**ASSIGNMENT 1 FRONT SHEET**

|  |  |  |  |
| --- | --- | --- | --- |
| **Qualification** | **BTEC Level 5 HND Diploma in Business** | | |
| **Unit number and title** | **Unit 30: Application Development** | | |
| **Submission date** |  | **Date Received 1st submission** |  |
| **Re-submission Date** |  | **Date Received 2nd submission** |  |
| **Student Name** |  | **Student ID** |  |
| **Class** |  | **Assessor name** |  |
| **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** |  |

**Grading grid**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **P1** | **P2** | **P3** | **M1** | **M2** | **D1** |
|  |  |  |  |  |  |

|  |  |  |
| --- | --- | --- |
| **❒ Summative Feedback: ❒ Resubmission Feedback:** | | |
| **Grade:** | **Assessor Signature:** | **Date:** |
| **Internal Verifier’s Comments:** | | |
| **Signature & Date:** | | |

**Software Requirements specification**

FOR

# <Project>

**Version <X.X>**

**Prepared by**

**Group Name: *<place your group name here>***

|  |  |  |
| --- | --- | --- |
| **<name>** | **<student #>** | **<e-mail>** |
| **<name>** | **<student #>** | **<e-mail>** |
| **<name>** | **<student #>** | **<e-mail>** |
| **<name>** | **<student #>** | **<e-mail>** |

|  |  |
| --- | --- |
| **Instructor:** | ***<place your instructor’s name here>*** |
| **Course:** | ***<place your course name here>*** |
| **Date:** | ***<place the date of submission here>*** |

Table of contents

Introduction (P1)

# Document Purpose

# Product Scope

# Intended Audience and Document Overview

# Definitions, Acronyms and Abbreviations

# References and Acknowledgments

Overall description (P1)

# Product Overview

# Product Functionality

Specific requirements (P1)

# Functional Requirements

# Use Case Model

# Wireflow

Risk assessment (P2)

# Document Purpose

Technologies evaluation (P3)

# Design Tools

## Tools to design UML

## Tools to design user interface

## Conclude which tools will be used for the design of the application

# Front-End technology stack

## Front-end programming language

## JavaScript library / Framework

## CSS framework

## Conclude which Front-end technologies will be used for the development

# Back-End technology stack

## Back-end programming language

## Operating system

## Web server

## Database

## Hosting

## Framework

## Conclude which Back-end technologies will be used for the development

# Tools for source control management

## Git, GitHub, GitLab

### Git

Any large project may be efficiently managed with Git, a distributed version control system. It's open-source and free of cost. Its foundation is the idea of distributed software development, wherein several developers may examine the source code of an application and make changes that are visible to other developers. It gives the user the ability to have "versions" of a project that show the code changes that have been made over time and let them undo those changes on the fly.

The following are a few of Git's key components:

* Distributed system: Distributed systems enable people to collaborate on a project from any location in the globe. Using a version control system, several distant contributors can access a central repository inside a distributed system. One of the most widely used version control systems available today is called Git. It is a distributed system that lets people work on a project concurrently without affecting each other's progress. Following completion of their work, a user pushes their modifications to the repository, whereupon all other remote users who grab the most recent version of the project also receive an update.
* Compatibility: It is possible to utilize Git with any modern operating system. Other version control systems' repositories, such as SVN, CVK, etc., are accessible via Git repositories. Due to Git's ability to emulate a CVS server, Git repositories may be accessed using IDE plugins and existing CVS clients.
* Non-linear development: Git enables users to work remotely on a project from anywhere in the globe. Any portion of the project may be selected by the user, who can then change it after doing the necessary tasks. The Git's non-linear development nature can do this. In addition to offering specialized tools for navigating and visualizing a non-linear development history, Git facilitates quick branching and merging. Git uses a Tree form to document the project's current status. Anytime a new branch is added to the tree, it is further merged with the finished product after it's finished.
* Lightweight: While cloning is occurring, Git keeps all of the data from the central repository on the local repository. The amount of data in the central repository may be enormous since hundreds of people could be working on the same project. Git uses lossless compression, which compresses data and keeps it in a local repository with very little space used up. It uses the opposite method whenever this data is required, which conserves a lot of RAM.
* Speed: It is far more efficient to get data from the local repository rather than the remote repository since Git maintains all project-related data in the local repository via the cloning process. Compared to other version control systems, Git is incredibly quick and scalable, which makes it ideal for managing huge projects. A local repository can retrieve data approximately 100 times faster than a distant server.
* Open-Source: Git is a distributed version control system that can manage any size project quickly and effectively. It is free and open-source. Because it allows users to alter the source code to suit their needs, it is known as open-source. Having an open source nature Git makes it incredibly simple and effective for numerous people to collaborate on the same project at the same time. Git is therefore regarded as the greatest version control system on the market at the moment.

### Github

A for-profit business called GitHub provides cloud-based Git repository hosting. In essence, it makes using Git for version control and collaboration simpler for both individuals and teams. Even inexperienced programmers may benefit from Git because to GitHub's user-friendly interface. Without it, utilizing Git often calls for a higher level of command line proficiency and technical know-how. However, some individuals find GitHub to be so user-friendly that they use it to handle other kinds of projects, including producing novels. Furthermore, anybody may register and host a public code repository for free, which is why open-source projects particularly like GitHub.

Here are a few of GitHub's key features:

* Iteration support: GitHub introduced the iteration field type, which included iteration functionality. Development teams may now see whether project phases are started, in progress, or finished thanks to this functionality. Teams may utilize the iteration assistance to improve the accuracy of their future sprint planning as they refine their work. The iteration type improves code structure and saves time when forming new sprints. The most significant addition to iteration support is the ability to build all of the sprints for a release at once, instead of having to repeat the procedure for every sprint. In addition to saving developers time, this gets rid of a time-consuming administration operation that needs to be done every sprint.
* Command Palette navigation control: Developers may access a multitude of functions with GitHub's Command Palette navigation tool, including the ability to search and execute commands from any location within the platform, receive suggestions on-demand based on context and resources utilized, and have suggestions automatically updated and improved with the addition of content. Developers have the ability to personalize keyboard shortcuts for opening and utilizing the Command Palette.
* Codespaces: Codespaces is a cloud-hosted development environment that increases the number of repeated configuration possibilities and facilitates code development. Codespaces allow teams to personalize their work, even down to the establishment of a reusable project setup. Codespaces can connect locally from Visual Studio (VS) Code or remotely via a browser. It can be configured on two to thirty-two virtual machine (VM)-based servers.
* Customizable fields: Teams may define fields on the platform and plan and track work using customisable fields. GitHub projects automatically keep track of pull requests, track issues, and save notes and ideas. Team members may also access and share data within projects using custom fields. Within a project, custom fields are located in the sidebar of problems or pull requests. Text, numbers, dates, single select, and iteration fields are the only kind of custom fields that are permitted.
* Sort and group issues or pull requests: Utilizing project boards and dynamic tables, GitHub also offers a helpful function for grouping and sorting pull requests and problems. Keyboard shortcuts and the drag-and-drop functionality allow users to rearrange, shift cards within columns, or alter the order of columns completely.
* Copilot: Code comments may be turned into code using GitHub Copilot's AI. Copilot suggests possible coding alternatives based on the code comment input. Subsequently, developers have the option to adopt a proposal made by AI or modify it. And with a single comment, Copilot may develop whole functions. For tasks like pair programming, completing code conversion projects, or assisting development teams, this GitHub functionality could be helpful. Teams have to think about the feature and determine whether it might save money and enhance the code.

### Gitlab

For large-scale DevOps and DevSecOps projects, GitLab is an open source code repository and collaborative software development platform. GitLab provides an online code storage place as well as CI/CD and issue tracking features. Users may examine older code and roll back to it in case of unanticipated issues, and the repository facilitates hosting many development chains and versions. GitLab and GitHub are competitors as they employ the same Git version control system and function similarly when it comes to source code management.

The primary features that set GitLab apart are as follows:

* Continuous integration capabilities: GitLab's strong continuous integration and delivery (CI/CD) capabilities are a noteworthy feature. It enables programmers to automate application construction, testing, and deployment, resulting in quicker software release cycles and higher-quality software. Many development teams favor GitLab because of its highly configurable CI/CD pipelines, which can be set up to match unique project requirements.
* Easily package and distribute applications: The usage of Docker, a containerization platform that streamlines the deployment process, is another noteworthy aspect of GitLab. GitLab comes with a built-in registry that may be used right away without any further setup. Additionally, GitLab offers a full infrastructure management solution by managing third-party servers for the deployment of Docker images.
* Simplify the deployment process: Kubernetes is a well-liked container orchestration tool, and GitLab integrates with it seamlessly. GitLab is an ideal platform for Kubernetes integration because to its well-designed Kubernetes structure. Development and quality assurance teams may work together more easily thanks to this connection, which also makes it possible for each branch to be deployed efficiently.
* Diverse expansion tools: GitLab offers a wide range of flexible tools relevant to project management. A versatile workflow management system is provided by GitLab boards, which include features like CSV export preparation, group milestones, burnup and burndown charts, targeted dates for problems, and the ability to transfer issues across projects. In addition, it supports quick code export to other projects and can import larger projects than GitHub.
* Smart integrated monitoring system: Users may describe the consequences of merging branch and production systems by collecting performance metrics for servers and apps using GitLab's integrated monitoring system. With this monitoring feature, users are guaranteed total visibility and control over their infrastructure and apps.
* Strong security measures: To safeguard code and data, GitLab also offers strong security features. It provides role-based access control, integrated authentication, and external identity provider integration. In order to detect and reduce any security threats, it has tools including code scanning, vulnerability management, and container scanning.

## Conclude which tools will be used for the development

Our decision to adopt Git and GitHub for web application development was made in light of the advantages mentioned above.

Several developers can work together on a project by monitoring code changes using Git, a distributed version control system. Features like controlling code versions, branch creation and merging, and conflict resolution are all included.

GitHub is a cloud-based Git repository hosting service. In addition to project management and teamwork, it offers extra capabilities like pull requests, problem tracking, and code review tools.

While GitLab is a cloud storage service that is comparable to Github, it is not as well-known for the following reasons:

* GitLab faces stiff competition from GitHub as a late entrant into the market.
* Despite its widespread use, GitLab could not get much support from a range of institutions, including major software firms like GitHub.
* A bigger community ecosystem, with lots of open source projects and integrations, may be found on GitHub. Developers are drawn to platforms with a larger number of users and projects as a result, creating a network effect.

Thus, Git and GitHub offer a strong and dependable combination of tools for project management, code reviews, issue tracking, version control, and web application development management. This combination would offer all the capabilities and tools required to create a scalable and high-quality web application.

# Software development models

## Introduce several SDLC models

### Scrum

Sprints, or iterations of a project, are grouped together into projects using the Scrum agile project management methodology. Although the Product Owner prioritizes work and interacts with stakeholders, the Scrum Team bears the responsibility of providing value. Scrum can produce work rapidly and test it because to its iterative methodology. Scrum Project Management, to put it briefly, is a methodology that places an emphasis on incremental and iterative product delivery through ongoing collaboration and feedback.

Advantages of the Scrum model:

* Teams may accomplish project deliverables more quickly and effectively by using Scrum.
* Effective utilization of time and resources is ensured by Scrum.
* Big projects are broken up into smaller sprints.
* Codes and tests are conducted during the sprint review, which is ideal for projects with rapid development pace.
* The scrum meetings provide the team with clear visibility.
* Scrum is agile and welcomes input from stakeholders and consumers.
* Short sprints make it much easier to make adjustments in response to feedback.
* During daily scrum meetings, each team member's unique contribution is evident.

Disadvantages of the Scrum model:

* Because scrum lacks a set completion date, scope creep is frequently the result.
* Failure to cooperate or be highly dedicated by individuals increases the likelihood of a project failing.
* It's difficult to implement Scrum in large teams.
* The framework needs seasoned team members to succeed.
* Team members might become frustrated with daily meetings.
* A team member's departure in the middle of a project might have a devastating effect on it.
* Implementing quality is challenging until the team goes through a rigorous testing procedure.

### Waterfall

First introduced as a Process Model, it was the Waterfall Model. Another name for it is a linear-sequential life cycle model. It's really easy to use and comprehend. A waterfall paradigm requires that each step be finished completely before moving on to the next. It is well renowned for software development projects with precise and well-defined specifications. When the final objective is well-defined and the software's required functionality is understood, a linear, sequential method functions well.

Advantages of the Waterfall model:

* Every stage of development needs to be finished before moving on to the next.
* Suitable for smaller projects with clearly specified needs.
* Prior to finishing each step, they ought to conduct quality assurance tests (Verification and Validation).
* At each stage of the software development cycle, comprehensive documentation is produced.
* Project team is fully responsible for the project, with little involvement from the client.
* Any modifications to the program are done as it is being developed.

Disadvantages of the Waterfall model:

* For complicated projects with regularly changing requirements, it is not ideal.
* The testing phase of the development process occurs quite late.
* For developers and testers, documentation takes up a lot of time.
* The useful feedback from clients cannot be included into the continuing development process.
* Numerous issues might occur from minor adjustments or mistakes in the finished product.

### V-model

One kind of SDLC model is the V-model, in which the process is carried out in a V-shape and in a sequential fashion. The Verification and Validation model is another name for it. Its foundation is the assignment of a testing phase to every related development step. Every step's development is closely related to the testing stage. The subsequent phase does not begin until the preceding phase is finished, i.e., there is a testing activity for every development activity.

Advantages of the V-model:

* Straightforward and simple to use.
* Testing preparation and design processes take place far in advance of coding. This is quite time-saving. greater likelihood of success compared to the waterfall methodology.
* Early defect detection is achieved by proactive defect tracking.
* prevents the faults from spiraling downhill.
* Functions well for small projects with clear needs.

Disadvantages of the V- model:

* Least bendable and extremely stiff.
* Software is not designed in advance of implementation; instead, it is developed during that process.
* The requirement documents and test documents must be updated if there are any modifications made in the middle.

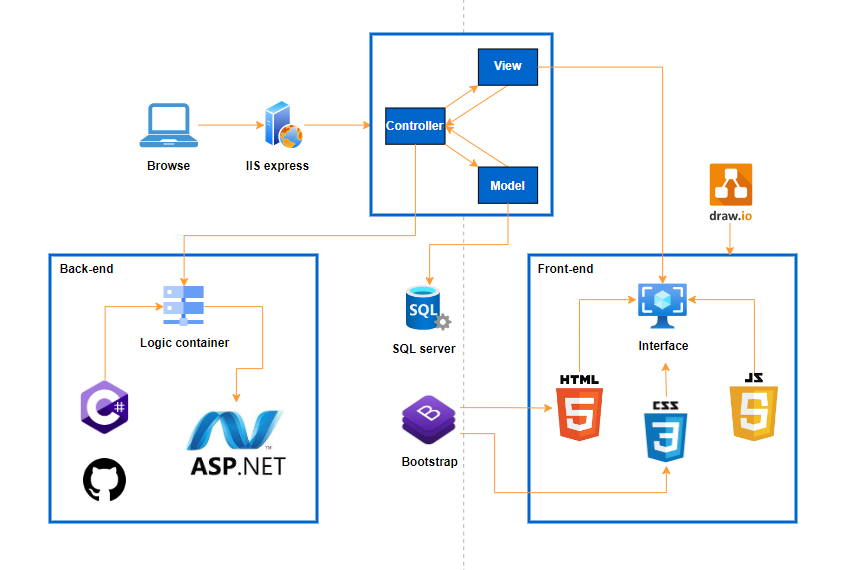
## Conclude which SDLC will be used to develop the system

Software development life cycle (SDLC) models such as Scrum, Waterfall, and V-model describe the phases and procedures involved in creating and completing software projects. Of them, the Waterfall approach is frequently seen to be the most appropriate for developing web applications because of the following reasons:

* Well-defined requirements: Because of its sequential stages, the Waterfall approach minimizes scope creep and ambiguity by guaranteeing that all criteria specified in the first stage are fulfilled before moving on.
* Predictability: The Waterfall model facilitates improved project planning and estimate because it employs a sequential methodology with distinct deliverables and milestones. This consistency comes in handy when operating under strict budget or timeline restrictions.
* Documentation: It is simpler to measure progress, monitor modifications, and guarantee accurate documentation throughout the whole development process when using the Waterfall approach, which places a strong focus on documentation at every stage. For web application initiatives that could eventually need upgrades or maintenance, it is crucial.
* Stakeholder involvement: Stakeholder participation is encouraged early in the process by the Waterfall approach, particularly during the requirements collection and design phases. By being involved, the development team and stakeholders can better align and meet the needs and expectations of the latter.

Although the Waterfall approach might offer structure and predictability, it might not be as adaptable or flexible as the Scrum model, which works better for projects with surroundings that change over time or with rising requirements. The field is evolving quickly. The Waterfall approach is still a common option, nevertheless, for web application projects where reliability and well-defined specs are crucial.

# Illustrate all your findings on how to use these by drawing the overview



References

Anon (2019) Git Features, GeeksforGeeks, GeeksforGeeks, [online] Available at: https://www.geeksforgeeks.org/git-features/ (Accessed 28 November 2023).

Anon (2023) What Is GitHub? A Beginner’s Introduction to GitHub, Kinsta®, [online] Available at: https://kinsta.com/knowledgebase/what-is-github/ (Accessed 28 November 2023).

Reichert, A. (2022) 7 essential GitHub features for dev, project management, Software Quality, TechTarget, [online] Available at: https://www.techtarget.com/searchsoftwarequality/tip/Evaluate-GitHub-features-for-dev-and-project-management (Accessed 28 November 2023).

TechTarget Contributor (2020) GitLab, WhatIs.com, TechTarget, [online] Available at: https://www.techtarget.com/whatis/definition/GitLab (Accessed 28 November 2023).

Anon (2020) What is GitLab | Features and Impotance of GitLab, EDUCBA, [online] Available at: https://www.educba.com/what-is-gitlab/ (Accessed 28 November 2023).

Vasiliauskas, V. (2022) 14 Scrum Advantages and Disadvantages in 2023, Teamhood, [online] Available at: https://teamhood.com/agile/scrum-advantages-disadvantages/ (Accessed 28 November 2023).

Chandana (2013) Scrum Project Management: Advantages and Disadvantages, Simplilearn.com, Simplilearn, [online] Available at: https://www.simplilearn.com/scrum-project-management-article (Accessed 28 November 2023).

Anon (2023) What is Waterfall Model in SDLC? Advantages and Disadvantages, Guru99, [online] Available at: https://www.guru99.com/what-is-sdlc-or-waterfall-model.html (Accessed 28 November 2023).

Anon (2022) What is Waterfall model- Examples, advantages, disadvantages & when to use it?, Tryqa.com, [online] Available at: https://tryqa.com/what-is-waterfall-model-advantages-disadvantages-and-when-to-use-it/ (Accessed 28 November 2023).

Anon (2023) Advantages and Disadvantages of Waterfall Model in SDLC, EDUCBA, [online] Available at: https://www.educba.com/advantages-and-disadvantages-of-waterfall-model/ (Accessed 28 November 2023).

Anon (2020) What is V-model- advantages, disadvantages and when to use it?, Tryqa.com, [online] Available at: https://tryqa.com/what-is-v-model-advantages-disadvantages-and-when-to-use-it/ (Accessed 28 November 2023).

Anon (2018) SDLC V Model Software Engineering, GeeksforGeeks, GeeksforGeeks, [online] Available at: https://www.geeksforgeeks.org/software-engineering-sdlc-v-model/ (Accessed 28 November 2023).