.
- Views: Views are responsible for presenting content through the user interface. They use the Razor view engine to embed .NET code in HTML markup.
- Controllers: Controllers are the components that handle user interaction, work with the model, and ultimately select a view to render.

Key Features:

- Cross-platform: ASP.NET Core MVC is a lightweight, open-source, highly testable presentation framework optimized for use with ASP.NET Core.
- Separation of Concerns: This separation allows the model to be built and tested independent of the visual presentation.
- Scalability: This delineation of responsibilities helps you scale the application in terms of complexity because it's easier to code, debug, and test something (model, view, or controller) that has a single job.

Advantages:

- Cross-platform compatibility: ASP.NET Core MVC is cross-platform, meaning it can run on Windows, Linux, and macOS. This flexibility allows developers to choose their preferred development environment.
- High performance: ASP.NET Core is known for its high performance, especially when compared to its predecessor, ASP.NET MVC. It offers features like built-in dependency injection, asynchronous programming, and the ability to host on lightweight servers like Kestrel.
- Modular and lightweight: ASP.NET Core is designed to be modular, allowing developers to include only the necessary components in their applications. This results in smaller application footprints and faster startup times.
- Open-source: ASP.NET Core is open-source, which means the community can contribute to its development and provide feedback. This leads to continuous improvement and ensures that the framework stays up-to-date with modern web development practices.
- Built-in support for modern web standards: ASP.NET Core MVC includes built-in support for features like dependency injection, RESTful APIs, and model binding, making it easier to develop modern web applications.

Disadvantages:

- Less mature ecosystem: Compared to other web development frameworks like Node.js or Ruby on Rails, the ASP.NET Core ecosystem may be considered less mature. This can mean fewer third-party libraries and resources available for developers.
- Compatibility issues with legacy code: Migrating existing ASP.NET MVC applications to ASP.NET Core MVC can be challenging due to compatibility issues with legacy code and third-party dependencies.
- Performance overhead for small-scale applications: While ASP.NET Core offers high performance, the overhead of the framework may be considered unnecessary for small-scale applications or projects with simple requirements.
- Limited support for older .NET Frameworks: ASP.NET Core primarily targets the .NET Core and .NET platforms, which may not be compatible with older .NET Framework applications without significant modifications.
- Community support and documentation: While ASP.NET Core has a growing community, it may not be as extensive as communities for other web development frameworks. This could lead to challenges in finding solutions to specific problems or getting timely support.

3.3. Visual Studio 2022

Figure 11: Visual Studio 2022

Visual Studio 2022 is a powerful Integrated Development Environment (IDE) developed by Microsoft. It's designed to support the entire development cycle in one place, including writing, editing, debugging, building, and deploying your app.

Here are some key features of Visual Studio 2022:

- Code Editor: Visual Studio provides a powerful code editor with features like syntax highlighting, code completion, and refactoring tools to enhance productivity.
- Debugger: It offers a built-in debugger with advanced features such as breakpoints, watch windows, and real-time code execution tracking, which helps developers identify and fix bugs efficiently.
- Project Management: Visual Studio enables developers to manage their projects effectively by providing project templates, solution explorer, and project properties for configuration.
- Version Control Integration: It seamlessly integrates with popular version control systems like Git, enabling developers to manage their source code repositories directly within the IDE.

- Extensions and Customization: Visual Studio supports a wide range of extensions and customization options, allowing developers to tailor their development environment to suit their specific needs.
- Testing Tools: It includes various testing tools such as unit testing frameworks and code coverage analysis to help developers ensure the quality and reliability of their code.
- Performance Profiling: Visual Studio offers performance profiling tools to analyze and optimize the performance of applications, helping developers identify bottlenecks and improve overall performance.
- Cross-platform Development: With support for multiple programming languages and development platforms, Visual Studio enables developers to build applications for Windows, macOS, Linux, iOS, Android, and the web.

Advantages:

- Feature-rich: Visual Studio 2022 offers a comprehensive set of tools and features for software development, including code editing, debugging, testing, version control, and more.
- Cross-platform development: It supports development for various platforms including Windows, macOS, and Linux, and provides tools for building applications targeting different environments.
- IntelliSense: Visual Studio's IntelliSense feature provides intelligent code completion, code suggestions, and automatic code generation, which can significantly improve productivity.
- Integration: It seamlessly integrates with other Microsoft tools and services such as Azure, Microsoft 365, and GitHub, making it easier to build, test, and deploy applications.
- Extensibility: Visual Studio supports a wide range of extensions and plugins, allowing developers to customize their IDE and add additional functionality as needed.
- Community support: With a large user base and active community, developers can easily find help, tutorials, and resources online when using Visual Studio.

Disadvantages:

- Resource-heavy: Visual Studio can be resource-intensive, especially on older or lower-spec hardware, which may lead to slower performance and longer startup times.
- Cost: While there is a free Community edition available, the Professional and Enterprise editions of Visual Studio come with a cost, which may not be feasible for individual developers or small teams.
- Platform dependence: While Visual Studio supports cross-platform development, some features and tools may be more tailored for Windows development, which could be a limitation for developers targeting other platforms.
- Overwhelming interface: The interface of Visual Studio can sometimes feel cluttered or overwhelming, especially for new users, with numerous windows, panels, and menus.

3.4. SQL Server

Figure 12: SQL Server

SQL Server is a relational database management system (RDBMS) developed by Microsoft. SQL Server is optimally built to be able to operate on very large databases, up to Terabytes. SQL Server provides users with a full range of tools for data management from GUI interface to SQL query language. It is primarily designed to compete with MySQL and Oracle database. SQL Server supports ANSI SQL, which is the standard SQL (Structured Query Language) language. However, SQL Server comes with its own implementation of the SQL language, T-SQL (Transact-SQL). T-SQL is a Microsoft propriety Language known as Transact-SQL.

Advantages:

- SQL Server can be combined with many popular platforms such as ASP.NET, C# to build Winform or SQL Server can also operate independently.
- Data Storage and Manipulation: SQL Server is excellent for storing and manipulating data.
- Highly Scalable: It can handle a large amount of data and users.
- Easy Integration: It integrates well with many frameworks, including .NET.
- Enterprise-Grade Management Software: SQL Server comes with powerful management software.
- Excellent Data Recovery Support: It provides robust support for data recovery.
- Increases Data Security: One of the primary purposes of SQL Server is to ensure the security of your database.

Disadvantages:

- Expensive Pricing: The Enterprise edition can be costly.
- Complicated Licensing: The licensing options can be complex.
- Limited Compatibility: SQL Server has limited compatibility with some systems.
- Requires a lot of Maintenance: It needs regular maintenance to function optimally.

M2 Compare the differences between the various software development tools and techniques researched and justify your preferred selection as well as your preferred software development methodology

1. Compare and justify the preferred selection of development tools and techniques

1.1. Intergrated development environment

1.1.1. Visual Studio 2022

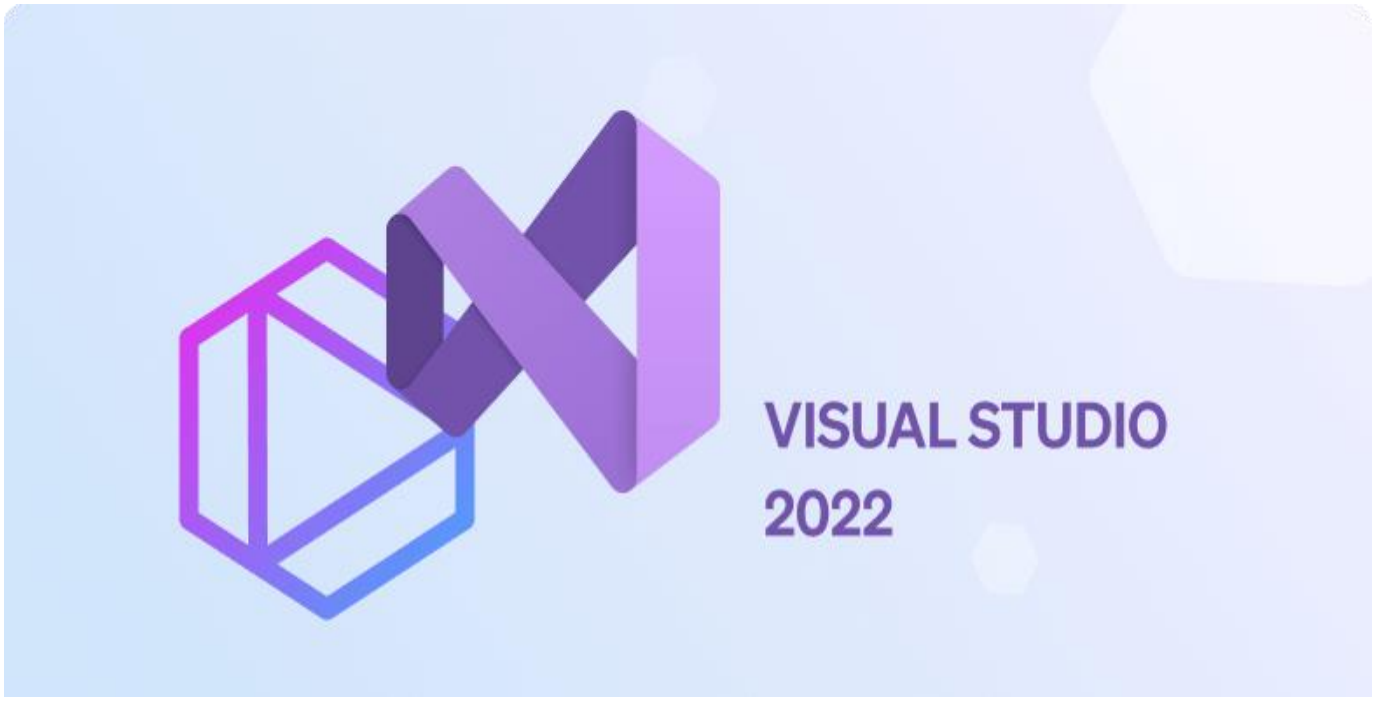


Figure 13: Visual Studio 2022

Visual Studio 2022 is a robust integrated development environment (IDE) designed by Microsoft to support various programming languages and development platforms. It offers a wide range of tools and features for developing applications for multiple platforms, including desktop applications, mobile applications, websites, cloud services and more. Some highlights of Visual Studio 2022 include:

- Support for many programming languages such as C#, C++, JavaScript, Python, and many more.
- Powerful tools for software development, including source code editor, debugger, code checker, and integrated source code management.
- Integrated support for cloud platforms such as Azure, AWS, and Google Cloud Platform.
- Integrate with new technologies like .NET 6, Blazor, Xamarin, and ASP.NET Core.
- Highly customizable to tailor the working environment to user needs.

Advantages:

- Feature-rich: Visual Studio 2022 offers a comprehensive set of tools and features for software development, including code editing, debugging, testing, version control, and more.
- Cross-platform development: It supports development for various platforms including Windows, macOS, and Linux, and provides tools for building applications targeting different environments.
- IntelliSense: Visual Studio's IntelliSense feature provides intelligent code completion, code suggestions, and automatic code generation, which can significantly improve productivity.
- Integration: It seamlessly integrates with other Microsoft tools and services such as Azure, Microsoft 365, and GitHub, making it easier to build, test, and deploy applications.
- Extensibility: Visual Studio supports a wide range of extensions and plugins, allowing developers to customize their IDE and add additional functionality as needed.
- Community support: With a large user base and active community, developers can easily find help, tutorials, and resources online when using Visual Studio.

Disadvantages:

- Resource-heavy: Visual Studio can be resource-intensive, especially on older or lower-spec hardware, which may lead to slower performance and longer startup times.
- Cost: While there is a free Community edition available, the Professional and Enterprise editions of Visual Studio come with a cost, which may not be feasible for individual developers or small teams.
- Platform dependence: While Visual Studio supports cross-platform development, some features and tools may be more tailored for Windows development, which could be a limitation for developers targeting other platforms.
- Overwhelming interface: The interface of Visual Studio can sometimes feel cluttered or overwhelming, especially for new users, with numerous windows, panels, and menus.

1.1.2. Sublime Text

```
<p>
  <a href="https://svelte.dev">
  <img alt="Cybernetically enhanced web apps: Svelt
  </a>
  <a href="https://www.npmjs.com/package/svelte">
    <img src="https://img.shields.io/npm/v/svelte.s
  </a>
  <a href="https://github.com/sveltejs/svelte/blob/
    <img src="https://img.shields.io/npm/l/svelte.s
  </a>
  <a href="https://svelte.dev/chat">
    <img src="https://img.shields.io/discord/457912
  </a>
</p>


## What is Svelte?

Svelte is a new way to build web applications. It's

Learn more at the [Svelte website](https://svelte.d


## Supporting Svelte

Svelte is an MIT-licensed open source project with
```

```
# Contributing to Svelte

Svelte is a new way to build web applications.
It's a compiler that takes your declarative
components and converts them into efficient
JavaScript that surgically updates the DOM.

The [Open Source Guides](https://opensource.guide/
) website has a collection of resources for
individuals, communities, and companies. These
resources help people who want to learn how to
run and contribute to open source projects.
Contributors and people new to open source alike
will find the following guides especially useful:

* [How to Contribute to Open Source](
https://opensource.guide/how-to-contribute/)
* [Building Welcoming Communities](
https://opensource.guide/building-community/)

## Get involved

There are many ways to contribute to Svelte, and
many of them do not involve writing any code.
Here's a few ideas to get started:

- Simply start using Svelte. Go through the [
```

Figure 14: Sublime Text

Sublime Text is an IDE (Integrated Development Environment) for software development, focused on providing the best source code editing experience. It is a powerful and flexible IDE, popular for its speed and performance, clean interface, flexibility, and extensive extensibility. Here are some highlights of Sublime Text:

- Speed and performance: Sublime Text is known for its high performance and fast processing speed. It starts up quickly and responds smoothly when you're working with large files or complex projects.
- Clean and flexible user interface: Sublime Text's interface is designed to be simple and clean, with tools and features arranged logically. Users can customize the interface according to their preferences through themes and packages.
- Flexibility and extensibility: Sublime Text supports a lot of features and plugins to customize and extend its functionality. The Sublime Text community develops diverse packages and plugins, from programming language support tools to special features like debugging, project management, and more.
- Multiple programming language support: Sublime Text supports many popular programming languages, including C/C++, Java, JavaScript, Python, HTML, CSS, and many others.
- Powerful search and replace tools: Sublime Text provides powerful search and replace tools, allowing you to perform source code editing operations quickly and conveniently.

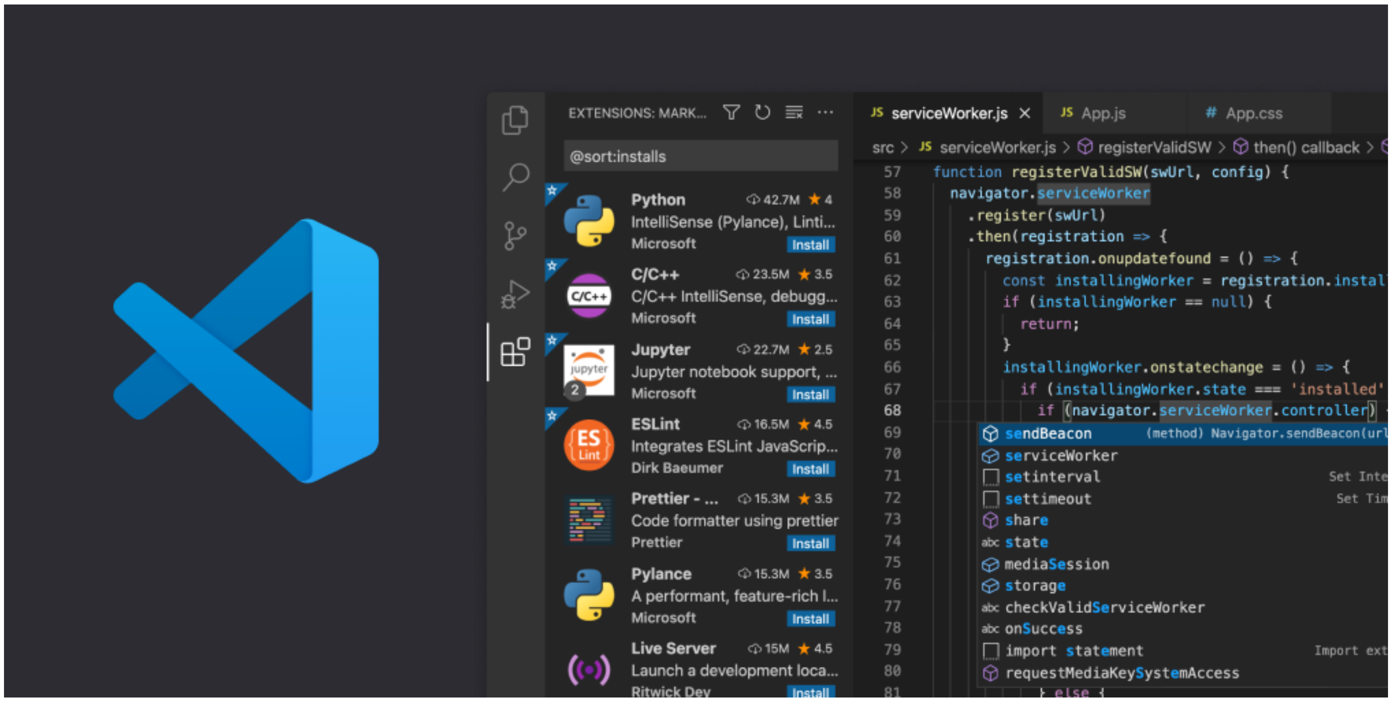
Advantages:

- Speed and performance: Sublime Text is highly rated for its startup and response speed, helping users work efficiently and quickly.
- Clean and flexible interface: Sublime Text's interface is designed to be simple and clean, with tools and features arranged logically, creating a clean and easy-to-use feel.
- Flexibility and extensibility: Sublime Text supports a wide range of plugins and packages, allowing users to customize and extend its functionality according to their specific needs.
- Multi-language support: Sublime Text supports many popular programming languages, making diverse application development convenient.

Disadvantages:

- Cost: Although Sublime Text offers a free trial version, the paid version is quite expensive compared to some other IDEs with similar features.
- Lacks some advanced features: Compared to some other IDEs like Visual Studio or IntelliJ IDEA, Sublime Text may lack some advanced features like built-in debugging or large project support.
- Slow updates: Sublime Text's updates and development are not always quick, and it often takes time to integrate new features or patch bugs.

1.1.3. Visual Studio Code (VS Code)



```
function registerValidSW(swUrl, config) {
  navigator.serviceWorker
    .register(swUrl)
    .then(registration => {
      registration.onupdatefound = () => {
        const installingWorker = registration.instal
        if (installingWorker == null) {
          return;
        }
        installingWorker.onstatechange = () => {
          if (installingWorker.state === 'installed'
            if (navigator.serviceWorker.controller)
sendBeacon            (method) Navigator.sendBeacon(url
serviceWorker
setinterval                                  Set Inte
settimeout                                   Set Tim
share
state
mediaSession
storage
checkValidServiceWorker
onSuccess
import statement                             Import ext
requestMediaKeySystemAccess
            } else {
```

Figure 15: Visual Studio Code

Visual Studio Code (VS Code) is a free and open source IDE developed by Microsoft. Here are some highlights of Visual Studio Code:

- Easy to use and flexible: Visual Studio Code's interface is designed to be simple and intuitive, making it easy for users to get acquainted and take advantage of the IDE's features effectively. It is also highly customizable, allowing users to tailor their work environment to their needs.
- Strong support for many programming languages: Visual Studio Code provides good support for many popular programming languages, including C++, C#, Java, JavaScript, Python, HTML, CSS, and many others. This makes it an ideal choice for multilingual projects.
- Diverse Extension and Marketplace: Visual Studio Code has a rich extension and Marketplace system, allowing users to extend and customize the IDE's features according to their specific needs. These extensions include programming support tools, themes, snippets, and many other features.
- Git integration: Visual Studio Code is natively integrated with Git, providing powerful and convenient source code version management tools. Users can perform operations such as commit, pull, push, and merge easily from within the IDE.
- Good performance and cross-platform compatibility: Visual Studio Code is built on Electron, which is a cross-platform desktop application that can run on Windows, macOS, and Linux. It has good performance and is widely compatible across these operating systems.
- Large and active community: Visual Studio Code has a large and active community, with many users contributing extensions, themes, and tutorials from the community, helping users take full advantage of the IDE's features and solve problems. solve the problem effectively.

Advantages:

- Free and open source: Visual Studio Code is released as free and open source, allowing users to customize and extend according to their needs.
- Supports many programming languages: VS Code supports many popular programming languages, including C++, C#, Java, JavaScript, Python, and many others.
- Lightweight: Compared to some other IDEs, VS Code is compact in size and requires few system resources, helping to increase performance on low-configuration computers.
- Large community and strong support: VS Code has a large and active community, with many extensions, themes, and tutorials from the community, helping users take full advantage of the IDE's features.
- Git integration: VS Code integrates natively with Git, allowing users to conveniently manage source code versions from within the IDE.

Disadvantages:

- Lacks some special features: Compared to full IDEs like Visual Studio, VS Code may lack some special features like built-in debuggers for specific languages or UI design tools .
- Requires relatively high computer configuration: Although lighter than some other IDEs, VS Code still requires a computer with relatively good configuration to run smoothly.

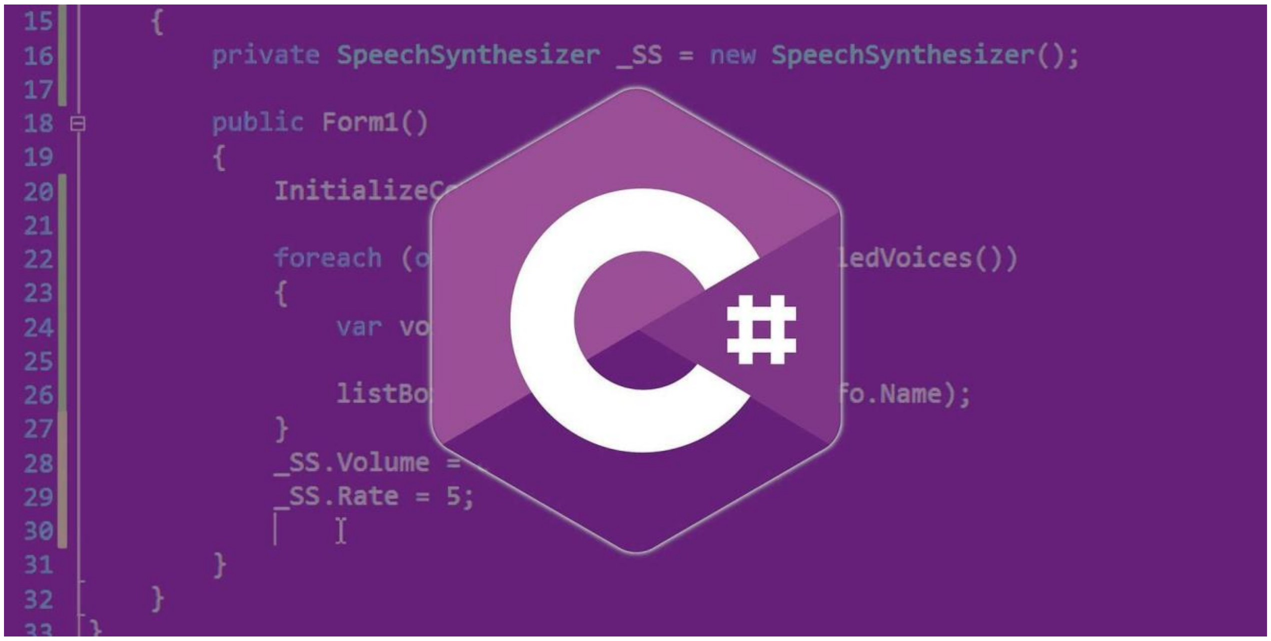
1.1.4. Compare

	Visual Studio	Sublime Text	Visual Studio Code
Performance and Stability	Integrates many powerful features and tools but can consume a lot of system resources and may experience occasional delays.	Very fast and light, quick startup and smooth response. Often chosen for small to medium projects.	Fast, light and stable. Designed to provide high performance with fewer system resources than Visual Studio.
Features and Extensions	Provides a rich set of features, including a powerful debugger, project management, and many other tools. However, there are fewer extensions and plugins compared to other open source IDEs.	Although quite simple by default, it can be strongly extended through community packages and plugins.	Provides a large number of free extensions and plugins developed by the community, helping to expand the IDE's features according to needs.
Programming language support	Supports many programming languages, but mainly .NET and C#.	Supports many programming languages through plugins and packages.	Wide support for many popular programming languages and has a large community developing extensions for different languages.
Multi-platform	Primarily for Windows, there is a Visual Studio Code version for macOS and Linux.	Available on many operating systems, including Windows, macOS and Linux.	Supported on Windows, macOS and Linux.
Price	There is a free version called Visual Studio Community Edition, but professional and enterprise versions are paid.	Free trial version available, but you need to purchase a license for commercial use.	Completely free and open source.

Table 1: Compare IDE

1.2. Programming language

1.2.1. C#



```
    {
        private SpeechSynthesizer _SS = new SpeechSynthesizer();

        public Form1()
        {
            InitializeC

            foreach (o                          ledVoices())
            {
                var vo

                listBo                          fo.Name);
            }
            _SS.Volume =
            _SS.Rate = 5;

        }
    }
}
```

Figure 16: C#

C# (pronounced C Sharp) is a modern, innovative, open-source, cross-platform object-oriented programming language developed by Microsoft. It runs on the .NET Framework and is used to develop a variety of applications, including web apps, desktop apps, mobile apps, and games. It is a powerful and flexible programming language that is widely used in developing different types of applications, from desktop applications to mobile and web applications. Here are some highlights about the C# programming language:

- Multi-purpose: C# is designed to develop a wide range of applications from desktop applications to mobile applications, web applications and software for embedded devices. It is a general-purpose programming language, capable of running on Windows, macOS and Linux platforms.
- Modernity: C# is a modern programming language with many advanced features such as object orientation, type safety, readable syntax, lambda expressions, and LINQ (Language Integrated Query) for querying data. Whether.
- Security: C# is designed with high security in mind, which means it provides protection mechanisms such as access control and type control, which help prevent errors and attacks from external sources.
- Integration with .NET Framework: C# is the main language of the .NET Framework, a popular application development platform from Microsoft. The .NET Framework provides a series of libraries and tools for developing Windows applications, web and cloud services using C#.

- Large community and strong support: C# has a large and active developer community, with extensive documentation, forums, and online learning resources. This makes it easier for C# programmers to seek help and support when needed.

Advantages:

- Object-oriented programming: C# is a completely object-oriented language.
- Cross-platform: Thanks to the wide features of .NET, C# becomes a flexible language that can be used on cross-platform.
- Inbuilt garbage collector: C# has an inbuilt garbage collector. It is a memory manager that keeps track of unused items and automatically releases memory.
- Version control: Version control is efficiently addressed via the assembly approach.

Disadvantages:

- Windows-based platforms: Since C# is a part of the .NET framework, almost all of its applications are for Windows-based platforms. Some of the C# features might not function or be available if you decide to work with a different OS.
- Relies on .NET resources: To run on various operating systems or platforms, C# largely relies on .NET resources. If you're not using .NET as your primary technological stack, it's not all that adaptable on its own.

1.2.2. Java

Figure 17: Java

Java is a high-level, class-based, object-oriented programming language that is designed to have as few implementation dependencies as possible. It is a general-purpose programming language intended to let application developers write once, run anywhere (WORA), meaning that compiled Java code can run on all platforms that support Java without the need for recompilation. Java is a powerful, general-purpose and cross-platform programming language, widely used in the software industry for a variety of applications ranging from desktop applications to mobile and web applications. Here are some highlights about the Java programming language:

- Platform independence: Java is designed to make applications run on many different platforms without having to rewrite the source code.
- Multi-purpose: Java is a multi-purpose language, widely used for developing computer applications, mobile applications, web applications, games, web services, and more.
- Security: Java has a strong security model, including a type checking system and access restrictions. This security model helps prevent common security vulnerabilities such as memory overflows and compromise vulnerabilities.
- Cross-platform: Java source code can be compiled into bytecode, an intermediate language between source code and machine code, and then executed on any Java virtual machine (JVM). This allows Java applications to run on many different platforms without changing the source code.
- Strong support: Java has a large and active community, with extensive documentation, forums, and online learning resources. It also has several popular integrated development tools (IDEs) such as Eclipse, IntelliJ IDEA, and NetBeans.
- Integration of libraries and frameworks: Java has a rich ecosystem of libraries and frameworks, which helps reduce development time and increase application flexibility.

Advantages:

- Platform Independence: Java code can run on any platform that supports Java Virtual Machine (JVM), making it highly portable.
- Object-Oriented: Java is an object-oriented programming language, which promotes modularity, reusability, and maintainability of code.
- Rich API: Java provides a vast standard library with a wide range of classes and methods that simplify development tasks.
- Strong Community Support: Java has a large and active community of developers, which means ample resources, libraries, and frameworks are available for almost any task.
- Automatic Memory Management: Java's built-in garbage collection mechanism automatically manages memory allocation and deallocation, reducing the risk of memory leaks and simplifying memory management.
- Multi-threading: Java supports multithreading, allowing developers to create applications that can execute multiple tasks simultaneously, enhancing performance and responsiveness.
- Security: Java's security features, such as bytecode verification and sandboxing, make it a preferred choice for developing secure applications, especially for web and enterprise environments.

Disadvantages:

- Performance Overhead: Java programs can suffer from a performance overhead due to the need for interpretation by the JVM and garbage collection processes.
- Complexity: Java can be verbose and sometimes complex, especially for beginners. The extensive syntax and various design patterns may require a steep learning curve for some developers.
- Memory Consumption: Java applications tend to consume more memory compared to languages like C or C++, mainly because of the JVM overhead and automatic memory management.
- Limited Hardware Access: Java's platform independence comes with limitations, especially in terms of accessing hardware directly, which might be necessary for certain types of applications.
- Speed of Development: While Java promotes robust and scalable applications, the development process may sometimes be slower compared to languages with more concise syntax and rapid prototyping capabilities.
- GUI Development: Although Java provides GUI development libraries like Swing and JavaFX, building complex and visually appealing user interfaces can be challenging compared to some other modern frameworks.
- Runtime Dependency: Since Java applications require the JVM to be installed on the target system, they have a runtime dependency, which may increase deployment complexity and potentially cause version compatibility issues.

1.2.3. Python

Figure 18: Python

Python is an interpreted, easy-to-learn and flexible programming language, developed by Guido van Rossum in the 1980s and first released in 1991. It is famous for its simple and easy-to-understand syntax, which helps Coding becomes easier and more natural than in many other programming languages. Python has become one of the most popular programming languages in the world and is widely used in many different fields from web development to data analysis and artificial intelligence.

Here are some highlights about Python:

- Easy to learn and read: Python has a syntax very close to natural language, making the source code easy to read and understand. This makes Python an ideal choice for both beginners and experienced programmers.
- Multi-purpose: Python is a multi-purpose language that can be used for a variety of purposes including web development, data analysis, machine learning, artificial intelligence, systems programming, and more.
- Large Community and Strong Support: Python has a large and active developer community. There are many documents, forums, and learning resources available online. This makes learning and developing Python applications easier.
- Portability: Python is available on a variety of platforms, including Windows, macOS, and Linux, making it a good choice for cross-platform software development.
- Rich with libraries and frameworks: Python has a rich ecosystem of libraries and frameworks, which helps reduce development time and increase application flexibility. Popular libraries and frameworks include Django and Flask for web development, NumPy and Pandas for data analysis, and TensorFlow and PyTorch for machine learning and artificial intelligence.
- Security: Python provides tools and libraries to develop safe and secure applications. However, like any other programming language, building secure applications requires solid knowledge and good practices on the part of the developer.

Advantages:

- Simple and Readable Syntax: Python's syntax is designed to be intuitive and readable, making it easier for beginners to grasp and write code quickly.
- Wide Range of Libraries: Python has an extensive standard library and a vast ecosystem of third-party libraries, making it suitable for various tasks such as web development, data analysis, machine learning, and more.
- Community Support: Python has a large and active community of developers who contribute to its development, create libraries, provide support, and share knowledge through forums, online communities, and conferences.
- Cross-platform Compatibility: Python is available on multiple platforms such as Windows, macOS, and Linux, making it versatile for developing applications across different operating systems.
- High-level Language: Python abstracts many complex details away from the programmer, allowing them to focus more on problem-solving rather than low-level implementation details.
- Dynamic Typing: Python is dynamically typed, meaning you don't need to specify the data type of variables explicitly. This can lead to faster development time and easier maintenance.
- Extensive Documentation: Python has comprehensive and well-maintained documentation, which makes it easier for developers to find information and learn the language.

Disadvantages:

- Performance: Compared to lower-level languages like C or C++, Python can be slower in terms of execution speed and memory consumption, especially for CPU-intensive tasks.
- Global Interpreter Lock (GIL): Python's Global Interpreter Lock can limit concurrency in multi-threaded programs, affecting performance in CPU-bound applications.
- Mobile Development: While Python can be used for mobile app development using frameworks like Kivy or BeeWare, it's not as commonly used or supported as languages like Java or Swift.
- Dependency Management: Managing dependencies in Python projects can sometimes be challenging, especially when dealing with conflicting versions or complex dependency trees.
- Weak in Mobile and Game Development: While Python has libraries and frameworks for mobile and game development, it's not as strong in these areas compared to languages like Java for Android or C# for game development with Unity.
- Design Restrictions: Python's enforced indentation can sometimes be seen as restrictive, especially for developers coming from languages with more flexible syntax rules.

1.2.4. Compare

	C#	Java	Python
Use	C# has an easy to read and understand syntax, designed to minimize errors and increase developer productivity.	Java also has quite clear syntax, but it may take some time to get used to concepts such as data types, classes and exceptions.	Python has a simple and easy-to-understand syntax, making it a good choice for beginners. It offers a concise approach to programming and has a wealth of online documentation and learning resources.
Multi purpose	C# is often used in developing Windows applications, games and applications for Microsoft systems.	Java is also a multi-purpose language and is widely used in mobile application development (Android), web development, and many other applications.	Python is a powerful general-purpose language, used in web development, data analysis, machine learning, artificial intelligence, and many other fields.
Community and support	C# also has a strong community, especially in Windows application and game development.	Java also has a large and active community, with many online learning and support resources.	Python has a large and active developer community, with many documents, forums, and online learning resources.
Portability and cross-platform	C# is available on the Windows platform and can be used on other operating systems through projects such as .NET Core. However, mobile application development is not as powerful as Java.	Java is a cross-platform language and is designed to run on any platform that has a JVM.	Python was not originally designed for cross-platform support but can run on many different operating systems.

Table 2: Compare Programming language

1.3. Web framework

1.3.1. ASP.NET Core MVC

Figure 19: ASP.NET Core MVC

ASP.NET Core MVC is a rich framework for building web apps and APIs using the Model-View-Controller (MVC) design pattern. The MVC architectural pattern separates an application into three main groups of components: Models, Views, and Controllers. This pattern helps to achieve separation of concerns:

- Models: The Model in an MVC application represents the state of the application and any business logic or operations that should be performed by it.
- Views: Views are responsible for presenting content through the user interface. They use the Razor view engine to embed .NET code in HTML markup.
- Controllers: Controllers are the components that handle user interaction, work with the model, and ultimately select a view to render.

ASP.NET Core MVC provides a structured and modular approach to building web applications, making it easier to manage code complexity, separate concerns, and maintainability. It also offers features such as routing, middleware, dependency injection, and authentication to help developers build robust and scalable web applications.

Advantages:

- Cross-platform compatibility: ASP.NET Core MVC is cross-platform, meaning it can run on Windows, Linux, and macOS. This flexibility allows developers to choose their preferred development environment.
- High performance: ASP.NET Core is known for its high performance, especially when compared to its predecessor, ASP.NET MVC. It offers features like built-in dependency injection, asynchronous programming, and the ability to host on lightweight servers like Kestrel.
- Modular and lightweight: ASP.NET Core is designed to be modular, allowing developers to include only the necessary components in their applications. This results in smaller application footprints and faster startup times.
- Open-source: ASP.NET Core is open-source, which means the community can contribute to its development and provide feedback. This leads to continuous improvement and ensures that the framework stays up-to-date with modern web development practices.
- Built-in support for modern web standards: ASP.NET Core MVC includes built-in support for features like dependency injection, RESTful APIs, and model binding, making it easier to develop modern web applications.

Disadvantages:

- Less mature ecosystem: Compared to other web development frameworks like Node.js or Ruby on Rails, the ASP.NET Core ecosystem may be considered less mature. This can mean fewer third-party libraries and resources available for developers.
- Compatibility issues with legacy code: Migrating existing ASP.NET MVC applications to ASP.NET Core MVC can be challenging due to compatibility issues with legacy code and third-party dependencies.
- Performance overhead for small-scale applications: While ASP.NET Core offers high performance, the overhead of the framework may be considered unnecessary for small-scale applications or projects with simple requirements.
- Limited support for older .NET Frameworks: ASP.NET Core primarily targets the .NET Core and .NET platforms, which may not be compatible with older .NET Framework applications without significant modifications.
- Community support and documentation: While ASP.NET Core has a growing community, it may not be as extensive as communities for other web development frameworks. This could lead to challenges in finding solutions to specific problems or getting timely support.

1.3.2. Spring MVC

Figure 20: Spring MVC

Spring MVC is a widely-used web application framework built on top of the Java-based Spring Framework. It follows the Model-View-Controller (MVC) architectural pattern, providing developers with a flexible and powerful approach to developing web applications in Java.

At the core of Spring MVC is the DispatcherServlet, serving as the front controller responsible for managing incoming HTTP requests. This servlet handles request routing, exception handling, and overall request lifecycle management.

Controllers play a pivotal role in Spring MVC, serving as the primary component responsible for processing incoming requests, executing business logic, and returning appropriate responses. Controllers are Java classes annotated with `@Controller` or `@RestController` annotations, indicating their role in handling HTTP requests. These controllers contain handler methods annotated with `@RequestMapping` or other mapping annotations, specifying the URL patterns and HTTP methods they can handle.

The Model in Spring MVC represents the application's data and business logic. Typically, the Model is comprised of Plain Old Java Objects (POJOs) or Data Transfer Objects (DTOs). Controllers interact with the Model to retrieve or update data, which is then passed to the View for rendering. Views in Spring MVC are responsible for presenting the user interface and displaying data to the user. Spring MVC supports various view technologies such as JSP (JavaServer Pages), Thymeleaf, FreeMarker, and Velocity.

Handler Mapping is another essential component of Spring MVC, responsible for mapping incoming requests to the appropriate handler methods in the controllers. Spring MVC provides several built-in handler mapping strategies to facilitate request handling. Additionally, View Resolver plays a crucial role in resolving logical view names returned by the controllers to actual view implementations. Different view resolver implementations are available in Spring MVC to support various view technologies.

Interceptors are a key feature in Spring MVC, allowing developers to intercept and pre/post-process HTTP requests and responses. Interceptors are commonly used for tasks such as logging, authentication, authorization, and request/response manipulation. They provide a way to apply cross-cutting concerns to multiple controller methods within an application.

Advantages:

- Modularity and Flexibility: Spring MVC follows the principle of dependency injection, allowing for loose coupling between components. This makes the application more modular and easier to maintain.
- Testability: Spring MVC facilitates unit testing and integration testing by allowing dependencies to be injected into controllers, making it easier to isolate and test individual components.
- Annotation-Based Configuration: Spring MVC supports annotation-based configuration, which reduces boilerplate code and simplifies the configuration of controllers and mappings.
- Integration with Other Spring Framework Modules: Spring MVC seamlessly integrates with other Spring modules like Spring Security, Spring Data, and Spring Boot, providing additional functionality and features for building robust applications.
- Convention over Configuration: Spring MVC provides sensible defaults and conventions, reducing the need for explicit configuration. This helps developers focus on business logic rather than configuration details.
- Extensive Community Support: Spring MVC has a large and active community, which means there are plenty of resources, tutorials, and libraries available to help developers solve problems and build applications.
- Built-in Support for Validation: Spring MVC provides built-in support for data validation using annotations or custom validators, making it easier to enforce data integrity and validate user input.

Disadvantages:

- XML Configuration: Although Spring MVC supports annotation-based configuration, some legacy projects may still use XML configuration, which can be verbose and harder to maintain compared to annotations.
- Boilerplate Code: While Spring MVC reduces boilerplate code compared to traditional Java EE frameworks, it still requires developers to write some repetitive code for setting up controllers, mappings, and configurations.
- Performance Overhead: Spring MVC, like any framework, introduces some performance overhead compared to plain Java Servlets due to its abstraction layers and additional features. However, for most applications, the overhead is negligible.
- Version Compatibility: Managing dependencies and ensuring compatibility between different versions of Spring modules can be challenging, especially in large projects with many dependencies.
- Configuration Complexity: Although Spring MVC simplifies configuration in many cases, configuring advanced features or integrating with complex systems can still be challenging and require deep understanding of the framework.
- Overkill for Small Projects: For small projects or simple web applications, the overhead of using Spring MVC may outweigh the benefits. In such cases, simpler frameworks or libraries may be more suitable.

Overall, Spring MVC offers a comprehensive and modular framework for building robust and scalable web applications in Java. Its adherence to the MVC architectural pattern, coupled with its extensive set of features and components, makes it a popular choice among Java developers for developing web applications.

1.3.3. Django

Figure 21: Django

Django is a high-level Python web framework that encourages rapid development and clean, pragmatic design. It follows the Model-View-Controller (MVC) architectural pattern, although Django developers often refer to it as a Model-View-Template (MVT) framework due to its slight variation. Django provides developers with a set of tools and conventions to build web applications efficiently, allowing them to focus on writing their application's logic.

At the core of Django is the concept of "apps," which are reusable components that encapsulate related functionality. Each app typically consists of models, views, and templates. Models define the data structure of the application using Python classes, representing database tables and their relationships. Views handle HTTP requests and generate HTTP responses, typically using functions or class-based views. Templates are HTML files that define the presentation layer of the application and are rendered dynamically with data provided by views.

Django's architecture promotes the separation of concerns, making it easier to maintain and extend applications over time. The framework provides built-in support for common web development tasks such as URL routing, form handling, authentication, and session management. Django's Object-Relational Mapping (ORM) system abstracts away the complexities of database interaction, allowing developers to work with database objects using Python syntax rather than SQL.

One of Django's key features is its admin interface, which automatically generates a user-friendly interface for managing application data. Developers can customize the admin interface to suit their application's specific needs, making it easy to perform CRUD (Create, Read, Update, Delete) operations on database objects without writing any additional code.

Advantages:

- Rapid Development: Django follows the "don't repeat yourself" (DRY) principle, which means you can write less code while accomplishing more. It provides a plethora of built-in features and libraries for common web development tasks, allowing developers to focus on business logic rather than boilerplate code.
- Scalability: Django is capable of handling high-traffic websites and scaling horizontally. It offers support for caching, database sharding, and load balancing, enabling developers to build scalable applications.
- Security: Django provides built-in protection against common security threats such as SQL injection, cross-site scripting (XSS), cross-site request forgery (CSRF), and clickjacking. It encourages secure coding practices and follows industry best practices for security.
- Community and Ecosystem: Django has a large and active community of developers who contribute plugins, packages, and tutorials. This vibrant ecosystem makes it easier to find solutions to problems, learn new techniques, and stay updated with the latest developments.
- Documentation: Django has extensive and well-maintained documentation, including tutorials, guides, and references. This makes it easy for developers, especially beginners, to get started with the framework and troubleshoot issues.

Disadvantages:

- Performance Overhead: Django's high-level abstractions and features come with a performance overhead compared to microframeworks like Flask. For performance-critical applications, developers may need to optimize Django applications by caching, optimizing database queries, and using efficient algorithms.
- ORM Limitations: While Django's ORM simplifies database interactions, it may not be suitable for complex database operations or performance-sensitive applications. In such cases, developers may need to bypass the ORM and write raw SQL queries, which can increase complexity and reduce portability.
- Opinionated: Django enforces certain conventions and best practices, which may not align with every project's requirements or development style. Developers who prefer more flexibility and freedom may find Django's opinionated nature restrictive.

In summary, Django is a powerful and versatile web framework that empowers developers to build web applications quickly and efficiently. Its clean and pragmatic design, along with its rich set of features, makes it an excellent choice for projects of all sizes and complexities.

1.3.4. Compare

	ASP.NET Core MVC	Spring MVC	Django
Language	Developed by Microsoft, ASP.NET Core MVC is primarily used with C# language.	Spring MVC is based on Java and is part of the larger Spring Framework ecosystem.	Django is written in Python, making it a popular choice among Python developers.
Ecosystem	Part of the Microsoft ecosystem, ASP.NET Core MVC integrates seamlessly with other Microsoft technologies such as Visual Studio, Azure, and Entity Framework.	Spring MVC is part of the Spring Framework ecosystem, which offers a wide range of modules and integrations for enterprise Java development.	Django has a rich ecosystem of third-party packages and libraries available through the Python Package Index (PyPI). It also provides integration with popular Python tools and frameworks.
Architecture	ASP.NET Core MVC follows the Model-View-Controller (MVC) architectural pattern.	Spring MVC also follows the MVC pattern, although it's often referred to as a Model-View-Controller (MVC) framework.	Django follows a slight variation of the MVC pattern known as Model-View-Template (MVT), where templates are used instead of views.
Features	ASP.NET Core MVC provides features such as dependency injection, middleware pipeline, Razor view engine, built-in authentication, and authorization.	Spring MVC offers features such as robust support for dependency injection (using the Spring IoC container), powerful configuration options, aspect-oriented programming (AOP), and integration with other Spring modules.	Django provides features such as an Object-Relational Mapping (ORM) system, built-in admin interface, automatic URL routing, template engine, authentication, and authorization.
Community Support	ASP.NET Core MVC has a strong community of developers and contributors, with extensive documentation and resources available on the official Microsoft website and other platforms.	Spring MVC benefits from a large and active community of Java developers, with comprehensive documentation, tutorials, and forums available.	Django has a vibrant and welcoming community of Python developers, with a wealth of resources, tutorials, packages, and community-driven projects available.

Table 3: Compare Web framework

1.4. Database management system

1.4.1. SQL Server

Figure 22: SQL Server

SQL Server is a relational database management system (RDBMS) developed by Microsoft. It is a powerful and feature-rich database platform used by organizations of all sizes for storing, retrieving, and managing data. SQL Server offers a wide range of functionalities, tools, and services for database administration, development, and business intelligence. Here are some key aspects of SQL Server:

- Data Storage: SQL Server stores data in tables, organized into databases. It supports structured data types such as integers, strings, dates, and floating-point numbers, as well as complex data types like XML and JSON.
- Query Language: SQL Server uses the Transact-SQL (T-SQL) query language for interacting with the database. T-SQL is a powerful extension of SQL that provides additional features such as procedural programming constructs, error handling, and transaction management.
- Security: SQL Server offers robust security features to protect data from unauthorized access, including role-based security, encryption, auditing, and row-level security.
- Scalability: SQL Server is designed to scale from small single-server deployments to large, distributed architectures. It supports features like partitioning, parallel processing, and in-memory processing to handle growing workloads.
- Community and Support: SQL Server has a large and active community of developers and database administrators, with forums, documentation, tutorials, and online resources available to help users troubleshoot issues, learn new features, and share best practices.

Advantages:

- SQL Server can be combined with many popular platforms such as ASP.NET, C# to build Winform or SQL Server can also operate independently.
- Data Storage and Manipulation: SQL Server is excellent for storing and manipulating data.
- Highly Scalable: It can handle a large amount of data and users.
- Easy Integration: It integrates well with many frameworks, including .NET.
- Enterprise-Grade Management Software: SQL Server comes with powerful management software.
- Excellent Data Recovery Support: It provides robust support for data recovery.
- Increases Data Security: One of the primary purposes of SQL Server is to ensure the security of your database.

Disadvantages:

- Expensive Pricing: The Enterprise edition can be costly.
- Complicated Licensing: The licensing options can be complex.
- Limited Compatibility: SQL Server has limited compatibility with some systems.
- Requires a lot of Maintenance: It needs regular maintenance to function optimally.

1.4.2. MySQL

Figure 23: MySQL

MySQL is an open-source relational database management system (RDBMS) developed by Oracle Corporation. It originated in 1989 and has since become one of the most popular database management systems worldwide, widely used in web applications, enterprises, and open-source projects. Here are some key features of MySQL:

- Open Source: MySQL is an open-source project, allowing users to download, use, and modify the source code freely. This fosters a vibrant community of developers and users worldwide.
- Lightweight and Fast: MySQL is designed to operate efficiently on servers with limited resources. It has a small footprint, consumes minimal system resources, and can handle thousands of queries per second.
- Cross-Platform Support: MySQL is available for various operating systems such as Windows, Linux, macOS, and others, making it suitable for a wide range of applications and deployment environments.
- Modular Architecture: MySQL uses a modular architecture, allowing users to choose and activate specific features based on their needs. This helps reduce overhead and optimize performance.
- Diverse Development Tools: MySQL provides a range of development and management tools such as MySQL Workbench, phpMyAdmin, MySQL Shell, and MySQL Connector to help users easily manage their databases.
- Security: MySQL offers robust security features such as user management, access control, data encryption, and logging to protect data from security threats.
- Community and Support: MySQL has a large and active community of developers and database administrators, with extensive documentation, forums, and online resources to help users troubleshoot issues and share knowledge.

Advantages:

- Open Source: MySQL is an open-source RDBMS, meaning it's freely available for use and modification under the GNU General Public License (GPL).
- Community Support: MySQL has a large and active community of developers and users who contribute to its development, provide support, and share resources like tutorials and plugins.
- Scalability: MySQL is designed to handle large volumes of data and is highly scalable. It can be used in small projects as well as large enterprise-level applications.
- Cross-Platform Compatibility: MySQL is compatible with various operating systems including Linux, Windows, macOS, and others, making it versatile for different environments.
- High Performance: MySQL is known for its fast performance and efficient storage engines. With proper optimization and indexing, it can handle high traffic and large datasets efficiently.
- Security Features: MySQL provides various security features such as user authentication, encryption support, access controls, and auditing, which help in securing data against unauthorized access and attacks.

Disadvantages:

- Limited Functionality: Compared to some other RDBMS like PostgreSQL or Oracle, MySQL may have fewer advanced features and functionalities, although it's continuously evolving to address this.
- Transactions and ACID Compliance: While MySQL supports transactions and ACID (Atomicity, Consistency, Isolation, Durability) properties, it may not be as robust in this aspect compared to some other RDBMS.
- Lack of Full-text Searching: MySQL's built-in full-text searching capabilities are not as advanced as some other databases, although this can be supplemented with external tools or plugins.
- Storage Engine Limitations: MySQL's performance and functionality can vary depending on the storage engine used (e.g., InnoDB, MyISAM). Some engines may have limitations in terms of features or scalability.

1.4.3. MongoDB

Figure 24: MôngDB

MongoDB is a NoSQL database management system that stores data in a flexible, JSON-like format called BSON (Binary JSON). It is designed for scalability, flexibility, and high performance, making it suitable for a wide range of applications, from small startups to large enterprises. MongoDB is known for its ease of use, powerful querying capabilities, and ability to handle unstructured and semi-structured data efficiently. Here are some key aspects of MongoDB:

- Document-Oriented: MongoDB stores data in flexible, JSON-like documents, making it easy to represent complex hierarchical relationships and handle semi-structured data.
- Scalability: MongoDB is designed to scale horizontally across multiple servers, allowing it to handle large volumes of data and high traffic loads. It supports sharding, replication, and auto-scaling to distribute data and workload efficiently.
- High Performance: MongoDB provides high-performance read and write operations by using in-memory caching, indexing, and native JSON/BSON storage format. It also supports various query optimization techniques to improve query performance.
- Rich Query Language: MongoDB offers a powerful query language with support for a wide range of query operators, aggregation functions, and indexing options. It allows developers to perform complex queries, aggregations, and data manipulations efficiently.
- Built-in Replication and Failover: MongoDB supports automatic failover and data redundancy through built-in replication features. It can maintain multiple copies of data across different servers to ensure data availability and reliability.
- Community and Support: MongoDB has a large and active community of developers, administrators, and contributors. It offers comprehensive documentation, tutorials, forums, and online resources to help users learn, troubleshoot, and optimize their MongoDB deployments.

Advantages:

- Schema Flexibility: MongoDB is schema-less, which means you can store heterogeneous data types together without a predefined schema. This flexibility is suitable for agile development and evolving data models.
- Scalability: MongoDB can horizontally scale by distributing data across multiple servers. It supports sharding, which allows for automatic partitioning of data across servers, enabling seamless scalability as your data grows.
- High Performance: MongoDB uses internal memory for storing the working set, which results in faster read and write operations. It also supports indexing and aggregation frameworks for efficient querying and data processing.
- Document-Oriented: MongoDB stores data in a JSON-like format called BSON (Binary JSON), which makes it easy to work with for developers accustomed to JavaScript and JSON.
- Community Support and Documentation: MongoDB has a large and active community, providing extensive documentation, tutorials, and resources. This makes it easier for developers to learn and troubleshoot issues.
- Ad Hoc Queries: MongoDB supports ad hoc queries, allowing developers to query data in a flexible and dynamic manner without the need for predefined schema or complex joins.

Disadvantages:

- Eventual Consistency: MongoDB offers eventual consistency by default, which means data consistency is not guaranteed immediately after a write operation. This can lead to inconsistencies in distributed environments and may require additional effort to ensure data integrity.
- Memory Consumption: MongoDB's memory usage can be relatively high, especially when working with large datasets. In-memory operations can consume a significant amount of RAM, potentially leading to increased hardware requirements.
- Data Durability: By default, MongoDB acknowledges write operations after data is written to memory (with the option of writing to disk asynchronously). While this enhances performance, it may compromise data durability in case of sudden failures.
- Complexity of Operations: While MongoDB's flexibility is a strength, it can also introduce complexity, especially in large-scale deployments. Proper schema design, indexing strategies, and understanding of query optimization are crucial for optimal performance.

1.4.4. Compare

	SQL Server	MySQL	MongoDB
Data Model	SQL Server follows a relational data model, where data is organized into tables with rows and columns. It enforces a predefined schema for data integrity.	MySQL also follows a relational data model similar to SQL Server, with support for structured data organized in tables.	MongoDB uses a document-oriented data model, storing data in flexible, JSON-like documents. It allows for schema-less data storage and supports nested data structures.
Query Language	SQL Server uses the Transact-SQL (T-SQL) query language, which is a dialect of SQL with extensions for procedural programming.	MySQL uses standard SQL for querying and data manipulation, with additional features and extensions specific to MySQL.	MongoDB provides a query language similar to SQL, but optimized for querying JSON-like documents. It also supports aggregation pipelines for complex data transformations and aggregations.
Scalability	SQL Server supports vertical scaling, where additional resources are added to a single server instance. It also offers features for horizontal scaling, such as clustering and replication, but with limitations compared to NoSQL databases.	MySQL supports both vertical and horizontal scaling, with features like replication, clustering, and sharding for distributing data across multiple servers.	MongoDB is designed for horizontal scalability, with built-in support for sharding, replication, and auto-scaling. It can handle large volumes of data and high traffic loads efficiently.
Performance	SQL Server is known for its high performance and reliability, especially for transactional workloads. It offers features like indexing, caching, and query optimization for improved performance.	MySQL is optimized for performance and efficiency, with features like in-memory storage engines, query caching, and optimized storage formats.	MongoDB provides high-performance read and write operations, leveraging in-memory caching, indexing, and native JSON/BSON storage format. It offers fast query execution and data retrieval for document-based workloads.

Use Cases	SQL Server is commonly used for traditional relational database applications, such as transaction processing systems, data warehousing, and business intelligence.	MySQL is widely used for web applications, content management systems, e-commerce platforms, and other data-driven applications that require relational database capabilities.	MongoDB is suitable for modern applications with flexible data requirements, such as content management systems, real-time analytics, mobile app backends, and IoT platforms.

Table 4: Compare Database management server

2. Preferred selection justification

After providing some information and making comparisons, in this section, I will give my priorities for Integrated development environment, Programing language, Web framework and Database management server.

For the Integrated development environment, I will choose Visual Studio 2022 for this project. Visual Studio 2022 is a powerful integrated development environment (IDE) that offers a wide range of tools and features suitable for developing the internal training management system for the company. With its diverse set of integrated tools, Visual Studio 2022 enhances productivity and reduces development time. It supports multiple programming languages, including C#, VB.NET, JavaScript, HTML, CSS, and Python, providing flexibility for the development team to choose the most suitable language for the project. Additionally, Visual Studio 2022 integrates seamlessly with ASP.NET Core and Entity Framework, facilitating the development of web applications with robust data management capabilities. Furthermore, Visual Studio 2022 benefits from a large and active community of developers and technical experts, providing extensive documentation, tutorials, and online resources for support. With regular updates and new features, Visual Studio 2022 remains a reliable choice for developing high-quality, scalable, and efficient software solutions for our project.

For the Programing language, I will choose C#. First, because C# integrates strongly with Microsoft technologies like ASP.NET Core, making system development and management easy and efficient. Besides, C#'s high security makes it a safe choice for handling important data in the system. C# is also supported by a rich range of libraries and frameworks, which help reduce development time and increase system flexibility. Not only is it easy to learn and use, C# also receives strong support from the community of programmers and technical experts, thereby helping to solve problems and find solutions when necessary.

For the Web framework, ASP.NET Core MVC was my choice for this project because of its flexibility, high performance, and deep integration with Visual Studio. The platform provides strong security, cross-platform capabilities, and support for Web API development, simplifying application development and deployment. Besides, the support from the large community of ASP.NET Core provides necessary documentation and support to the development team. In short, ASP.NET Core MVC is a reliable choice for building a company's internal training management system.

For the Database management server, I will choose SQL Server for our project. SQL Server is highly reliable and stable, providing features such as transaction control and data recovery, helping to protect important company data. Furthermore, SQL Server provides performance optimization tools and features, including indexing and automatic maintenance scheduling, that help improve query performance and system response time. SQL Server's flexibility and extensibility allow it to support a wide range of applications and technologies, from web applications to business intelligence. With strong security features and compliance with security regulations, SQL Server ensures the safety of corporate data. Finally, SQL Server is also supported by a large and active community, providing documentation and online support to help development teams solve problems and optimize the system.

3. Preferred software development methodology

3.1. Agile

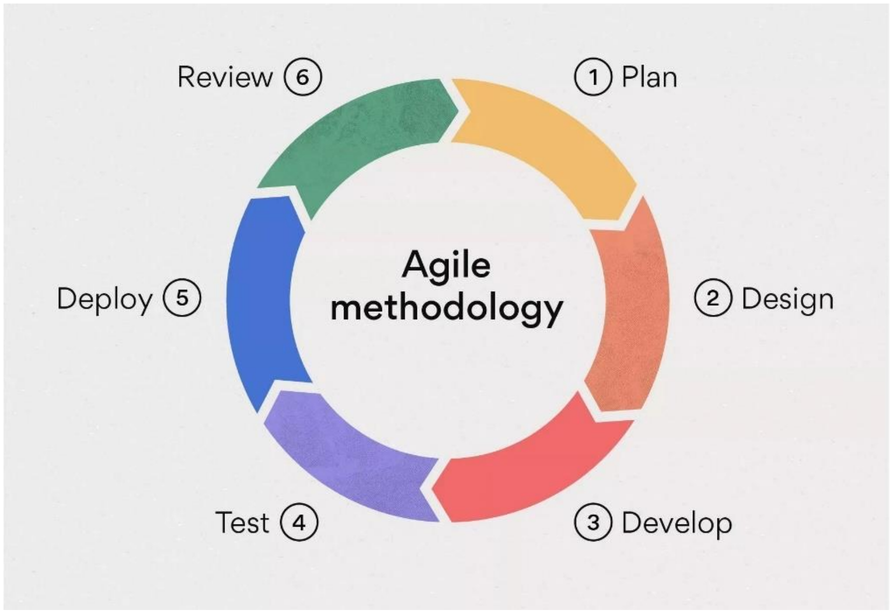


Figure 25: Agile

Agile SDLC model is a combination of iterative and incremental process models with focus on process adaptability and customer satisfaction by rapid delivery of working software product. Agile model believes that every project needs to be handled differently and the existing methods need to be tailored to best suit the project requirements. In Agile, the tasks are divided to time boxes (small time frames) to deliver specific features for a release. Iterative approach is taken and working software build is delivered after each iteration. Each build is incremental in terms of features; the final build holds all the features required by the customer.

The Agile software development life cycle consists of six phases:

- Requirements Gathering: In this Agile model phase, you must define the requirements. The business opportunities and the time and effort required for the project should also be discussed. By analyzing this information, you can determine a system's economic and technical feasibility.
- Design the Requirements: Following the feasibility study, you can work with stakeholders to define requirements. Using the UFD diagram or high-level UML diagram, you can determine how the new system will be incorporated into your existing software system.
- Develop/Iteration: The real work begins at this stage after the software development team defines and designs the requirements. Product, design, and development teams start working, and the product will undergo different stages of improvement using simple and minimal functionality.
- Test: This phase of the Agile Model involves the testing team. For example, the Quality Assurance team checks the system's performance and reports bugs during this phase.
- Deployment: In this phase, the initial product is released to the user.
- Feedback: After releasing the product, the last step of the Agile Model is feedback. In this phase, the team receives feedback about the product and works on correcting bugs based on the received feedback.

Advantages of the Agile model:

- Provides a very realistic approach to software development.
- People and interactions are emphasized rather than process and tools. Customers, developers and testers constantly interact with each other.
- Agile Model in software engineering enables you to draft efficient designs and meet the company's needs.
- Updated versions of functioning software are released every week.
- Changes are acceptable at any time.
- You can reduce the overall development time by utilizing this Agile Model.
- It allows concurrent development and delivery within an overall planned context.
- The final product is developed and available for use within a few weeks.

Disadvantages of the Agile model:

- There is a higher risk of sustainability, maintainability, and extensibility.
- In some corporations, self-organization and intensive collaboration may not be compatible with their corporate culture.
- Documentation and design are not given much attention.
- Without clear information from the customer, the development team can be misled.
- Not a suitable method for handling complex dependencies.

3.2. Waterfall

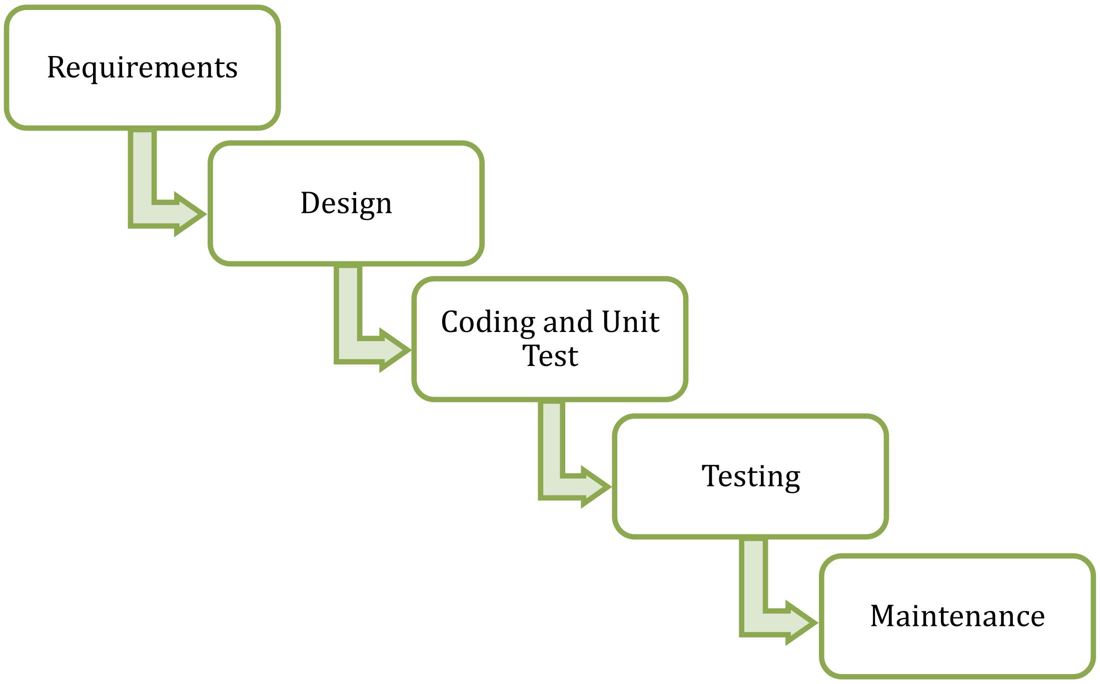


Figure 26: Waterfall

The waterfall model is a linear, sequential approach to the software development lifecycle (SDLC) that is popular in software engineering and product development. The waterfall model uses a logical progression of SDLC steps for a project. It sets distinct endpoints or goals for each phase of development. Development is seen as flowing steadily downwards through the phases of requirements analysis, design, implementation, testing (validation), integration, and maintenance.

When used for a software development process, the waterfall model has seven phases:

- Requirements and Specifications Analysis: is the stage of determining the functional and non-functional Requirements that the software system should have. This phase requires active customer participation and ends with a document known as the SRS (software requirement specification). The Requirement Specification document is the foundation for further activities until the end of the project.
- System Analysis and Design: is the stage of determining how the software system will meet the requirements that the customer requires in the SRS document.
- Coding and Unit Test: is the stage of implementing requirements in the "System Analysis and Design" phase.
- Testing: includes integration testing for groups of components and system testing. A final stage of testing often performed is acceptance testing, with the customer participating in a key role to determine whether the software system meets their requirements.

- Maintenance: this is the installation, configuration and training phase for the customer. This phase fixes software bugs (if any) and develops new changes requested by the customer (such as modifying, adding or removing system functionality/features).

Advantages of the Waterfall model:

- Simple to use and understand.
- Management simplicity thanks to its rigidity: every phase has a defined result and process review.
- Development stages go one by one.
- Perfect for the small or mid-sized projects where requirements are clear and not equivocal.
- Easy to determine the key points in the development cycle.
- Easy to classify and prioritize tasks.

Disadvantages of the Waterfall model:

- The software is ready only after the last stage is over.
- High risks and uncertainty.
- Not the best choice for complex and object-oriented projects.
- Inappropriate for the long-term projects.
- The progress of the stage is hard to measure while it is still in the development.
- Integration is done at the very end, which does not give the option of identifying the problem in advance.

3.3. V-Model

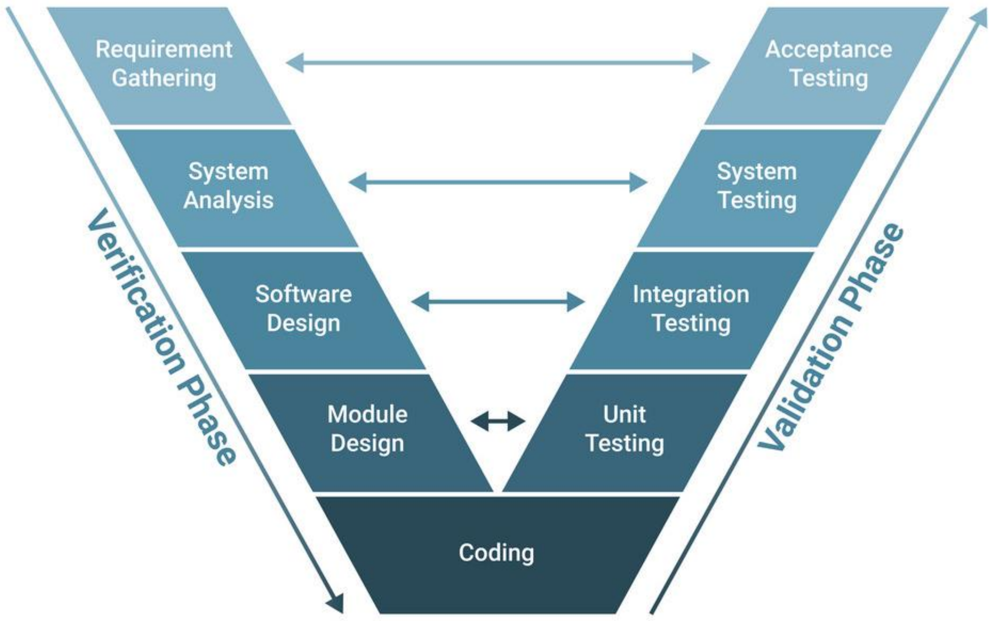


Figure 27: V-Model

The V-model is a type of SDLC model where process executes in a sequential manner in V-shape. It is also known as Verification and Validation model. It is based on the association of a testing phase for each corresponding development stage. Development of each step directly associated with the testing phase. The next phase starts only after completion of the previous phase i.e. for each development activity, there is a testing activity corresponding to it.

More specifically, Verification involves static analysis (review) technique performed without executing code. It is the process of evaluating the product development stage to find out whether specific requirements are met. Validation involves dynamic analysis techniques (functional, non-functional), testing is done by executing code. Validation is the process of evaluating software after the completion of the development phase to determine if the software meets customer expectations and requirements.

Phases of the V-model:

- Requirement Analysis: This phase contains detailed communication with the customer to understand their requirements and expectations. This stage is known as Requirement Gathering.
- System Design: This phase contains the system design and the complete hardware and communication setup for developing product.
- Architectural Design: System design is broken down further into modules taking up different functionalities. The data transfer and communication between the internal modules and with the outside world (other systems) is clearly understood.
- Module Design: In this phase the system breaks down into small modules. The detailed design of modules is specified, also known as Low-Level Design (LLD).
- Unit Testing: Unit Test Plans are developed during module design phase. These Unit Test Plans are executed to eliminate bugs at code or unit level.
- Integration testing: After completion of unit testing Integration testing is performed. In integration testing, the modules are integrated and the system is tested. Integration testing is performed on the Architecture design phase. This test verifies the communication of modules among themselves.
- System Testing: System testing test the complete application with its functionality, inter dependency, and communication.It tests the functional and non-functional requirements of the developed application.
- User Acceptance Testing (UAT): UAT is performed in a user environment that resembles the production environment. UAT verifies that the delivered system meets user’s requirement and system is ready for use in real world.

Advantages of the V-model:

- This is a highly disciplined model and Phases are completed one at a time.
- V-Model is used for small projects where project requirements are clear.
- Simple and easy to understand and use.
- This model focuses on verification and validation activities early in the life cycle thereby enhancing the probability of building an error-free and good quality product.
- It enables project management to track progress accurately.

Disadvantages of the V-model:

- High risk and uncertainty.
- It is not a good for complex and object-oriented projects.
- It is not suitable for projects where requirements are not clear and contains high risk of changing.
- This model does not support iteration of phases.
- It does not easily handle concurrent events.

3.4. Preferred software development methodology: Agile

Choosing the appropriate development model for a company's internal training management project needs to consider factors such as the flexibility of requirements, the complexity of the project, and the importance of having high quality products. early adoptable products, and the organization's readiness to change and collaborate during development. Internal training management projects often have fluctuations in requirements and require flexibility to adapt to changes in company and employee needs. So, an agile development model like Agile is a good choice for this type of project. More specifically, here are some reasons that made Agile the most suitable choice for our project:

First, Agile's flexibility allows us to adapt to frequent fluctuations in requests from departments and employees.

Second, by breaking the project into smaller parts, Agile helps create usable versions of the product early, which improves testing and user feedback.

Third, Agile working methods enhance collaboration and interaction between development team members, as well as between the team and the customer. This helps ensure that the product developed meets the needs of the company and its users.

Ultimately, Agile focuses on delivering real value to customers through continuous development and early product release, while enhancing the ability to adapt to changing environments. With these advantages, Agile is a choice worth considering to ensure internal training management projects are implemented effectively and flexibly.

III. CONCLUSION

In short, I have researched the use of software development tools and techniques and have identified the tools and techniques chosen for application development for this application. Additionally, I have also compared the differences between the various software development tools and techniques studied and demonstrated the preferred choice and preferred software development methodology. The above information and knowledge are provided in this report.

IV. REFERENCES

Visual-paradigm.com. (2024). *What is Unified Modeling Language (UML)?* [online] Available at: https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-uml/ [Accessed 26 Jan. 2024].

Lê Quốc Tuấn (2020). *Tìm hiểu về cách thiết kế Class Diagram*. [online] Viblo. Available at: https://viblo.asia/p/tim-hieu-ve-cach-thiet-ke-class-diagram-L4x5xLyY5BM [Accessed 26 Jan. 2024].

Lucidchart. (2024). *UML Use Case Diagram Tutorial*. [online] Available at: https://www.lucidchart.com/pages/uml-use-case-diagram [Accessed 26 Jan. 2024].

Visual-paradigm.com. (2024). *What is Use Case Diagram?* [online] Available at: https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-use-case-diagram/ [Accessed 26 Jan. 2024].

Lucidchart. (2024). *UML Activity Diagram Tutorial*. [online] Available at: https://www.lucidchart.com/pages/uml-activity-diagram [Accessed 26 Jan. 2024].

Visual-paradigm.com. (2024). *What is Activity Diagram?* [online] Available at: https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-activity-diagram/ [Accessed 26 Jan. 2024].

Visual-paradigm.com. (2024). *What is Component Diagram?* [online] Available at: https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-component-diagram/ [Accessed 26 Jan. 2024].

Lucidchart. (2024). *Component Diagram Tutorial*. [online] Available at: https://www.lucidchart.com/pages/uml-component-diagram [Accessed 26 Jan. 2024].

Gleek.io. (2023). *Diagram maker for developers | Gleek*. [online] Available at: https://www.gleek.io/ [Accessed 30 Jan. 2024].

W3schools.com. (2023). *C# Tutorial (C Sharp)*. [online] Available at: https://www.w3schools.com/cs/index.php [Accessed 13 Feb. 2024].

ardalis (2023). *Overview of ASP.NET Core MVC*. [online] Microsoft.com. Available at: https://learn.microsoft.com/en-us/aspnet/core/mvc/overview?view=aspnetcore-8.0 [Accessed 13 Feb. 2024].

Nguyễn Hưng (2024). *SQL Server là gì? Hướng dẫn cài đặt SQL Server*. [online] Vietnix. Available at: https://vietnix.vn/sql-server-la-gi/ [Accessed 19 Feb. 2024].

W3schools.com. (2024). *Introduction to Java*. [online] Available at: https://www.w3schools.com/java/java_intro.asp [Accessed 19 Feb. 2024].

W3schools.com. (2024). *Introduction to Python*. [online] Available at: https://www.w3schools.com/python/python_intro.asp [Accessed 19 Feb. 2024].

Knowledge Base| Kiến thức Dịch vụ P.A Việt Nam. (2023). *Spring MVC một số kiến thức cơ bản*. [online] Available at: https://kb.pavietnam.vn/spring-mvc-mot-so-kien-thuc-co-ban.html [Accessed 2 Mar. 2024].

Amazon Web Services, Inc. (2020). *Django là gì? - Giải thích về phần mềm Django - AWS*. [online] Available at: https://aws.amazon.com/vi/what-is/django/ [Accessed 2 Mar. 2024].

W3schools.com. (2024). *MySQL Tutorial*. [online] Available at: https://www.w3schools.com/MySQL/default.asp [Accessed 2 Mar. 2024].

W3schools.com. (2024). *MongoDB Tutorial*. [online] Available at: https://www.w3schools.com/mongodb/ [Accessed 2 Mar. 2024].

Hamilton, T. (2023) *Agile model in software engineering*, *Guru99*. Available at: https://www.guru99.com/agile-model.html [Accessed 2 Mar. 2024].

Lutkevich, B. and Lewis, S. (2022) *What is the waterfall model? - definition and guide*, *Software Quality*. TechTarget. Available at: https://www.techtarget.com/searchsoftwarequality/definition/waterfall-model [Accessed 2 Mar. 2024].

Software engineering: SDLC V-model (2022) *GeeksforGeeks*. GeeksforGeeks. Available at: https://www.geeksforgeeks.org/software-engineering-sdlc-v-model/ [Accessed 2 Mar. 2024].