ITS Documentation

High Level Design Document – V 1.1

Table of Contents

[1. Overview 4](#_Toc128544201)

[2. Context 4](#_Toc128544202)

[2.1 Abbreviations & Definitions 4](#_Toc128544203)

[2.2 References 5](#_Toc128544204)

[Low-Level Design Document 5](#_Toc128544205)

[Markdown syntax for wikis 5](#_Toc128544206)

[Syntax guidance for basic Markdown usage 5](#_Toc128544207)

[Markdown tools 5](#_Toc128544208)

[Azure DevOps Integration Documentation Project 5](#_Toc128544209)

[Integration Documentation Web Site 5](#_Toc128544210)

[3. Goals and Non-Goals 6](#_Toc128544211)

[4. Existing Solution 9](#_Toc128544212)

[5. Proposed Solution 9](#_Toc128544213)

[6. Alternative Solutions 9](#_Toc128544214)

[7. Open questions, items to follow-up 16](#_Toc128544215)

Table of Figures

[Figure 1 - Markdown file sample 6](#_Toc128732981)

[Figure 2 - Markdown file can contain complex diagrams, formulas, videos, and other documents. 7](#_Toc128732982)

[Figure 3 - Sample documentation project created and deployed in an Azure App Service during POC 8](#_Toc128732983)

[Figure 4 - PDF file opened from inside a documentation’s page. 8](#_Toc128732984)

[Figure 5 - Select a project from Nodinite to view its documentation. 10](#_Toc128732985)

[Figure 6 - Documentation page displayed in Nodinite. 11](#_Toc128732986)

[Figure 7 - Central project's structure 12](#_Toc128732987)

[Figure 8 - Deployment alternative 1 13](#_Toc128732988)

[Figure 9 - Deployment alternative 2 14](#_Toc128732989)

[Figure 10 - Deployment alternative 3 15](#_Toc128732990)

**Document Version Control**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Document Version | Action | Author | Date | Changes |
| 1.0 | Create | Eugen Frunza |  |  |
|  |  |  |  |  |

**Reviewers**

|  |  |
| --- | --- |
| Name | Title |
|  |  |
|  |  |

# Overview

The present design describes the process of adding documentation to the ITS projects.

At the same time, it defines the process of creating and deploying the documentation added to each ITS project to a central documentation web site that will store the overall documentation made available by the ITS department to its users.

The ITS documentation will be consumed by developers, Business Analysts and Project Managers belonging to the ITS department, as well as other employees from Bow Valley College.

# Context

The documentation for the projects implemented by the ITS department is stored in SharePoint and in several folders in MS Teams. To find the documentation artifacts that define the business requirements, the designs, the implementation, and the testing details of a project, one needs to ask for the appropriate documents stored in several places and displayed in various formats.

At the same time, the developers do not have a unified and versatile way to create, display and publish their internal documentation.

## Abbreviations & Definitions

Table 2: Abbreviations and definitions

|  |  |
| --- | --- |
| **Abbreviation** | **Definition** |
| HLD | High Level Design |
| LLD | Low-Level Design |
| Markdown | A lightweight markup language that you can use to add formatting elements to plaintext text documents |
| Ruby | A dynamic, open-source programming language with a focus on simplicity and productivity. |
| Jekyll | Ruby script for building static web sites from simple HTML and markdown content using templates and add-ons. |

## 

## References

Table 3: References

|  |  |
| --- | --- |
| **Reference** | **Location** |
| Low-Level Design Document | [ITS Central Documentation LLD.docx](https://bowvalley.sharepoint.com/:w:/s/DevelopmentTeam/EWE304nwdalOqjovV_JT9LgBkNaksBFWrJD_Mbus-rQJHA?e=QzYLYq) |
| Markdown syntax for wikis | <https://learn.microsoft.com/en-us/azure/devops/project/wiki/wiki-markdown-guidance?source=recommendations&view=azure-devops> |
| Syntax guidance for basic Markdown usage | <https://learn.microsoft.com/en-us/azure/devops/project/wiki/markdown-guidance?view=azure-devops> |
| [Markdown tools](#_Markdown_tools) | <https://www.markdownguide.org/tools/> |
| Azure DevOps Integration Documentation Project |  |
| Integration Documentation Web Site | <https://wa-integrationdocs-dev-001.azurewebsites.net/> |

# Goals and Non-Goals

The goal is to create the documentation required for any ITS project in a common way, using tools easy to use that allows to incorporate and display all type of artifacts (documents, images, diagrams, videos) and design a process to create and deploy the documentation to a centralized web site.

The documentation will be written as Markdown files that uses a simple syntax as described in the links from the [References](#_References) section.

Diagram

Description automatically generated

Figure 1 - Markdown file sample

Timeline

Description automatically generated

Figure 2 - Markdown file can contain complex diagrams, formulas, videos, and other documents.

One other goal is to define a process which consolidates the documentation for all the individual projects into a central project that is deployed as a stand-alone web site controlled by Role Base Access Security, with minimal effort from the developer’s side.

Graphical user interface, application

Description automatically generated

Figure 3 - Sample documentation project created and deployed in an Azure App Service during POC

Diagram

Description automatically generated

Figure 4 - PDF file opened from inside a documentation’s page.

# Existing Solution

There is no current solution that provides a unified way of creating and publishing the documentation for each ITS project as well as consolidating this documentation in a central web site.

# Proposed Solution

The solution proposes the creation of a o documentation project that will build and run the markdown files as HTML.

The project can be created using a variety of tools and runtimes as described in the LLD.

The solution we have explored in the LLD uses Ruby and Jekyll to build, run and deploy the documentation’s project.

The project can be downloaded, built, and run locally so the developers can either modify or add additional documentation to it or change the look and feel of the documentation’s project web site.

# Alternative Solutions

There are several tools available to build the markdown files/project and build the html static site as described in the LLD.

(More information about this can be found at this link: [Markdown tools](#_Markdown_tools)

Each of these tools uses a different runtime to write the project and build the local environment / site (Ruby, Python, NodeJs).

Each tool uses a different UI format (theme) to display the content (MkDocs, Docusaurus, ReadTheDocs / Sphinx, Jekyll, etc.)

Each of these tools has some limitations in terms of what complex markup they can handle.

Each of these tools has a theme used to generate the documentation’s site UI layer and some of these themes allow for easier use and provide more options when adding content to the site.

|  |  |  |
| --- | --- | --- |
| Alternative explored | Pros | Cons |
| Ruby & Jekyll | Used in the POC and well documented in the LLD. | May have some limitations of what parts of markdown is able to display. |
| MkDocs | Must confirm if the web site UI provides more functionality and it is easier to use |  |
| ReadTheDocs / Sphinx | Well documented in a Pluralsight course - <https://app.pluralsight.com/library/courses/build-contribute-documentation-git-based-workflow/table-of-contents> |  |

**To Do: Research and document above the limitations and ease of use of each solution’s UI layer.**

One other alternative considered was to create markdown documentation in **Nodinite**.

**Nodinite** is alogging tool used in the development of the ITS applications which can incorporate markdown documentation inside the projects.

Zingee Ngan assessed the capabilities of this tool to see if it can achieve the goals mentioned by this HLD.

More technical details about the Nodinite implementation can be found in the LLD.

Graphical user interface, text, website

Description automatically generated

Figure 5 - Select a project from Nodinite to view its documentation.

Graphical user interface, text, application, email

Description automatically generated

Figure 6 - Documentation page displayed in Nodinite.

|  |  |
| --- | --- |
| Pros | Cons |
| Easy to use. | Documentation it is linked to Nodinite and created per project. |
| The documentation will be picked up and deployed into Nodinite by the pipeline. | Does not have the capability to provide a centralize web site focused on searching any document in the documentation’s web site. |
| Developers do not need to be concerned about the pipeline deployment steps and concentrate just on creating documentation files. | Pipeline setup can be costly especially when we have several projects, folders and files per project and the project’s structure it is not standardized or keeps changing. |

The documentation can be deployed as:

* Azure Wiki documentation,
* Azure static web site as part of an Azure Storage account
* As a standalone web site in Azure App Services.

All these 3 deployment alternatives have been described in the LLD document.

|  |  |  |
| --- | --- | --- |
| Alternative explored | Pros | Cons |
| Azure Wiki | Simplest alternative, working well for publishing documentation per individual project. | Accessed via the DevOps project and not as a standalone web site. |
| Azure static web site | Simple Html web site. | Permission via signed URIs shared with the users. |
| Standalone web site in Azure App Service | Standalone web site providing custom look and feel, granular access permission. | Must centralize documentation from individual projects. |

When centralizing the individual projects documentation, a certain folder structure will have to be respected by the developers for their individual project.

A process will be defined inside the Azure DevOps pipelines to merge all the individual project’s folders into a central folder structure.

Chart, schematic, bar chart

Description automatically generated

Figure 7 - Central project's structure

To create a central documentation web site, we have explored several alternatives:

* Create a single project for the whole documentation site which will be downloaded locally by the developers, having files modified and checked in the repository. Upon the files being checked in, the pipeline will trigger a build and the central web site gets updated.

Diagram

Description automatically generated

Figure 8 - Deployment alternative 1

|  |  |
| --- | --- |
| Pros | Cons |
| Simplest alternative, with no need to create central documentation project and pipeline. This is the solution from the POC, documented in the LLD. | Developers will have to get familiar with the web site runtime (Ruby, Python, etc.) |
| It is each developer responsibility to maintain the project. | It is each developer responsibility to maintain the project. |

* Have each developer add documentation local to their ITS project they are working on. Create a central documentation project and build a pipeline which copies the relevant documentation from each individual project, build it and deployed it to the central documentation web site. Schedule the process to run nightly.

Diagram

Description automatically generated

Figure 9 - Deployment alternative 2

|  |  |
| --- | --- |
| Pros | Cons |
| Developers will concentrate only on their own project documentation | The central project requires permission to individual projects to run its pipeline. |
| Files are not copied in the central repository, but they are used by the pipeline. No duplicated files. | Complex pipeline to build and maintain. |
|  | Someone else needs to have the ownership of this project and perform the configuration. |

This alternative has been researched in the POC and documented in detail in the LLD.

* Have each developer add documentation local to their ITS project they are working on. Upon adding the modified documentation file into their repository, each individual project’s pipeline will send their changes to a central repository and trigger a build for the central documentation’s project which in turn will build and deploy to the central web site.

Diagram, schematic

Description automatically generated

Figure 10 - Deployment alternative 3

|  |  |
| --- | --- |
| Pros | Cons |
| Developers will concentrate only on their own project documentation | Individual projects may require permission to the central DevOps project and repo. |
| A central pipeline will still have to be created but it will have a predefined structure where the individual projects will merge their own folders and files. The central content will get created dynamically, without the need for the central project to have intimate knowledge about the individual projects. | Must investigate if the modified files need to be copied in the central repository. |
|  | Still requires a certain amount of work to be done on both the individual and central projects’ pipelines. |