Nottingham Trent University

School of Science and Technology

**Breaking Down Silos: Developing a Knowledge Sharing Platform for Internal Business**

by

Callum White

in

2023

Project report in part fulfilment

of the requirements for the degree of

Bachelor of Science with Honours

Digital & Technology Solutions Professional

Software Engineer

I hereby declare that I am the sole author of this report. I authorise the Nottingham Trent University to lend this report to other institutions or individuals for the purpose of scholarly research.

This work is covered for confidentiality in accordance with the contract signed by Nottingham Trent University and my employer and should not be reproduced without permission of my employer.

Signature

Callum John White

Abstract

The fast-paced and competitive nature of modern business environments can cause a lack of communication and collaboration due to silos, which form between departments and teams. This can lead to inefficiencies, including missed opportunities and decreased innovation. In response to this problem, we developed a knowledge-sharing platform for internal businesses.

To achieve this, we conducted market research on existing knowledge-sharing platforms, including interviews to identify employee needs. Additionally, we designed and developed a user-friendly platform to enable employees to share and access knowledge across departments and teams. The platform included features such as discussion forums, document sharing, and a searchable database of employee expertise.

Our findings indicated the knowledge-sharing platform was successful in breaking down silos, as well as improving communication and collaboration across the organisation. Employees reported increased efficiency, improved decision-making, and a greater sense of shared purpose. Additionally, the platform helped to identify areas where further training and resources are needed, leading to improvements in the organisation’s overall performance.

In conclusion, the internal implementation of knowledge-sharing platforms is an effective solution to break down silos and improve department and team collaboration. By leveraging employee expertise, organisations can achieve greater success and innovation in ever-changing business environments.

***Acknowledgements***

University Mentor – Thomas Johnson

Workplace Mentor #1 – Hannah Jeacock

Workplace Mentor #2 – Kate Featherstone

Table of Contents

[Abstract ii](#_Toc137393123)

[Table of Contents iv](#_Toc137393124)

[List of Figures xii](#_Toc137393125)

[List of Tables xiv](#_Toc137393126)

[CHAPTER 1 1](#_Toc137393127)

[Introduction 1](#_Toc137393128)

[1.1 Introduction 1](#_Toc137393129)

[1.2 Opportunity 3](#_Toc137393130)

[1.3 Gap Analysis 4](#_Toc137393131)

[1.4 Feasibility Analysis 4](#_Toc137393132)

[1.4.1 Operational 5](#_Toc137393133)

[1.4.2 Technical 5](#_Toc137393134)

[1.4.3 Schedule 5](#_Toc137393135)

[1.4.4 Economic 5](#_Toc137393136)

[1.4.5 Feasibility outcome 6](#_Toc137393137)

[1.5 Risk Analysis 6](#_Toc137393138)

[1.6 Aim & Objectives 7](#_Toc137393139)

[CHAPTER 2 8](#_Toc137393140)

[Background 8](#_Toc137393141)

[2.1 Introduction 8](#_Toc137393142)

[2.2 Technical literature review 8](#_Toc137393143)

[2.3 Team on MS Teams 9](#_Toc137393144)

[2.3.1 Limitations 10](#_Toc137393145)

[2.4 Dedicated SharePoint site 10](#_Toc137393146)

[2.4.1 Limitations 10](#_Toc137393147)

[2.5 Miro board 11](#_Toc137393148)

[2.5.1 Limitations 12](#_Toc137393149)

[2.6 Nuclino 12](#_Toc137393150)

[2.6.1 Limitations 13](#_Toc137393151)

[2.7 Gap in Existing Solutions 14](#_Toc137393152)

[2.8 Similar Tools/Software/Approaches 14](#_Toc137393153)

[2.9 Commercial Context 15](#_Toc137393154)

[2.10 Research Studies 15](#_Toc137393155)

[2.10.1 Deloitte's knowledge-sharing platform 15](#_Toc137393156)

[2.10.2 Knowledge Sharing in Digital Platform Ecosystems: A Textual Analysis of SAP's Developer Community 17](#_Toc137393157)

[2.10.3 Social Media for Knowledge-Sharing: A Systematic Literature Review 18](#_Toc137393158)

[2.10.4 Comparison 19](#_Toc137393159)

[2.11 Summary 20](#_Toc137393160)

[2.12 Analysis of Legal, Social, Ethical and Professional Issues 21](#_Toc137393161)

[2.12.1 Legal Issues 21](#_Toc137393162)

[2.12.2 Social Issues 22](#_Toc137393163)

[2.12.3 Ethical Issues 23](#_Toc137393164)

[2.12.4 Professional Issues 23](#_Toc137393165)

[CHAPTER 3 25](#_Toc137393166)

[Project Definition 25](#_Toc137393167)

[3.1 Introduction 25](#_Toc137393168)

[3.2 Aims & Objectives 25](#_Toc137393169)

[3.2.1 Objectives 25](#_Toc137393170)

[3.3 Requirement Analysis 26](#_Toc137393171)

[3.3.1 Research and Discovery 26](#_Toc137393172)

[3.4 Software Development Methodologies 28](#_Toc137393173)

[3.4.1 Waterfall 28](#_Toc137393174)

[3.4.2 Agile/Scrum 28](#_Toc137393175)

[3.4.3 Chosen Development Method 29](#_Toc137393176)

[3.5 Functional and Non-Functional requirements 30](#_Toc137393177)

[3.5.1 Functional Requirements 31](#_Toc137393178)

[3.5.2 Non-functional Requirements 32](#_Toc137393179)

[3.5.3 Conclusion 33](#_Toc137393180)

[3.6 Success Criteria 33](#_Toc137393181)

[3.7 Use Case Analysis 34](#_Toc137393182)

[3.7.1 Use Case #1 35](#_Toc137393183)

[3.7.2 Use Case #2 36](#_Toc137393184)

[3.7.3 Use Case #3 37](#_Toc137393185)

[3.8 MoSCoW 38](#_Toc137393186)

[3.8.1 Must-have: 38](#_Toc137393187)

[3.8.2 Should-have: 38](#_Toc137393188)

[3.8.3 Could-have: 38](#_Toc137393189)

[3.8.4 Won't-have: 39](#_Toc137393190)

[3.9 Project management 39](#_Toc137393191)

[3.10 Scope 39](#_Toc137393192)

[3.11 Schedule Plan 40](#_Toc137393193)

[3.12 Resource Planning 41](#_Toc137393194)

[3.13 Gantt Chart & PERT Diagram 42](#_Toc137393195)

[3.14 Risk Assessment 43](#_Toc137393196)

[3.15 Azure DevOps 44](#_Toc137393197)

[3.16 Cost Model (COCOMO) 44](#_Toc137393198)

[CHAPTER 4 47](#_Toc137393199)

[Design 47](#_Toc137393200)

[4.1 Introduction 47](#_Toc137393201)

[4.2 High-level architecture design 47](#_Toc137393202)

[4.2.1 Web application framework 49](#_Toc137393203)

[4.2.2 Open ID Single Sign On (SSO) 50](#_Toc137393204)

[4.2.3 Azure container instances 50](#_Toc137393205)

[4.3 Use Case diagram 52](#_Toc137393206)

[4.4 Graph Database Design 53](#_Toc137393207)

[4.4.1 Node Properties 54](#_Toc137393208)

[4.5 Class Diagrams 56](#_Toc137393209)

[4.5.1 Domain Class Diagram 56](#_Toc137393210)

[4.5.2 Service Orientated Class Diagram 57](#_Toc137393211)

[4.6 UI Design 57](#_Toc137393212)

[4.7 Test Plans 63](#_Toc137393213)

[CHAPTER 5 64](#_Toc137393214)

[IMPLEMENTATION 64](#_Toc137393215)

[5.1 Introduction 64](#_Toc137393216)

[5.2 Coding Standards 64](#_Toc137393217)

[5.2.1 SOLID 64](#_Toc137393218)

[5.2.1 Design Patterns 65](#_Toc137393219)

[5.3 NET Project Layout (Layered Architecture) 66](#_Toc137393220)

[5.4 Automated Deployment to Azure 68](#_Toc137393221)

[5.4.1 Github Workflow 69](#_Toc137393222)

[5.5 Test Driven Development 70](#_Toc137393223)

[5.5.1 Benefits of Test-Driven Development 70](#_Toc137393224)

[5.5.2 Using NUnit in C# to Create Tests 71](#_Toc137393225)

[5.6 Development Iterations 71](#_Toc137393226)

[5.6.1 Sprint 1 72](#_Toc137393227)

[5.6.2 Sprint 2 77](#_Toc137393228)

[5.6.3 Sprint 3 82](#_Toc137393229)

[5.7 Sprint Summary 90](#_Toc137393230)

[5.8 UX Feedback 91](#_Toc137393231)

[5.9 Unit & Integration Tests 92](#_Toc137393232)

[5.9.1 Integration Test Setup 92](#_Toc137393233)

[5.9.2 Unit & Integration Test Results 93](#_Toc137393234)

[CHAPTER 6 95](#_Toc137393235)

[RESULTS 95](#_Toc137393236)

[6.1 Introduction 95](#_Toc137393237)

[6.2 Survey Feedback 95](#_Toc137393238)

[6.2.1 SurveyMonkey Feedback 97](#_Toc137393239)

[6.2.2 Survey Feedback Summary 103](#_Toc137393240)

[6.2.3 In-depth Feedback 104](#_Toc137393241)

[6.2.4 Overall Feedback 105](#_Toc137393242)

[6.3 Testing 106](#_Toc137393243)

[6.4 Gap Analysis Review 106](#_Toc137393244)

[6.5 Success Criteria 107](#_Toc137393245)

[6.6 Hosting Cost 110](#_Toc137393246)

[CHAPTER 7 112](#_Toc137393247)

[CONCLUSIONS / FUTURE WORK 112](#_Toc137393248)

[7.1 Conclusions 112](#_Toc137393249)

[7.1.1 Project Conclusion 112](#_Toc137393250)

[7.1.2 Project PSEL Issues 114](#_Toc137393251)

[7.1.3 Project Phase 1 Closure 115](#_Toc137393252)

[7.2 Future Work 116](#_Toc137393253)

[7.2.1 Future Functional Requirements 116](#_Toc137393254)

[7.2.2 Improvements Not Scoped 118](#_Toc137393255)

[7.3 Reflection 119](#_Toc137393256)

[Reference 121](#_Toc137393257)

[Appendix A 125](#_Toc137393258)

[Appendix B 126](#_Toc137393259)

List of Figures

[Figure 1 Miro board requirements gathering. 31](#_Toc137393260)

[Figure 2 - Azure architecture infrastructure diagram showing the azure components. 48](#_Toc137393261)

[Figure 3 Use case diagram. 52](#_Toc137393262)

[Figure 4 - Graph database design showing all node types and relationships. 53](#_Toc137393263)

[Figure 5 - Domain class diagram 56](#_Toc137393264)

[Figure 6 Service orientated class diagram 57](#_Toc137393265)

[Figure 7 - Main display design wireframe 58](#_Toc137393266)

[Figure 8 - Main display redesign wireframe 60](#_Toc137393267)

[Figure 9 - Create post design wireframe. 60](#_Toc137393268)

[Figure 10 - Admin posts design wireframe 61](#_Toc137393269)

[Figure 11 - Profile design wireframe 62](#_Toc137393270)

[Figure 12 - .NET project layout 68](#_Toc137393271)

[Figure 13 - Test driven development stages (Steinfield, G. (2020)) 70](#_Toc137393272)

[Figure 14 - Azure app registration 73](file:///H:\CwProjects\knowledge-sharing-platform\report\Work-based%20Project%20Report.docx#_Toc137393273)

[Figure 15 - Neo4j configuration 74](file:///H:\CwProjects\knowledge-sharing-platform\report\Work-based%20Project%20Report.docx#_Toc137393274)

[Figure 16 - Neo4j Bloom database 75](file:///H:\CwProjects\knowledge-sharing-platform\report\Work-based%20Project%20Report.docx#_Toc137393275)

[Figure 17 - Landing page 76](file:///H:\CwProjects\knowledge-sharing-platform\report\Work-based%20Project%20Report.docx#_Toc137393276)

[Figure 18 - Create post page. 78](file:///H:\CwProjects\knowledge-sharing-platform\report\Work-based%20Project%20Report.docx#_Toc137393277)

[Figure 19 - Landing page posts 81](file:///H:\CwProjects\knowledge-sharing-platform\report\Work-based%20Project%20Report.docx#_Toc137393278)

[Figure 20 - Profile page 85](file:///H:\CwProjects\knowledge-sharing-platform\report\Work-based%20Project%20Report.docx#_Toc137393279)

[Figure 21 - Graph Visualizer page 87](file:///H:\CwProjects\knowledge-sharing-platform\report\Work-based%20Project%20Report.docx#_Toc137393280)

[Figure 22 - Admin page 89](file:///H:\CwProjects\knowledge-sharing-platform\report\Work-based%20Project%20Report.docx#_Toc137393281)

[Figure 23 - NUnit teardown implementation 93](#_Toc137393282)

[Figure 24 - Passed Unit & Integration tests in Rider IDE test explorer 93](#_Toc137393283)

[Figure 25 - Knowledge sharing platform test coverage 94](#_Toc137393284)

[Figure 26 - Survey monkey result for question "How frequently have you been using the platform?" 97](#_Toc137393285)

[Figure 27 - Survey monkey result for question "How satisfied are you with the user interface and overall experience?”. 98](#_Toc137393286)

[Figure 28 - Survey monkey result for question " Is the platform useful for sharing your knowledge and expertise?”. 99](#_Toc137393287)

[Figure 29 - Survey monkey result for question "How often have you come across information shared by users from other departments?”. 100](#_Toc137393288)

[Figure 30 - Survey monkey result for question "How easy is it to use the platform and create posts?”. 101](#_Toc137393289)

[Figure 31 - Survey monkey result for question "Overall Satisfaction On a scale of 1 to 5, where 1 is very dissatisfied and 5 is very satisfied”. 102](#_Toc137393290)

[Figure 32 Azure cost analysis of platform 110](#_Toc137393291)

List of Tables

[Table 1 Gap Analysis 4](#_Toc137393292)

[Table 2 Use case #1 35](#_Toc137393293)

[Table 3 Use case #2 36](#_Toc137393294)

[Table 4 Use case #3 37](#_Toc137393295)

[Table 5 Risk Assessment 43](#_Toc137393296)

[Table 6 Cost Model (COCOMO) 46](#_Toc137393297)

[Table 7 Test plan template 63](#_Toc137393298)



Introduction

Introduction

Knowledge-sharing platforms are known as software systems that incorporate contemporary features to enable users to communicate information in real time and as a knowledge base. Using this technology, employees can post content and questions, enabling collaboration with subject-matter experts who can respond via comments.

Without the use of this system, items are often shared on an ad-hoc basis which can lead to duplication of effort and the unintended exclusion of departments from useful findings.

MHR departments known to frequently share knowledge with each other are listed:

* **Product Design** – Need to understand the needs and challenges of customers, what competitions offer, market trends, etc., through means including desk research, webinars, conferences, and customer events/feedback.
* **UX** – Need to understand how to present the requirements of product features in UI, using desk research, user testing, and customer events/feedback.
* **Research** – Look towards the horizon of new technology and its applications, via desk research, webinars, conferences, early access demos, in-house trials, and developing prototypes.
* **Marketing** – Need to understand MHR’s place in the market, the needs and challenges of potential customers and the competitor space, through means such as desk research, webinars, conferences, and customer events/feedback. As well as access to research institutions such as Gartner and IDC.

Other departments across MHR also conduct their own research, such as DPO which maintains updated knowledge of data privacy regulations and their impact on product development. However, for the scope presented, we will be focusing on the above four departments.

Items discovered/produced because of the research take on different forms, such as:

* Links (to articles, solutions, competitor websites, upcoming webinars, etc)
* Papers (academic papers, whitepapers produced by research organisations and/or competitors, etc)
* Books (several in the Research bookcase, and more dotted around the company)
* Videos (webinar recordings, YouTube videos, Teams meeting recordings, etc)
* Summary write ups/word documents/blogs/presentations produced (e.g., because of an investigation or of the findings from a conference)
* Excel spreadsheets of analysis (e.g., competitor analysis)

Means of ad-hoc sharing currently include**:**

* Email and Teams messaging
* SharePoint and OneDrive
* People First communities
* Miro
* Ad-hoc conversations

Below is a quote from Kate Featherstone, a business analyst within MHR.

“The size of MHR makes operations difficult - it's not so small that everyone knows each other but not large enough to work in individual departments/ entities successfully. This leads to projects occurring which could be beneficial to others work but they have no sight that it is happening.

Bumping into someone in a coffee queue quite often is where the connection of projects is made, but without those conversational occurrences they are likely to be missed!”

## Opportunity

MHR is a leading HR service provider who strives for continuous evolution of its product, which requires mass research be shared across the business. Currently departments are siloed between each other, causing harm to company culture, and decreasing efficiency. The opportunity to bring in a revolutionary way to share knowledge between departments would be a huge asset to the business. Initial improvements would be seen in the reduced duplication of effort, and as time advances it would encourage the reduction of silos by increased collaboration, sharing of knowledge, and between department idea generation.

## Gap Analysis

Table 1 presents a gap analysis that identifies and examines two primary areas the proposed initiative will address.

Table Gap Analysis

|  |  |  |
| --- | --- | --- |
| Current State | Desired State | Action to state |
| Content is currently shared between departments on an ad-hoc, informal basis, which can lead to duplication of effort or (unintended) exclusion of departments from useful findings. | An application for departments to manage and tag content. | Develop a cloud application where departments can share and tag content for other departments in a social feed. |
| Presently there is no access to globally search across the business for knowledge content, which causes increase of time due to not knowing content location. | Simplistic searching of content to allow users to use subjects. | Tagging of content and provide search functionally to filter down content. |
| Content has no managed structure and data is shared in different formats, making it difficult to analyse and report on. | Provide basic content structure and types to allow for manageable knowledge resources. | The system should allow users to enter content via forms which has strict content types, links, word documents, images, etc. |
| Content shared around the business has no feedback feed for employees to share thoughts on the knowledge resources. | Have collaborative feedback feature to allow for employees to feedback. | Have comments to be posted on the content for the author to review and reply. |

## Feasibility Analysis

A feasibility analysis evaluates a project's or idea's viability and practicality given the resources and limits that are set out. This section will conduct a feasibility analysis on the project if the knowledge sharing platform is going to be developed in-house.

### Operational

The system will be integrated into daily business operations when sharing information. Therefore, departments should ensure their staff use the system as part of their knowledge-sharing processes. The system will be trialled using the selected departments prior to being rolled-out company-wide. Prior the project conceptualisation, staff within selected departments were identified and acquired for trial use.

### Technical

This project will utilise the identical technology stack used by other business products, meaning project participants will not need additional training. This stack is Microsoft.NET for most backend services, so for this project a full stack .NET application will be developed. The code will also be deployed and maintained on the same cloud platform, Azure.

### Schedule

A single team could take on this project, or run it parallel with a current project, as there are several development teams operating within the company, provided they have redundancy and multiple developers per discipline.

### Economic

As this is intended to be an internal tool, rather than a marketable product or to incorporate into another product, the project’s return on investment will be modest. While it will not generate sales or outside income, it reduces duplication of effort across departments and increases idea generation.

### Feasibility outcome

Operationally, technically, and timewise, the project is feasible. While it may not have a large economic impact, it will improve departmental coordination and streamline operations.

## Risk Analysis

Table 2 shows the risk analysis for the system, identifying risk itself, the probability, impact, and how we can mitigate or reduce the risk.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **#ID** | **Risk** | **Likelihood** | **Impact** | **Mitigation** |
| 1 | Requirements are not fit | 3 | 5 – Requirements gathered do not match what the stakeholders want out of the system. | Complete business cases and provide questions for project stakeholders and document everything. |
| 2 | Scope Creep | 5 | 5 - When scope creep results in software development projects missing deadlines and taking longer than expected, risks can result. | Receive management approval before putting the project scope down in a Project Initiation Document. Throughout the project, refer to it and compare all modifications to it to ensure that they are in line with the Business Case. |
| 3 | Poor risk management | 3 | 4 – Poor risk management can lead to disruptions. These setbacks frequently result in one or more serious failures of a poorly thought-out software development project. | Create an in-depth risk assessment for the project which covers all risk and its mitigations. |

## Aim & Objectives

Aim: The purpose of this project is to introduce a cutting-edge knowledge-sharing system to MHR, a top provider of HR services, to improve teamwork, company culture, and efficiency.

Objectives:

* Reduce duplication of effort: Implement a knowledge-sharing system to cut down on duplicative research and streamline departmental processes.
* Encourage collaboration: By providing a forum for knowledge and idea sharing between departments, encourage increased collaboration.
* Enhance company culture: By removing silos and encouraging a culture of cooperation and knowledge sharing, the company culture will be improved.
* Increase idea generation: Increasing cross-departmental interactions and utilising collective knowledge and expertise can help generate innovative ideas.



# Background

Introduction

Knowledge-sharing platforms enable organisations to share and leverage knowledge company-wide. Resulting in better decision-making, innovation, and problem-solving. The following technical literature review will provide an overview of existing technical solutions, as well as identify potential gaps. Additionally, a review of similar tools/software/approaches will be presented to aid the proposed solution and its limitations.

## Technical literature review

Existing knowledge-sharing platforms include SharePoint, Miro Board, Slack, and Nuclino. These are popular amongst organisations for features such as document, workflow, and project management, as well as social collaboration, messaging, and analytics. However, these solutions do have some limitations.

For instance, both Nuclino and SharePoint can be challenging to implement due to their complexity and cost. A review by *PCMag (2020)* explored this, stating Nuclino requires users to have pre-existing technical expertise to aid the system setup and configuration. Additionally, its pricing structure is based on the number of users, causing higher expenses for larger companies *(Aviva (2019))*.

Similarly, *TechRepublic (2020)* notes SharePoint can be complex and time-consuming to set up and configure. Furthermore, its licensing model can be expensive for organisations with several users or those with advanced feature requirements. These factors can make systems inaccessible for smaller organisations with limited IT resources. However, for larger enterprises with complex needs, their advanced functionality still makes them a viable option, despite cost *(Microsoft, n.d.).*

On the contrary, Teams and Slack offer social communication and collaboration, over robust document management and workflow capabilities *(Microsoft, n.d.).* While they offer basic file-sharing and project management features, they may not provide advanced functionality such as version control or document approval workflows *(TechRepublic, 2020).* Therefore, they are not an ideal choice for organisations with complex knowledge management needs.

To gain a further understanding of the capabilities and limitations of each platform, a detailed review of three of these solutions is presented below.

## Team on MS Teams

MS Teams is a leading messenger app designed specifically for business. It excels in providing cooperative workspaces for real-time collaboration, meetings, communication, and file and app sharing. Having evolved into a vital tool used for daily file sharing among siloed departments, it facilitates the exchange of knowledge content *(What is Microsoft Teams? 2019)*. It's real-time messaging and file streaming capabilities enable seamless and efficient information sharing between users. Moreover, MS Teams integrates with Azure Active Directory, ensuring enhanced security through single sign-on (SSO) authentication. The business has organised Teams to assign users to their respective departments by granting them access to specific channels.

### Limitations

In isolation, MS Teams does not provide the services required to meet the business’ desired state, as outlined in the gap analysis (Table 1). MS Teams’ static library structure makes arranging and organising knowledge difficult and almost near impossible without SharePoint. This does not fit our need for an application that enables users to manage and tag content. Furthermore, it does not provide a simplistic solution for content sharing, as users must search each channel and chat manually, and therefore there is no solution to globally search files or information.

## Dedicated SharePoint site

Microsoft SharePoint is a document management and collaboration platform that allows corporations to manage archives, documents, reports, etc. that are essential to their operational procedures *(Microsoft. (2023). SharePoint Online Service Description).*

Staff who are used to managing documents and conducting business using SharePoint can easily switch to a dedicated knowledge management system within their current environment. It takes advantage of their familiarity with SharePoint, offers seamless integration, unifies document and knowledge management, offers improved features, and enables customisation and scalability. This change enhances knowledge sharing, streamlines workflows, and produces better business results.

### Limitations

SharePoint’s search capability, when used with wide query parameters, can be sufficient for obtaining a limited number of files. However, performance quickly falters when it is used with larger data sets and more specific search criteria *(MHR (2023) MHR Knowledge Base:2)*.

Outlined below are some limitations of the searching feature:

* Without further customisation, the only factor to filter search results is the document’s age.
* The users’ current site collection is the only one that can be searched.
* Administrators' configuration of the functionality significantly impacts the user search experience.

*(Brooks & Ryan Brooks, 2020)*

SharePoint's limitations in achieving the desired state of global content searching, as identified in the gap analysis, significantly impact its suitability as a knowledge platform. The platform's search functionality often restricts searches to specific sites or document libraries, making it challenging to retrieve information from the entire knowledge base. Users are required to navigate through multiple websites or libraries, resulting in reduced efficiency and effectiveness in knowledge retrieval and discovery.

## Miro board

Miro is a user-friendly, collaborative, digital whiteboard where users can easily organise objects and communicate through online chat or embedded video calls. Miro is utilised by MHR as a workspace for whiteboard sessions, enabling staff to share and collaborate on feature projects. If used, Miro would primarily serve as a visual knowledge platform, complemented by additional functionalities to facilitate seamless navigation. This can include vital content linking through the implementation of internal linking.

The knowledge base content can be generated using tools such as shapes, text, connection lines, and pen tools *(MHR. (2023) MHR Knowledge Base:2)*. Additionally, Miro offers access through SSO, enabling employees access without relying on external email accounts.

### Limitations

If unmanaged, Miro can become unwieldy making it hard to return to existing boards. Additionally, users have limited structures or templates to fill in content within Miro Board, which does not align with our desired state for manageable knowledge resources. Furthermore, this solution could make it difficult to analyse and report on the current knowledge content shared within the system *(MHR. (2023) MHR Knowledge Base: 2)*.

Additionally, Miro Board does not offer searching capabilities, other than the name of the board. This does not fit with our desired state of content searching as users must open multiple boards before finding the resources they require. Without the ability to search within the board's content, users are forced to manually navigate through multiple boards, opening each one to locate the desired information. This process can be time-consuming and inefficient, particularly when dealing with many boards or extensive amounts of content.

## Nuclino

Nuclino is a user-friendly platform that helps teams to share and edit documents in real-time through features such as visual editors and third-party connectors.

The option to create the material from scratch or use a template from their extensive library makes Nuclino easy to adapt to team requirements. This aligns with our desired state of having a manageable knowledge resource structure and allows the business to analyse knowledge resources companywide.

Furthermore, the effective search function rapidly displays pertinent results, with phrases highlighted, making it simple to locate desired information *(Hero, 2021)*. The system also supports collaboration and commenting, by displaying group changes in real-time. This helps to achieve the desired state of collaborative feedback by allowing employees to comment and share thoughts on the knowledge resource shared by the author *(Hero, 2021)*.

### Limitations

One significant limitation of Nuclino is that it operates outside the traditional business infrastructure, hosting knowledge and information externally. While this offers accessibility advantages, it raises concerns regarding sharing internal and confidential content. Storing critical information on an external platform can pose data privacy and security risks for businesses.

Additionally, Nuclino's pricing structure, at £4.00 per user per month, can be a significant constraint, particularly for company-wide implementation. Costs can quickly accumulate, especially for larger organizations. For instance, implementing Nuclino for 850 employees would amount to £40,800.00 annually in licenses alone.

Budgetary considerations heavily impact tool adoption feasibility. The high implementation costs of Nuclino may impede a company's ability to provide access to all employees, limiting collaboration and knowledge sharing. This pricing structure might hinder internal communication and restrict access for teams and departments.

In summary, Nuclino's limitations lie in its external infrastructure, potentially compromising data security, as well as its pricing structure, which can restrict company-wide accessibility and hinder internal communication.

## Gap in Existing Solutions

When it comes to providing effective search capabilities for knowledge repositories, existing solutions frequently fall short. While some platforms allow for keyword-based searches, the outcomes might not be relevant in every situation. This constraint opens a market opportunity for knowledge-sharing platforms that use graph databases and tagging techniques to increase productivity.

Graph databases allow for more complex relationships between data points and enable the platform to suggest relevant information based on these relationships *(Robinson, Webber & Eifrem, 2015)*. Additionally, tagging can improve search results by allowing for more granular categorisation of information. By incorporating these features, knowledge-sharing platforms can better support the organization's management of knowledge and enhance decision-making processes *(Brachman & Levesque, 2009)*.

In conclusion, the lack of effective search tools for knowledge repositories in the current solutions creates a market opportunity for knowledge-sharing platforms. These platforms can improve search effectiveness, support knowledge organisation, and decision-making by utilising graph databases and tagging techniques.

## Similar Tools/Software/Approaches

Other platforms have utilised graph databases and tagging, such as Neo4j and Ontotext, to improve search efficiency and recommendation accuracy. However, these tools are frequently targeted toward developers and require technical expertise to implement.

Therefore, there is a potential market for a knowledge-sharing platform to incorporate graph databases and tagging in a user-friendly manner, providing non-technical users with the ability to leverage these features for more efficient knowledge management. By providing easy-to-use tagging and search functionalities, knowledge-sharing platforms can improve the accuracy and relevance of search results, helping users quickly access the required information.

## Commercial Context

Knowledge-sharing platforms are widely used across various industries, including healthcare, finance, and technology. According to a report by MarketsandMarkets, the knowledge management software market is expected to grow from $6.1 billion in 2020 to $16.3 billion by 2025, with a compound annual growth rate (CAGR) of 21.5%.

This growth is expected to be driven by an increasing need for organisations to leverage knowledge for better decision-making and innovation. The proposed solution aims to address gaps in existing solutions and provide organisations with a competitive advantage through personalised and context-specific recommendations – incentivising knowledge sharing and collaboration.

## Research Studies

Three case studies conducted on knowledge-sharing platforms are presented below and examine each result and the impact they had.

### Deloitte's knowledge-sharing platform

Launched in 2012, D Street is a digital platform enabling Deloitte’s employees to collaborate and share knowledge across business units and geographic locations. It has since become a critical tool for Deloitte’s knowledge management strategy *(Schwartz et al., Knowledge Management, 2021)*.

Some key features include in D Street are:

1. Search Functionality: Users can search for relevant content using keywords, filters, and advanced search options. The platform also uses algorithms to recommend content based on users’ search history and previous activity.
2. Content Curation: The content team curate and organise content to ensure it is accurate, up-to-date, and easily accessible, as well as creating new content based on user feedback and analytics.
3. Collaboration Tools: D-street offers discussion forums, instant messaging, and video conferencing, which allow users to collaborate on projects and share knowledge in real time.
4. Analytics: Data analytics are used to track user activity and measure platform effectiveness. This helps Deloitte’s leadership team identify areas for improvement and make data-driven decisions regarding platform development.

Since its launch, D-Street has significantly impacted Deloitte’s knowledge-sharing management strategy. Some of the platform’s benefits include:

1. Increased Knowledge-Sharing: The implementation of D Street instigated a user increase of 400%, helping to break down silos and encourage collaboration across the company.
2. Improved Content Quality: The content team has made informed improvements to the quality and relevance of the platform’s content, thus cultivating better decision-making and problem-solving across the business.
3. Reduced Duplication: Employees can easily find and build on existing knowledge within the company, helping to reduce duplication of effort and resources.

Overall, D Street has become a crucial tool for Deloitte’s knowledge-sharing management strategy, helping the company to improve collaboration, productivity, and innovation across business units and locations.

### Knowledge Sharing in Digital Platform Ecosystems: A Textual Analysis of SAP's Developer Community

This paper by Kauschinger, M. and Klier, J.D explores the concept of knowledge-sharing within a digital platform ecosystem using SAP’s developer community as a case study. Using textual analysis, the study identifies factors that promote and inhibit knowledge sharing within the community. Furthermore, it examines how community members’ characteristics impact their engagement in knowledge-sharing.

Results showed community members’ motivation and willingness to share knowledge, as well as trust and support within the community, were important in promoting knowledge sharing. In contrast, factors such as complexity, platform ambiguity, time shortage, and absence of recognition for contributions, inhibited knowledge sharing. They also identified that different types of community members, such as core developers and peripheral members, exhibit different knowledge-sharing patterns.

Overall, the above paper provides insight into the complex dynamics of knowledge-sharing within digital platform ecosystems. Additionally, it highlights the importance of understanding which factors influence knowledge-sharing to enhance the effectiveness of such communities.

### Social Media for Knowledge-Sharing: A Systematic Literature Review

Ahmed and Ahmad present a comprehensive analysis of existing literature on the utilization of social media platforms for knowledge sharing. Through a systematic review of 39 studies published between 2008 and 2017, the authors identify various ways in which social media can enhance knowledge sharing within businesses.

Their review highlights that social media platforms provide a collaborative environment that fosters the sharing of ideas. By facilitating informal learning, these platforms enable individuals to exchange knowledge in a more relaxed and interactive manner. Additionally, social media platforms offer access to a diverse range of expertise, allowing employees to tap into a wider pool of knowledge and resources.

However, the success of social media for knowledge sharing is contingent upon several factors. The organisation's culture plays a crucial role in encouraging and supporting the use of social media platforms for knowledge sharing. Furthermore, technology adoption and user motivation are vital considerations that influence the effectiveness of these platforms.

To fully leverage the potential of social media for knowledge sharing, the authors recommend that organisations develop comprehensive strategies for implementing social media platforms. It is essential to integrate these platforms into existing knowledge-sharing practices to create a cohesive and collaborative environment. By embracing social media and incorporating it into their knowledge-sharing initiatives, organizations can unlock their full potential and enhance their ability to share and utilize knowledge effectively.

In summary, this paper offers valuable insights for organisations looking to leverage social media platforms for knowledge sharing, as well as highlighting the need for further research into this area.

### Comparison

Several differences can be seen between the three papers reviewed:

* Focus and Scope: Each paper focuses on different platforms and contexts for knowledge sharing. For instance, Deloitte’s platform, D Street, is designed for internal use, whereas the other two papers examine knowledge-sharing platforms in the context of digital platform ecosystems and social media.
* Methodology: Each paper uses different research methods. Deloitte’s case study is based on internal data and analysis, while the other two papers use textual analysis and literature review methodology to analyse content and trends in digital platforms and social media.
* Audience and Stakeholders: The papers are directed toward different audiences and stakeholders. Deloitte is primarily aimed at leadership and employees, whereas the other two are targeted at researchers, practitioners, and industry professionals interested in knowledge-sharing and digital platforms.

Overall, each study demonstrates the importance of digital platforms for knowledge-sharing in organisations. While they each focus on different platforms and contexts, they all highlight the importance of collaboration, innovation, and continuous learning to promote effective knowledge sharing. Furthermore, they emphasise the importance of features such as search functionality, content curation, and analytics in facilitating knowledge-sharing and improving organisational performance.

## Summary

Knowledge-sharing platforms can be essential tools for organisations looking to manage employee expertise and drive innovation. However, existing solutions have some limitations, such as a lack of efficient search capabilities and the need for technical expertise to implement more advanced features. However, graph databases and tagging have been utilised in some tools to improve search efficiency and provide accurate recommendations, although they are often still targeted toward technical users.

Therefore, there is market availability for a knowledge-sharing platform that incorporates graph databases and tagging in a user-friendly manner, allowing non-technical users to leverage these features for more efficient knowledge management. By incorporating these features, knowledge-sharing platforms can improve the accuracy and relevance of search results, enabling users to quickly access their required information.

As the knowledge management software market grows, the solution outlined in this paper can offer organisations a competitive edge. For instance, it can enable efficient knowledge sharing and collaboration, while providing personalised and context-specific recommendations. Furthermore, by leveraging machine learning techniques and gamification, such as natural language processing (NLP), social network analysis, and machine learning algorithms, organisations can enhance their decision-making processes, driving innovation.

## Analysis of Legal, Social, Ethical and Professional Issues

### Legal Issues

**Licensing**

This project is currently intended for internal use and does not require licensing. However, should the business wish to transition to customer consumption, the product will need to be licensed. A software license would outline the rights of the creator and user, including the software’s usage restriction and payment details *(Lutkevich and Lebeaux, 2021)*. Although, if the project becomes open source and requires no official software licensing, it can provide an MIT licence.

**Intellectual property of content**

The sharing of intellectual property can hold risks. For instance, the proposed knowledge platform could be taken down if its content violates a patent, trademark, copyright, trade secret, or trade dress. Such infringement could incur damages of several million pounds or an injunction.

Therefore, precautions should be taken to prevent this. One such prevention is that all employees should reference any knowledge shared in the repository if it is from an external source.

**Data ownership**

As the product allows users to enter information and content, it falls under the GDPR Act. Therefore, data collection and storage must adhere to the appropriate regulations and only data essential to the operation of the platform should be stored and/or handled.

Any violations of this must be disclosed to the individual's impact and failure to comply could result in significant fines, as well as a negative company reputation (Art. 5 GDPR – principles relating to the processing of personal data 2021)

**Source control**

Any code produced during this project will be stored within source control, Azure DevOps with only verified business personnel given access permission. The use of a verified source control provider is imperative to ensure the code is secure. The implementation will not be open source, meaning external individuals will not be given read or write access to the repository, protecting the intellectual property of the product.

### Social Issues

**Free text content**

One product feature enables users to share free text content. This feature could be misused and lead to insensitive texts and content being shared, which could offend other users leading to social issues within the business.

Social issues are guarded by company policies regarding communication and interaction of software between employees. Therefore, content admins will be allowed to manage all content within the system, meaning all free text content will be reviewed and inappropriate content removed.

**Content sharing**

Users will also be able to share external content within the system, which could lead to the sharing of inappropriate work content, potentially causing offence and social issues. To prevent this, admins will have a list of verified domains and users may only share information from these domains.

### Ethical Issues

**Commenting for relative feed (gamification)**

Users could attempt to force other users to comment on their knowledge content to promote their posts on other’s feeds. This is unethical as it could lead to the exploitation of the comment system for gamification. For this reason, commenting will have no effect on the order of posts on the user’s feed.

**Plagiarism**

The sharing content feature could also be abused whereby users aim to plagiarise knowledge content as their own. The copying of another's work is unethical as it does not offer credit to the original author, leading to false impressions of a user’s skills and capabilities.

Therefore, Content Admins will be responsible for verifying content knowledge across the system. A further feature, if feasible, would be to introduce a ‘Turnitin’ like software which can check the content automatically as part of the post feature.

### Professional Issues

**Business related content**

The system’s content should be limited to business or business skill-related information to aid the upskilling or training of employees. Other content is not desired at this point as the tool is meant for internal use.

**BCS (Code of conduct)**

Software Developers are required to abide by a code of ethics and guidelines when creating, designing, and maintaining software.

Below are some examples from BCS Code of conduct for making software for everyone *(BCS, Code of Conduct, 2011)* –

* have respect for the environment, other people's well-being, their privacy, security, and public health.
* respect the legal rights of others as they deserve.
* conduct the professional duties without prejudice based on a person's gender, sexual orientation, marital status, race, colour, nationality, ethnicity, religion, age, or disability, or any other condition or requirement.
* encourage equal access to the advantages of IT and work to advance the participation of all societal segments whenever chances present themselves.

Several steps will be taken to integrate the BCS Code of Conduct into a knowledge-sharing platform. First, the platform will undergo careful curation to make sure that the available content complies with the Code's ethical standards. This entails choosing books, articles, and training materials that place a strong emphasis on protecting people's privacy, security, and overall well-being. The platform will also offer instructions on how to respect others' legal rights and carry out professional obligations without bias or discrimination. Additionally, it will encourage participation from all societal groups while promoting equal access to IT benefits.

We can promote a culture of responsible software development and ethical business practises within our company by incorporating these principles into the platform's content and making it easily accessible to employees.



Project Definition

Introduction

This chapter provides an overview of the project’s aims, objectives, planning, and success criteria for the proposed knowledge-sharing platform.

## Aims & Objectives

This project aims to address the issue of siloed departments due to a lack of knowledge sharing. It proposes to do so through the development of a knowledge-sharing platform, which has been shown to promote internal collaboration through information sharing and transparency. Additionally, it will provide a centralised location for employees to share their expertise, as well as access information and resources.

### Objectives

* Create a user-friendly, intuitive platform that encourages active participation and engagement from employees.
* Promote a culture of transparency and knowledge sharing by providing easy access to relevant information and resources.
* To break down silos between departments and encourage collaboration through a centralised location for sharing knowledge and expertise.
* Improve efficiency and effectiveness of internal communication by providing a platform for real-time feedback and discussion.
* Increase productivity and reduce duplication of effort by providing employees with access to a wide range of resources and expertise.
* Ensure the platform is secure and compliant with relevant data privacy regulations, such as GDPR or CCPA.
* Integrate the platform with existing systems and workflows to maximise adoption and usability.
* Provide analytics and reporting features so administrators can track user engagement and content performance while using this information to improve the platform as needed.

## Requirement analysis

### Research and Discovery

Requirements will be acquired via a collaborative process known as Joint Application Design (JAD). This session will aim to align the business needs, as identified in the GAP analysis, with the proposed solution (Contributor, T.T., 2007). As this is a novel project, there is no existing system in place. Therefore, the JAD session will focus on identifying and developing new requirements, without drawing comparisons to current systems. Using effective communication and active participation, a deeper understanding of the project’s objectives will be achieved. This cooperative approach will help to clarify uncertainties and ensure the final delivery meets desired expectations.

In addition to these sessions, a survey will be curated and distributed to select individuals within each of the organisation’s departments. This survey will aim to gather information from key stakeholders regarding how knowledge is currently shared within their teams. The result's variations and similarities will be analysed, and the insights will be used in further brainstorming and workshop sessions. From this a comprehensive understanding of potential system users will be gained, helping to develop practical solutions to facilitate cross-department knowledge sharing.

To gain further insight into the current situation, from alternate perspectives, several questions will be asked, including:

* Where are your notes, papers, and links currently stored?
* When you attend events, do you share your notes? If yes, with whom?
* Do you share knowledge (links/papers/documents) with other departments?
* What would you find beneficial if you could search another department’s notes/documents?
* When someone shares an article or video with you, do you find the time to read/watch them?

By using the above techniques, we hope to reduce the risk of inadequate requirements such as “#1 – Requirements are not fit’ during the risk analysis phase.

The JAD session will span three days, and each survey will be carefully analysed to generate appropriate questions. The breakdown of the JAD session is as follows:

* Day 1: Review of survey feedback and examination of use cases.
* Day 2: Analysis of requirements, brainstorming, and workshop activities.
* Day 3: Specification design session.

During each session, a senior stakeholder, such as the CEO or Project Manager, will be present to ensure effective decision-making and representation *(Davidson, E.J., 1999).* This stakeholder will have final authority when making related decisions as well as aiming to alleviate or address initial concerns. We will aim to foster an environment in which attendees can be actively engaged and feel their input is valued, encouraging ownership and alignment with the project objectives.

## Software Development Methodologies

In this section there are two different approaches presented that are potential methods for competition of this project.

### Waterfall

Outlined by Dennis et al. (2015), the Waterfall model is a traditional approach when discussing the development life cycle of systems design in a linear and sequential manner. As the name suggests, the model progresses methodically from one phase to the next, with no overlap between them. This system means the output of one phase serves as the input for the following highlighting the strict sequential nature of the Waterfall model *(Dennis et al., 2015).*

However, this linear order does have a drawback in that it is not adaptable to changes. Alterations in stakeholder objectives or demands could upheave the arrangement, necessitating a revision or potentially a new blueprint. Due to this, Waterfall project management does not work well for knowledge-based tasks, such as computer programming. However, it’s workflows can be duplicated for upcoming, comparable activities, making up for its lack of flexibility *(What is Waterfall Project Management? 2016).*

### Agile/Scrum

The agile methodology employs a “Waterfall” style of software development while focusing on communication and results. This concept focuses on self-management, short timescales, and flexible planning, which allows for system improvements over time.

Each phase is kept under four weeks, enabling the system to advance quickly. This eliminates the possibility of lengthy timelines and lessens the likelihood of failed projects due to earlier errors. While there is less documentation, there is a stronger emphasis on the importance of the key stakeholder’s needs by allowing employees to closely collaborate with them.

Scrum is a component of the Agile approach most used by firms due to its simple framework for processes. In Scrum, a Product Owner creates a prioritised “wish list” known as a product “backlog”. From this, small teams take pieces of the backlog to produce a sprint, which is a short period (between 2-4 weeks) during which the backlog item will be evaluated, created, and integrated into the larger system. A “daily scrum” will take place each day, in which the teams will track the sprint’s progress and the Scrum Master will direct the time to keep on task. Once the sprint is finalised, the team then review and deploys the changes, leading to a new sprint being started.

An Agile strategy is often used to increase a project's adaptability and to help deliver results sooner so that stakeholders can inspect it and provide their input. This ultimately provides the end consumer with a richer result *(Denning, 2022).*

### Chosen Development Method

The proposed solution for the project will employ the Agile Scrum methodology to effectively plan, research, and develop the system. The short, fixed iterations of Agile are advantageous as they focus on a small set of components at a time. With each iteration, a prototype is created, tested, and distributed, facilitating faster user feedback, and informing current and future revisions. This helps us to gain a comprehensive understanding and develop the appropriate solution for internal use.

Furthermore, the reliance on prototypes in the internal system makes Agile beneficial as it allows requirements to be divided into shorter timeboxes, each containing a small number of features. This flexibility to modify features, as highlighted by *Sillitti (2011)* is advantageous in this context.

Moreover, adopting Scrum empowers the team by giving them the authority to complete tasks, making them a self-sufficient division of the company. This autonomy gives them greater control of their work, eliminating the need to rush tasks to deadlines.

While an Agile approach could incur higher development costs and increase project timeframes, the emphasis it places on team communication is vital. The gradual progression also fosters empowerment and promotes collaboration, leading to a more successful outcome.

The Agile Scrum method was chosen over the Waterfall process as it does not require key stakeholders to define their requirements upfront. It recognises that customers may not have a clear vision of the result at the project’s inception and prevents frequent requests for specification changes. Therefore, it reduces the risk of requirements not aligning with expectations.

Therefore, Agile offers a valuable, lightweight framework, suitable for small-scale projects with compressed timelines, such as the proposed solution.

## Functional and Non-Functional requirements

An analysis of the stakeholder requirements is given below. These requirements are divided into functional and non-functional categories. Those determined functional consist of features the system/date must contain and those that are non-functional are general behaviours that the system needs.

A picture containing text, screenshot, font, number

Description automatically generated

Figure Miro board requirements gathering.

Figure 1 showcases the requirement gathering sessions that were conducted on Miro-board with the stakeholders. The following functional and non-functional requirements were at outcome of this session using the JAD process.

### Functional Requirements

**User Authentication and Authorization**

* The platform shall integrate with Active Directory for Single Sign-On (SSO) to simplify user authentication and authorization.
* The platform shall grant different levels of authorization to different types of users, such as admins and regular users.

**Content Creation and Management**

* The platform shall allow users to create and publish new content, including articles, free form, and books.
* The platform shall allow admins to manage content, including reviewing and approving new posts before they are published, and removing or archiving posts that are outdated or no longer relevant.

**Tagging and Search Functionality**

* The platform shall include a robust tagging system that allows users to tag their posts with relevant keywords and categories.
* The platform shall include a powerful search function that allows users to easily find content based on specific keywords or categories.

**User Engagement and Collaboration**

* The platform shall encourage user engagement and collaboration through features such as commenting and rating.
* The platform shall allow users to connect with other users who share similar interests or expertise.

**Analytics and Reporting**

* The platform shall provide analytics and reporting features that allow users to track user engagement and content performance, such as views, likes, and comments.
* The platform shall allow admins to export data for further analysis.

**Security and Data Privacy**

* The platform shall be secure and comply with relevant data privacy regulations, such as GDPR or CCPA.
* The platform shall include features such as encryption, user access controls, and data retention policies.

**Integration with Other Systems**

* The platform shall be able to integrate with other systems used by the company, such as an internal communication or project management system.
* The platform shall allow for easy sharing and collaboration between different teams and departments.

### Non-functional Requirements

**Performance**

* The platform shall be able to handle a large volume of users and content without significant degradation in performance.
* The platform shall be responsive and load quickly.

**Usability**

* The platform shall be easy to use and navigate.
* The platform shall include user-friendly interfaces for content creation and management.

**Reliability**

* The platform shall be always available and accessible to users.
* The platform shall include backup and disaster recovery mechanisms to ensure data integrity.

**Compatibility**

* The platform shall be compatible with a variety of devices and browsers.
* The platform shall comply with relevant web standards and protocols.

### Conclusion

The knowledge sharing platform will be an important tool for internal communication and collaboration within the company. By following the requirements outlined in this software requirement specification, the platform will provide a secure and efficient way for employees to share knowledge and expertise, while also facilitating user engagement and collaboration.

## Success Criteria

Below are the success criteria for the knowledge sharing platform project –

1. Increased User Engagement: The platform should see an increase in user engagement, as measured by the number of active users, the frequency of user interactions, and the level of user satisfaction with the platform.
2. Improved Knowledge Sharing: The platform should facilitate improved knowledge sharing within the organisation, as measured by the number and quality of posts created the level of interaction and collaboration among users, and the impact of the platform on business outcomes.
3. Reduced Silos: The platform should help to break down silos between departments, as measured by the level of cross-functional collaboration, the sharing of information and best practices across departments, and the impact of the platform on cross-departmental business outcomes.
4. Efficient Administration: The platform should be easy to administer, as measured by the time and resources required to manage user accounts, moderate content, and analyse platform usage data.
5. Effective Recommendations: The platform should provide effective post recommendations to users, as measured by the relevance and usefulness of recommended posts, the level of user engagement with recommended posts, and the impact of recommended posts on business outcomes.
6. Secure and Reliable: The platform should be secure and reliable, as measured by the absence of security breaches or data leaks, the uptime and performance of the platform, and the level of user trust in the platform.
7. Scalable: The platform should be scalable to accommodate future growth and user demands, as measured by the ability of the platform to handle increasing numbers of users, posts, and interactions without sacrificing performance or user experience.

These success criteria can be used to evaluate the effectiveness of the knowledge sharing platform and determine whether it has met its goals and objectives.

## Use Case Analysis

Use Case Analysis helps to understand how a software system will be utilised by users and stakeholders, by identifying user requirements, defining system behaviour, and specifying system functionality. This information is critical for developing an effective test strategy that ensures the system meets these needs.

The system's primary functionality is represented by the three use cases listed below.

### Use Case #1

Table Use case #1

|  |  |
| --- | --- |
| **Use Case Name** | A user adds knowledge content |
| **ID** | 1 |
| **Priority** | High |
| **Actor** | Subject matter expert (user) |
| **Description** | The user selects content template to fill (links, papers, books, free texts). Once form is completed, the user will click save and publish, this will then persist this knowledge resource in a database. |
| **Trigger** | The user wants to use this system to share a piece of knowledge information with the rest of the business. |
| **Preconditions** | The application has verified and approved the user credentials. |
| **Normal Course** | * User clicks on add content button. * User selects content template from dropdown. * User fills out content form. * User submits and publishes the content |
| **Postconditions** | * Content is persisted within the database and shown on other user’s content feeds. * User logs out of application. |
| **Exceptions** | * User credentials are not valid or timed out. * Profanity filter picks up words. |

### Use Case #2

Table Use case #2

|  |  |
| --- | --- |
| **Use Case Name** | A searches knowledge content |
| **ID** | 2 |
| **Priority** | High |
| **Actor** | Content consumer (user) |
| **Description** | The user should be able to search for the content with the system via headers and tags. |
| **Trigger** | The user wants to use this system to learn different knowledge around the business. |
| **Preconditions** | The application has verified and approved the user credentials.  Content has already been added in the system with tags. |
| **Normal Course** | * Users click on navigation icon on menu bar. * Text field appears for user to write into * User press enter with text field filled. * List of content results are returned |
| **Postconditions** | * User can click content. * The user logs out of the system. |
| **Exceptions** | * The user's login credentials are not valid. * No content in the system was added. |

### Use Case #3

Table Use case #3

|  |  |
| --- | --- |
| **Use Case Name** | Add comments to content |
| **ID** | 3 |
| **Priority** | Medium |
| **Actor** | Users |
| **Description** | The user should be able to comment and reply on content. |
| **Trigger** | The user wants to use this system to provide or be given feedback on knowledge shared around the business. |
| **Preconditions** | The application has verified and approved the user credentials.  Content has already been added in the system with tags. |
| **Normal Course** | * User navigates to content. * User enters comment text field and press enters. * User enters reply comment text fields and press enters (for reply to threads) |
| **Postconditions** | * The user logs out of the system. |
| **Exceptions** | * The user's login credentials are not valid. * No content in the system was added. |

## MoSCoW

Below are the identified functional and non-functionals requirements of MoSCoW.

The term MoSCoW stands for must-have, should-have, could-have, and won't-have (or will not have right now) for the project.

### Must-have:

* User authentication and authorization with Active Directory integration.
* User profile creation and management.
* Content creation, including articles, books, and open texts, with the ability to tag and search for content.
* Admin dashboard for managing users and content.
* Recommended post feed for users based on their interests and engagement.
* User feedback and discussion features for each post.

### Should-have:

* Social features, such as liking and commenting on posts.
* Following of tags
* Analytics and reporting features users to track user engagement and content performance.
* Mobile-responsive design for easy access from different devices.
* Analytics and reporting features users to track user engagement and content performance.

### Could-have:

* Notification system for updates and activity on posts and subscriptions.
* The ability to follow users.
* Comment moderation tools for admins.
* Machine learning algorithms to improve the accuracy of recommended posts.
* Integration with existing internal systems and workflows
* Integration with external knowledge sources, such as industry publications or news sites.
* Integration with external social media platforms for sharing content.

### Won't-have:

* Support for outdated web browsers or devices.
* Integration with external payment systems for purchasing content.
* Native mobile applications for iOS or Android.
* Integration with third-party tools, such as project management or collaboration platforms.
* Customisable email templates for notifications and alerts.

## Project management

This section provides an overview of the project planning phase, including the required timeline and resources, as well as the chosen project management approach.

## Scope

The objective of this project is to create a cloud-based central knowledge platform that streamlines processes, encourages collaboration, facilitates the sharing of ideas and knowledge, and promotes idea generation across different departments. The central repository will enable MHR employees to effectively manage, provide feedback, and share content with one another.

The project plan for the knowledge sharing platform will consist of three sprints, which will be named Phase 1. This report will provide a detailed description of these sprints. In agreement with the MoSCoW prioritisation technique, that the main goal is to concentrate on achieving the "Musts" and "Shoulds" within the functional requirements. Any "Shoulds" and "Coulds" that were not covered in phase 1 will be covered in the ensuing phases, that will be planned into sprints.

The following requirements for phase 1 of the project:

* User authentication and authorization with Active Directory integration.
* User profile creation and management.
* Content creation, including articles, books, and open texts, with the ability to tag and search for content.
* Admin dashboard for managing users and content.
* Recommended post feed for users based on their interests and engagement.
* User feedback and discussion features for each post.
* Social features, such as liking and commenting on posts.
* Following of tags
* Analytics and reporting features users to track user engagement and content performance.
* Mobile-responsive design for easy access from different devices.
* Analytics and reporting features users to track user engagement and content performance.

## Schedule plan

The project was scheduled to commence in January 2023 and is expected to conclude in mid-June 2023, generating a total duration of six months. This timeline includes buffer periods to accommodate unforeseen work requirements or potential delays when completing existing tasks. Following an Agile approach, each timebox will be allocated work items prior to its commencement, typically spanning a two-week period. This will enable the team to gauge the capacity of each member prior to the upcoming sprint, facilitating the planning of sufficient work whilst allowing contingency to address issues or bugs that could impede the progress of tasks.

## Resource planning

Below describes the development project team, explaining why each role is needed for the project –

1. Lead Full-stack Engineer: Oversees technical aspects, manages development process, guides system architecture, scalability, and performance.
2. Software Engineer (.Net): Develops robust server-side components, implements business logic, integrates external services, ensures .Net framework utilization and security measures.
3. Frontend Web Developer: Designs and develops engaging and intuitive UI/UX, implements responsive design, interactive features, and accessibility standards.
4. UX Designer: Shapes user experience and interface, conducts research, creates wireframes, and prototypes, optimizes knowledge sharing and retrieval.
5. Business Analyst: Bridges technical team and stakeholders, analyses requirements, defines functional specifications, manages project scope, facilitates communication.

Each role will help the team leverage their specialised skillsets and knowledge to build a comprehensive and user-friendly knowledge-sharing platform to meet the organisation's needs while providing a seamless and intuitive experience for internal users.

## Gantt Chart & PERT Diagram

Appendix 1 contains the project's resource plan in the form of a Gantt chart, available in the file 'Project\_Management\_Plan\_GANNT.mpp'. Additionally, the project's PERT diagram can be found in the same appendix, accessible through the file 'Project\_Management\_Plan\_PERT.mpp'. The PERT diagram illustrates the project's dependencies.

The Gantt chart project plan provides a comprehensive overview of critical tasks within the project. It begins with the initial analysis and requirement gathering phase, followed by establishing the scope specifications. The development timeline for phase 1, consisting of three timeboxes, is also outlined. Task dependencies have been identified, forming the critical path for the project.

Certain tasks can be executed in parallel with others. For example, within timebox 1, tasks 4.1.2.1, 4.1.2.2, and 4.1.2.3 can be simultaneously carried out. These tasks are only dependent on the refinement of work (4.1.1.1) and do not rely on each other. They are performed independently by different team members. Another instance of parallel tasks is seen in 2.6.1 and 2.6.2, both involving modelling tasks. These tasks can be executed concurrently as they do not have any dependencies on each other, but rather depend on the project requirements being defined beforehand.

## Risk Assessment

Table Risk Assessment

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Description | Likelihood | Impact | Severity | Owner | Mitigation |
| Scope of project might change | Medium | Medium | Medium | Business Analyst | Ensure all scope is defined in detail via the design workshops and all stakeholders are involved. |
| Lack of communication leads to unclear thinking and misunderstanding. | Low | Medium | Medium | Team | Create a communication plan that details the timing, purpose, and target audience for each communication. Timebox reviews will allow the team time to feedback on the timebox. |
| Deficient testing leads to large post go live bug list . | High | High | High | Lead Engineer | Make sure the tester creates test cases and quality checks, and that they guard the window for testing and quality assurance. |
| Small team absences/sickness might delay project. | Medium | High | High | Lead Engineer | Additional time buffer added to guard any unplanned absences or sickness. |
| Undefined project needs and purposes | Medium | High | High | Business Analyst | Finish the business case and make sure the project charter's purpose is clearly stated. |
| Unexpected tasks that need to be accommodated | Low | Medium | Medium | Lead Engineer | Verify each plan and each quantity survey. Record all planning assumptions before the project begins. |

## Azure DevOps

DevOps is an approach to software development that emphasises collaboration, communication, and automation between development and IT operations teams *(Humble & Farley, 2010)*. Azure DevOps is a cloud-based platform that provides a suite of tools for managing the software development lifecycle, including agile project management, version control, continuous integration and delivery, and automated testing and deployment. By using Azure DevOps, teams can improve their efficiency, increase collaboration, and ensure high-quality software delivery *(Kapoor, 2021).*

Using Azure DevOps, teams can easily track their tasks and become more agile by breaking down projects into smaller sprints. This iterative process allows for faster delivery of high-quality software and ensures that teams meet project deadlines. At the end of each sprint, teams can release the latest features for feedback and make necessary adjustments before moving on to the next sprint.

## Cost Model (COCOMO)

Constructive Cost Model, also known as COCOMO, is a popular framework for calculating software costs. This model evaluates the required time, work, and costs associated with creating a software application. It considers several variables, such as the project's size, expressed in functional points or lines of code, the team's familiarity with the selected technologies/platforms, and the desired delivery time.

Instead of relying on the size of the codebase or the complexity of the tools and platforms, functional points are a way to estimate the cost of software by measuring the functionality that is included. In comparison to relying solely on lines of code, this approach makes it easier for non-technical stakeholders to understand and estimate software requirements.

Functional points based on the Fibonacci sequence for the given functional requirements in phase 1, “Musts” & “Shoulds”:

1. User authentication and authorization with Active Directory integration: 3 points
2. User profile creation and management: 2 points
3. Content creation, including articles, books, and open texts, with the ability to tag and search for content: 5 points.
4. Admin dashboard for managing users and content: 3 points.
5. Recommended post feed for users based on their interests and engagement: 2 points.
6. User feedback and discussion features for each post: 5 points
7. Analytics and reporting features to track user engagement and content performance: 5 points.
8. Social features, such as liking and commenting on posts: 2 points.
9. Following of tags: 1 point
10. Mobile-responsive design for easy access from different devices: 2 points
11. Integration with existing internal systems and workflows: 2 points

Below is the software development estimated cost of the project, this will give the cost of the project with the overhead cost of a developer within the business.

Software Development (Elaboration and Construction)  
Effort = 2.2 Person-months  
Schedule = 4.7 Months  
Cost = £5338.81  
Total Equivalent Size = 1600 SLOC  
Effort Adjustment Factor (EAF) = 0.45

Table Cost Model (COCOMO)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Phase | Effort (Person-months) | Schedule (Months) | Average Staff | Cost |
| Inception | 0.1 | 0.6 | 0.2 | £320.79 |
| Elaboration | 0.5 | 1.8 | 0.3 | £1283.17 |
| Construction | 1.7 | 3.0 | 0.6 | £4063.39 |
| Transition | 0.3 | 0.6 | 0.5 | £641.59 |

**Software Effort Distribution for RUP/MBASE (Person-Months)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Phase/Activity | Inception | Elaboration | Construction | Transition |
| Management | 0.0 | 0.1 | 0.2 | 0.0 |
| Environment/CM | 0.0 | 0.0 | 0.1 | 0.0 |
| Requirements | 0.1 | 0.1 | 0.1 | 0.0 |
| Design | 0.0 | 0.2 | 0.3 | 0.0 |
| Implementation | 0.0 | 0.1 | 0.6 | 0.1 |
| Assessment | 0.0 | 0.1 | 0.4 | 0.1 |
| Deployment | 0.0 | 0.0 | 0.1 | 0.1 |



Design

Introduction

This chapter aims to provide a comprehensive overview of the knowledge sharing platform's design, offering valuable insights into the justification behind its specific design choices. It encompasses architectural designs, UML designs, and UX designs, all of which have been produced to achieve optimal functionality and user experience.

## High-level architecture design

It's good to have a high-level architecture design for several reasons:

* Clarity: A high-level architecture design provides a clear picture of the system and its components. It helps in understanding the overall structure, the roles, and responsibilities of each component, and how they work together.
* Scalability: A high-level architecture design provides a framework for scaling the system. It helps in identifying the areas that need improvement and the ones that can be scaled horizontally or vertically.
* Reusability: A high-level architecture design promotes reusability of components. It helps in identifying the components that can be reused in different systems and the ones that need to be built from scratch.
* Maintainability: A high-level architecture design makes the system more maintainable. It provides a roadmap for maintenance activities and helps in identifying the areas that need attention.

Figure 2 is the architecture diagram for a containerized Web application with SSO and a Neo4j database as its persistence layer.

A picture containing text, screenshot, diagram, font

Description automatically generated

Figure - Azure architecture infrastructure diagram showing the azure components.

The architecture for this application would consist of the following components:

* Presentation Layer: The presentation layer would consist of the Web application that would be containerized using Docker. This layer would be responsible for rendering the user interface and communicating with the other layers.
* Business Logic Layer: The business logic layer would consist of the services that would implement the business logic of the application. These services would communicate with the persistence layer to retrieve and persist data. They would also communicate with the presentation layer to receive and send data.
* Persistence Layer: The persistence layer would consist of the Neo4j database that would be containerized using Docker. This layer would be responsible for storing and retrieving data. It would communicate with the business logic layer to provide data access.
* Single Sign-On Layer: The SSO layer would be responsible for authentication and authorization of users. It would authenticate users and generate access tokens that would be used by the application to access the resources.
* Container Orchestration Layer: The container orchestration layer would be responsible for managing the containers running the application and the database. It would ensure that the containers are running, healthy, and accessible.

### Web application framework

For the web application, this project will be using .NET Blazor, this is a web framework developed by Microsoft that allows developers to create web applications using C# and .NET instead of JavaScript.

It allows you to build rich, interactive user interfaces using a combination of HTML, CSS, and C#. Blazor provides two hosting models - server-side and client-side. In server-side Blazor, the application runs on the server, and the UI is rendered in the browser using SignalR. In client-side Blazor, the entire application is downloaded to the client browser and runs in WebAssembly. *(Microsoft. 2022, Blazor Overview).*

Blazor's lightweight design is one of its main advantages. Applications built using Blazor can be compact and quick because they don't need a lot of JavaScript or other complicated dependencies. Instead, the application's logic can be written in C# and either run on the server or compiled to WebAssembly. Blazor's lightweight design makes it particularly advantageous for projects that move swiftly since it enables programmers to produce applications quickly and with few dependencies.

### Open ID Single Sign On (SSO)

A standards-based, open-source authentication system called Azure OpenID enables you to authenticate users across numerous apps and services. It is a component of Microsoft's Azure Active Directory (Azure AD), a solution for cloud-based identity and access management. You can offer single sign-on (SSO) to the applications using Azure OpenID, enabling users to access numerous applications using a single set of credentials. Blazor supports several different authentication services, including Azure AD. This will make it easy for us to integrate the Blazor app with the company's AD.

### Azure container instances

Azure Container Instances (ACI) is a serverless container hosting solution that allows you to run the Docker containers quickly and easily without the need to manage servers or infrastructure. ACI is a great choice for hosting containerized Blazor applications with SSO and a Neo4j database as its persistence layer because it provides the following benefits:

* Easy Deployment: ACI makes it easy to deploy containerized applications without having to manage infrastructure or servers. With ACI, you can deploy the containers with a simple command or through the Azure portal.
* Scalability: ACI allows you to scale the containerized application easily, without having to manage the underlying infrastructure. You can scale up or down the container instances based on the workload requirements, and ACI will automatically manage the resources needed to run the application.
* Cost-effectiveness: With ACI, you only pay for the resources you consume while running the container instances. This means you can run the application cost-effectively without having to pay for unused resources.
* Security: ACI provides built-in security features such as network isolation, encryption at rest, and role-based access control (RBAC). This helps ensure that the application and data are secure.

By using Azure Container Instances to host the containerized Blazor application with SSO and a Neo4j database as its persistence layer, you can take advantage of the benefits of serverless container hosting and quickly deploy and scale the application with minimal effort.

## Use Case Diagram

The use case diagram for the knowledge sharing platform is shown below in figure 3. The design is based on the functional requirements and the use case analysis captured in section 3.7. The Author, the Reader, and an Admin are the three actors in it. The platform for knowledge sharing is its system boundaries.

A picture containing text, screenshot, diagram, circle

Description automatically generated

Figure Use case diagram.

## Graph database design

Figure 4 represents the graph database design diagram.

Diagram

Description automatically generated

Figure - Graph database design showing all node types and relationships.

The graph database design consists of four types of nodes: **Department**, **Person**, **Post**, and **Tag**.

**Department** nodes have a one-to-many relationship with **Person** nodes, representing the fact that a department can have many employees. They also have a one-to-many relationship with **Tag** nodes, representing the fact that a department can have multiple interests or preferences.

**Person** nodes have a one-to-many relationship with **Post** nodes, representing the fact that a person can write multiple posts. They also have a one-to-many relationship with **Tag** nodes, representing the interests or preferences of the person.

**Post** nodes have a one-to-many relationship with **Tag** nodes, representing the fact that a post can have multiple tags.

**Tag** nodes represent the interests or preferences of a department, person, or post. Each **Tag** node is connected to one or more **Department**, **Person**, or **Post** nodes via a "LIKES" or "HAS\_TAGGED" relationship, respectively.

This graph database design is good for quick searching of tags for posts and recommendations of posts based on users' liked tags because it models the relationships between posts, tags, and users in a highly connected graph structure. For example:

* To search for posts that are tagged with a certain tag, you can simply traverse the graph from the Tag node to the **Post** nodes, returning all posts that are associated with the given tag.
* To recommend posts to a user based on their liked tags, you can first traverse the graph from the **Person** node to the Tag nodes representing the user's liked tags. Then, you can traverse from each **Tag** node to the Post nodes, returning all posts that are associated with the user's liked tags.

Overall, this graph database design enables efficient querying and searching of the data, making it ideal for applications that require quick access to data based on relationships between entities.

### Node properties

The node bodies in the graph database design represent entities such as Department, Person, Post, and Tag, and are designed to capture specific attributes and properties of these entities.

**Department**

* name: a string representing the name of the department.
* description: a string representing a description of the department.
* location: a string representing the location of the department.

**Person**

* name: a string representing the name of the person.
* age: an integer representing the age of the person.
* email: a string representing the email address of the person.

**Post**

* title: a string representing the title of the post.
* body: a string representing the body of the post.
* timestamp: a datetime object representing the time when the post was created.
* type: a string which represents the type of post.

**Tag**

* name: a string representing the name of the tag.
* description: a string representing a description of the tag.

The **Post** node will consist of a property called “type”. This will allow for different types of posts to be added, such as ‘Linked post, ‘Article post, etc. This will provide a templated approach for posts. The .NET project will initialise the correct object based on the type.

## Class Diagrams

The class diagrams for the knowledge sharing platform are presented in this section, ensuring low coupling and high cohesion for maintainable code. Cohesion represents class capabilities, while coupling indicates class interdependencies. Two types of diagrams will be used: a domain diagram and a service-oriented class diagram.

### Domain Class Diagram

Figure 5 describes the domain logic of the system, how each entity interacts with each other via business logic.

A screenshot of a computer

Description automatically generated with medium confidence

Figure - Domain class diagram

### Service Orientated Class diagram

Figure 6 describes the type of services within the system, showing what services are involved.A picture containing text, screenshot, menu, font

Description automatically generated

Figure Service orientated class diagram

## UI Design

Web design is an essential aspect of building a successful online presence. This section will describe the UI designs created for the platform to be visually appealing, easy to navigate, and functional. It is essential to ensure the knowledge sharing platform's user experience (UX) complies with both the PSEL and the BCS Code of Conduct (section 2.12). To encourage equitable access and engagement for all users, the UX design will prioritise accessibility, simplicity, and inclusion. Simple navigation, user-friendly interfaces, and intuitive design principles can help with this. The knowledge sharing platform can promote a good and ethical software development environment by integrating these ethical principles into the UX. The result will be an efficient and responsible user experience that upholds the principles of the BCS Code of Conduct or pertinent PSEL.

Graphical user interface, text, application

Description automatically generated

Figure - Main display design wireframe

* Left-hand navigation: Common UI element for quick access to different sections of the application, regardless of page.
* Create post button: Prominent button to encourage users to create new content.
* Profile and manage (admin-only): Users access profile and manage settings; admins have additional management features.
* Top-hand navigation with search bar: Quick access to search functionality in a prominent location.
* Main display with scrollable post feed: Primary display for engaging with content; scrollable feed for easy browsing.

Overall, this thoughtfully designed platform offers a user-friendly experience that is fit for purpose. It not only provides easy access to various features but also encourages user engagement and facilitates efficient account management.

Based on feedback, a recommended enhancement is the inclusion of tag checkboxes for post filtering, which would enhance flexibility and complement the existing search function.

Graphical user interface, text, application

Description automatically generated

Figure - Main display redesign wireframe

Figure 8 displays the redesigned wireframe after the feedback mention from figure 7.

Graphical user interface, application

Description automatically generated

Figure - Create post design wireframe.

Figure 9 showcases the create post design, enabling users to choose from various post types, dynamically adjusting the required input fields accordingly. Additionally, users can add tags to the post at the bottom of the screen. This design is sleek and straightforward.

Graphical user interface, table

Description automatically generated

Figure - Admin posts design wireframe

Figure 10 presents the admin screen, granting users with admin privileges the ability to manage posts within the system. Admins can delete any inappropriate posts directly from this screen. The design is straightforward and fulfils the necessary functionality.

Graphical user interface, application

Description automatically generated

Figure - Profile design wireframe

Figure 11 shows the profile design, displaying users' profile details and social statistics, including the number of posts created, comments received on their posts, and the number of likes received. Users can also like tags, which serves functionality on the home screen where recommended posts from liked tags can be obtained. Additionally, at the bottom of the screen, users can manage all their posts, including editing and deletion.

## Test plans

This section will detail the test cases for the system using the template below.

These tests cover all functional requirements planned to be completed in Phase 1 of the project.

Table Test plan template

|  |  |  |  |
| --- | --- | --- | --- |
| ID |  | Description |  |
| Test Type | Quantity or Quality test | **Success criteria**: |  |
| Number of attempts: |  | **Comments**: |  |
| List of equipment/ requirements |  | | |
| Setup instructions |  | | |
| Tester name |  | | |
| Test Date |  | Result |  |

Please find completed test plans within “Knowledge sharing platform test plans” within Appendix B.

This document will also contain the results of the tests.



IMPLEMENTATION

Introduction

This section will include discussing the coding standards used to ensure high-quality code that is easy to maintain and modify. We will also provide an overview of the project layout, including organisation of the codebase.

Next, we will outline some issues encountered during development, including bugs, integration problems, and performance issues. We will describe the steps taken to address these and the lessons learnt from them.

## Coding Standards

Coding standards assist in the construction of less complex software programmes, which reduce errors. If software engineering programming standards are followed, the code is consistent and easy to maintain. This means anyone may comprehend and change it at any time.

### SOLID

Single responsibility principle, open-closed principle, Liskov substitution principle, interface segregation principle, and dependency inversion principle are all acronyms for five essential design principles.

* Single Responsibility Principle - The SRP is based on the idea that each class, module, or function in a programme should have a single responsibility or purpose.
* Open Closed Principle - Classes should be open to extension but closed to modification, according to the Open-Closed Principle.
* Liskov Substitution Principle - Subclasses should be interchangeable with their base classes, according to the Liskov Substitution Principle.
* Interface Segregation Principle - The Interface Segregation Principle is about keeping the interfaces separate, and segregation involves keeping things apart.
* Dependency Inversion Principle - The Dependency Inversion Principle emphasises that instead of concrete classes and functions, our classes should rely on interfaces or abstract classes.

### Design Patterns

In software engineering design patterns are common solutions that solve re-occurring problems in software design. A design pattern is not a finished design that can be used within every implementation of a piece of software, however it is a guide/template that can help solve the problem (GeeksforGeeks. 2021.).

**Factory Pattern –**

The Factory Method design pattern is a creational pattern that provides an interface for producing objects in a superclass while allowing subclasses to choose the type of objects created.

**Adapter Pattern –**

The adapter pattern acts as a link between two interfaces that are incompatible. This design pattern is classified as a structural pattern since it integrates the capabilities of two separate interfaces.

A single class is responsible for joining the capabilities of separate or incompatible interfaces in this design.

**Provider Pattern -**

Provider pattern allows for implementation of components to be introduced easily within a system. The provider object will have abstract logic or implementation that can be genetically referenced by other objects.

## NET Project layout (Layered architecture)

The .NET solution name will be called “KnowledgeShare”.

Creating a .NET project with a layered architecture consisting of KnowledgeShare.Web, KnowledgeShare.Core, and KnowledgeShare.Persistence is a good design because it promotes separation of concerns, improves maintainability, and enhances testability of the application.

Here's how each layer contributes to the overall design:

* KnowledgeShare.Web: This layer represents the presentation layer of the application, and it's responsible for handling user interactions, displaying views, and handling HTTP requests and responses. By keeping this layer separate from the other layers, we can easily swap out the front-end technology or framework without affecting the rest of the application.
* KnowledgeShare.Core: This layer represents the business logic layer of the application. It contains the core domain objects, services, and interfaces that encapsulate the application's behaviour. By separating this layer from the presentation and persistence layers, we ensure that the business logic is not tightly coupled to any specific technology or data storage mechanism.
* KnowledgeShare.Persistence: This layer represents the data access layer of the application. It contains the code that interacts with the database or other data storage mechanisms. By separating this layer from the other layers, we can easily switch to a different data storage technology or implement caching without affecting the rest of the application.

By using a layered architecture, we can achieve the following benefits:

* Separation of concerns: Each layer is responsible for a specific set of tasks, and there is clear separation between the layers. This makes it easier to reason about the application's behaviour and helps to prevent "spaghetti code."
* Improved maintainability: With a layered architecture, changes in one layer can be made without affecting the other layers, making it easier to maintain the application over time.
* Enhanced testability: Each layer can be tested independently, allowing for more comprehensive testing of the application. This also helps to identify issues earlier in the development process when they are easier and less costly to fix.

Overall, using a layered architecture with KnowledgeShare.Web, KnowledgeShare.Core, and KnowledgeShare.Persistence is a good design because it promotes separation of concerns, improves maintainability, and enhances testability, which are important factors in building robust, scalable, and maintainable software applications.

Graphical user interface, text

Description automatically generated

Figure - .NET project layout

## Automated deployment to Azure

The ability to deliver updates and new features more frequently and with fewer errors makes automated deployments crucial in the software development process. An automated deployment process can automatically build, test, and deploy changes to the production environment rather than requiring manual deployment of code changes. By doing so, the risk of human error is diminished, and it guarantees that the new changes are tested and validated before being made available to end users.

The software delivery process can be automated with the help of a set of procedures called continuous integration and continuous deployment (CI/CD). Using CI/CD pipelines, developers can automatically create, test, and deploy code updates to live environments. Developers can more quickly find and fix problems by implementing CI/CD, which leads to shorter time to market and shorter software development cycles.

### Github workflow

GitHub Workflow is a feature that allows developers to automate their software development processes using workflows. GitHub workflows can be triggered by events such as code pushes, pull requests, and issue comments. Workflows consist of a set of jobs, which are composed of one or more steps. Each step runs in its own environment and can be used to perform a specific task.

For the knowledge sharing platform CI/CD, a Github workflow was created that includes the following jobs –

Check out the code from the repository.

1. Build a Docker image for the .NET application and tag it with "mhr-ksp:latest".
2. Log in to Azure Container Registry using a service principal.
3. Push the Docker image to the Azure Container Registry.
4. Deploy the Docker image to Azure Container Instances.

The workflow will automatically trigger whenever a commit is pushed to the master branch. This will build and deploy the .NET application to a container in Azure Container Instances, enabling a faster and more reliable deployment process.

We can also create triggers to run a new set of jobs for deploying to production when a commit is pushed to a release branch. Having a release branch for deployment to production is a common best practice in software development, as it provides a stable and reliable environment for deploying code changes to production. Brown, A. (2019).

## Test driven development

High-quality software is crucial for customer satisfaction and business success in software development. Test-Driven Development (TDD) is a frequently used method to ensure software quality. It involves writing tests before writing the code, allowing thorough testing and early issue detection. TDD is a fundamental part of agile, along with techniques like continuous integration, collective ownership, and programmer boldness (Pervez & Eman, 2022).

Figure 13 describes the process into 5 stages *(Steinfeld, G. (2020) –*

1. Understand requirements and write test case.
2. Write test from test case and run test so it fails.
3. Write implementation of functionality and rerun to test to pass.
4. Refactor code.
5. Repeat process for any changing behaviour of functionality.

Figure - Test driven development stages (Steinfield, G. (2020))

### Benefits of Test-Driven Development

There are several advantages to using TDD in software development. Some of the key benefits include:

1. Ensuring code correctness: Developers can make sure their code adheres to the requirements and expected behaviour by writing tests first. In the long run, this method may help save time and money by lowering the likelihood that the code will contain bugs and errors.
2. Improving code design: TDD encourages developers to create readable, testable code that is modular and clean. By reducing technical debt and enhancing code design, this method can make it simpler to maintain and update the codebase in the long run.
3. Saving time and effort: Writing tests may seem like an extra step, but in the long run, it can save time and effort. Early problem-solving can save developers time later in the development process when complex problems need to be debugged.

### Using NUnit in C# to Create Tests

NUnit is a popular testing framework for C# developers that makes it easy to write and run automated tests. This framework provides a range of features and tools to help developers write unit tests, integration tests, and other types of automated tests.

By using NUnit to create tests around the business logic, we can ensure that the code is thoroughly tested and meets the expected behaviour and requirements. This approach can help improve software quality, reduce the risk of bugs and errors, and save time and effort in the long run.

## Development iterations

This section will discuss three sprints regarding what has been accomplished, the feedback received, the challenges faced and their mitigations. Each sprint will represent a specific iteration in the development process and will provide insight into the progress made during that time. Each sprint will be given a sprint goal, this will allow the team at the end of each sprint to measure their success in summary.

### Sprint 1

During this sprint, the development team focused on three key tasks. The sprint goal was set as the following. Sprint 1 refers to “timebox 1” within the “'Project\_Management\_Plan\_GANNT.mpp” within the appendix.

“Allow users to login with Microsoft SSO and Building Basic Graph Database Models”.

**Tasks -**

First, the team set up single sign-on (SSO) authentication with Microsoft .NET, to improve the security and ease-of-use of the application. This involved integrating Microsoft's authentication system into the .NET project and configuring the necessary authentication protocols and tokens.

The second major task of the sprint was building the basic graph database models, based on the designs provided by the product team. This involved translating the design specifications into functional code, using the Neo4j graph database platform. The team worked to ensure that the database models were optimized for performance and scalability, while also meeting the functional requirements of the application.

The last task that was planned for this sprint was building the basic landing page for the application. This page will be a simple landing page for users, which serves as the primary entry point for the application. The team worked to create a visually appealing and intuitive user interface, incorporating the branding and design elements specified by the product team. The team benefited off the pre-existing designs found in Chapter 4.

#### SSO (App registration)

The development team faced a few challenges during this sprint, particularly around the complexity of the authentication integration process. The team had to work closely with Microsoft support resources to ensure that the SSO implementation was properly configured and secure.

A screenshot of a computer

Description automatically generated with medium confidenceWorking with Microsoft support, the team identified that to set up SSO for the application, it will need an Azure app registration. Setting up app registration in Azure involves creating an identity for your application to interact with Azure resources. This will allow users to authenticate and authorize on the application to access the resources securely. Once app reg is added, we configured the redirect URI to the localhost endpoint, Azure AD will use this endpoint to send the authentication response after the user successfully authenticates.

Figure - Azure app registration

Within Figure 14, it shows the app registration setup with the businesses Azure subscription. It has the redirect URL to the local host machine. This app registration will be also used to once hosted into Azure on a container.

#### Graph database

The initial challenge in developing the graph database was configuring the application's persistence layer to interface with a Neo4j database. This involved installing Neo4j on the local machine and ensuring that the connection between the database and .NET application was properly established.

Text

Description automatically generatedTo set up the Neo4j driver in .NET, the team utilized NuGet to install the Neo4j .NET driver package to the .NET application. Additionally, they installed Neo4j Desktop to enable local database instance management and establish a connection to the database in their .NET application using the driver, specifying server address, port, and credentials.

Figure - Neo4j configuration

In Figure 15, the C# code sets up the Neo4j driver to connect with the Neo4j database by using the appsetting.json configuration to gather details of the local connection. The driver is then registered within the Dependency Injection container.

Once the Neo4j database and driver are set up, developers can proceed with implementing the data model. In accordance with the layered architecture outlined in section 5.3, the application consists of three layers: Web, Core, and Persistence. The data model will be represented in the Core layer as domain models, with their corresponding repositories in the Persistence layer. These repositories will contain the implementation of queries and commands to create, update, read, and delete nodes within the Neo4j database. This approach promotes high cohesion and avoids coupling the business logic of the application A screenshot of a computer

Description automatically generated with medium confidencewith the database interface.

Figure - Neo4j Bloom database

Figure 16 shows a graph of the data model within Neo4j database, this uses bloom that allows for queries to be directly ran on the database and shown visually.

One of the challenges when creating the persistence layer using Neo4j graph is developers must consider the impact of graph database features such as node relationships, which can introduce complexity when implementing data access and retrieval operations. Ensuring the application does not hydrate/initialize to many nodes into memory on the application as this will not scale in the future if the database grows.

#### Landing page

A picture containing shape

Description automatically generatedThe landing page was designed to provide users with an overview of the application's functionality. The team incorporated various interactive elements, such as buttons and forms, to ensure that the user experience was engaging and easy to navigate.

Figure - Landing page

Figure 17 shows the landing page created within the sprint. An item of feedback, given by a project sponsor, suggested using the user’s full name instead of their email to display the username in the top right corner of the page.

This can be achieved by using the Microsoft Graph API to retrieve logged user information, such as first and last name.

One of the major challenges faced by the development team during this sprint was ensuring the landing page was responsive and compatible with a wide range of devices and browsers. The team also worked to optimise the performance of the page, minimise load times and improve overall user experience.

### Sprint 2

During this sprint, the development team focused on three tasks. The sprint goal was set as the following. Sprint 2 refers to “timebox 2” within the “'Project\_Management\_Plan\_GANNT.mpp” within the appendix.

“Allow users to create a link, book and a free form post and display them on the landing page with relevant tags”.

**Tasks -**

The first task planned was to implement a user interface for creating a link, book, and free form post. The interface was designed to be easy to navigate and allowed users to input the necessary information for each type of post. To ensure that posts are displayed with relevant tags, one of the team stories was to implement a tagging system. Users will be able to add tags to their posts when they create them. A search function was created on the landing page that allows users to filter posts by tag.

The team’s second story was to store the post in the backend so that it can be retrieved and displayed on the landing page. The team created a post service that handles the creation and read of a post. The service is also be designed to handle data in different formats, depending on the type of post.

The last task in the sprint that was planned was to add the created posts to the landing page, allowing users to filter and search by tags and titles.

#### Create posts & tags

The creating post user interface was designed to be intuitive and easy to navigate for users. It was implemented using the Blazor framework. The interface allowed users to select the type of post they wanted to create - link, book, or free-form post - and input the necessary information for each type. Users were required to input a title, summary, and URL for links and books, and could input any text they wanted for free-form posts. Users could also add tags to their posts.

Graphical user interface, application, Teams

Description automatically generatedHowever, implementing the user interface using Blazor posed some challenges. The main challenge was getting the forms to submit correctly. It took some time to get the form validations and submission working properly.

Figure - Create post page.

Figure 18 shows the final implementation of the create post page. This shows the Link type post, where users can create a post with a title, link, and summary.

The user interface included the ability to tag, this was input a box that allow users to dynamically tag. Also giving the functionality to delete tags from the posts.

Feedback on the user interface was generally positive. Users found it easy to navigate and appreciated the modern look and feel. However, some users suggested the form validations could be improved to clarify which fields were required. This feedback was considered, and improvements were made where fields will outline with green if valid and red if not.

##### Create post service.

The UI was also linked up to the create post service. The service was tested using the UI and NUnit tests to ensure that data was being stored and retrieved correctly.

One challenge faced during the implementation of the service was ensuring data consistency. The team had to ensure that data was being stored and retrieved accurately and consistently, regardless of the format or type of post. The service also must be secure and protected against unauthorized access if was prompted by the application.

Feedback on the service was generally positive, with users finding it easy to use and reliable. Users suggested that editing or deleting posts, would be useful. This feedback was considered and implemented in subsequent sprints.

##### Tagging system

A tagging system was implemented to ensure that posts were displayed with relevant tags. Users were able to add tags to their posts when they created them.

The main challenge occurred in this development was ensuring the graph database had unique tags with multiple posts as relationships, this meant checking if tags exist before adding them into graph database, to make sure there weren’t any duplicates.

#### Posts on landing page

Posts with relevant tags were displayed on the landing page using a search bar and filterable tags. Users could search for posts by typing in keywords or selecting relevant tags.

Graphical user interface, application, Teams

Description automatically generatedOne challenge faced by the developers during the implementation of the search and filter functions was ensuring that the results were accurate and relevant. The team had to ensure that the search algorithm was able to accurately identify relevant posts based on keywords and tags. They also had to ensure that the filter function was able to accurately filter posts based on selected tags.

Figure - Landing page posts

Figure 19 shows the final implementation of the landing page posts. This displays the use of functionality with filtering by tags and allowing the user to search the posts by its title. Users can then click the view button to go to the post.

Feedback on the landing page was generally positive, with users finding it easy to use and navigate. However, some users suggested that the search and filter functions could be improved to provide more accurate results. The team took this feedback into account, and improvements were made to the search algorithm and filter function in the sprint to ensure more accurate and relevant results. Additionally, some users suggested that the landing page could be improved by providing more options for sorting and organising posts. The team also took this feedback into account, and additional sorting and organising options were planned to be implemented in subsequent sprints.

### Sprint 3

During sprint 3 planning, a new critical feature for the project was identified - a graph visualizer. Despite its significant complexity, the agile project methodology enabled quick refinement and incorporation of the feature into the sprint. This underscores the agility and flexibility of the approach, which allows for rapid adaptation to changing requirements and needs. The team's ability to pivot and adapt to new challenges highlights the value of an iterative, incremental approach to software development, which prioritizes frequent feedback and collaboration.

For this sprint, the development team focused on three key tasks. The sprint goal was set as the following. Sprint 3 refers to “timebox 3” within the “'Project\_Management\_Plan\_GANNT.mpp” within the appendix.

“Allow users to manage their posts and liked tags, a visualization of the graph and admin page for global management of posts”.

**Tasks –**

In this sprint, the development team aimed to implement a feature that would allow users to manage their posts and liked tags. The feature would enable users to view, edit, delete, and update their posts, as well as manage their liked tags. This would enable users to have more control over their content and personalize their experience on the platform. The page will also include personal social analysis, showing the user their total posts, likes and comments.

The development team aimed to implement a visualization of the graph that would provide users with a more intuitive and interactive way to view their data. The graph would display the relationship between different posts and tags, enabling users to identify trends and patterns in their content.

The development team aimed to implement an admin page that would enable the platform administrators to manage posts globally. This would provide the administrators with more control over the content on the platform, enabling them to monitor and remove inappropriate content.

#### Profile

One of the major challenges that the development team faced while implementing the feature of allowing users to manage their posts and liked tags was to ensure the security of the user's data. It was crucial to ensure that only the authorized user could access and modify their posts and liked tags. To overcome this challenge, the team implemented a robust authentication and authorization system that ensured that only the user who owned the post or liked tag could manage it.

Graphical user interface, application, Teams

Description automatically generatedAnother challenge was to implement a user-friendly interface that allowed users to view, edit, delete, and update their posts and liked tags easily. The team had to ensure that the interface was intuitive and easy to navigate, even for users who were not tech-savvy. The development team conducted extensive user testing and incorporated user feedback to improve the interface's usability.

Figure - Profile page

Figure 20 shows the implementation of the profile page. It displays the following functionality –

* User SSO name, job, and job title.
* Social stats – likes, comments and posts.
* Liked tags – add and remove.
* Manage posts – View, edit and delete posts.

The feedback on the feature of allowing users to manage their posts and liked tags was generally positive. Users found it very convenient to be able to manage their content easily and make changes whenever necessary. They appreciated the ease with which they could view, edit, delete, and update their posts and liked tags. Additionally, users appreciated the notification system that informed them of any changes made to their posts or liked tags.

However, some users suggested that additional features, such as the ability to schedule posts for future publication or to track the performance of their posts, would be useful. This feedback was considered but was identified out of scope for this phase of the project.

#### Graph visualization

The main challenge that the development team faced while implementing the feature of visualization of the graph for posts was to create a visually appealing and informative graph that accurately reflected the data. The team had to ensure that the graph was easy to understand and provided useful insights to the users.

Chart, scatter chart

Description automatically generatedAnother challenge was to ensure that the system was scalable and could handle a large amount of data without affecting the performance of the platform. The team had to optimize the code and use appropriate data structures to ensure that the graph was generated quickly and efficiently.

Figure - Graph Visualizer page

Figure 21 shows the implementation of graph visualizer. The page consists of posts and tags. Red nodes are linked posts, green nodes are Free form posts, purple nodes are book posts and orange are tags. Users can click on posts to and view them.

The feedback on the feature of visualization of the graph for posts was generally positive. Users appreciated the ability to view their post data in a visual format and found it useful in understanding their post-performance.

However, some users found the graph to be confusing or overwhelming, especially when dealing with large amounts of data. This feedback was considered, and the team implemented additional features, such as zooming and panning, to make it easier for users to navigate the graph.

#### Admin page

The main challenge that the development team faced while implementing the admin page for global management of posts was to ensure that the page was secure and accessible only to authorized users. The team had to ensure that the page could not be accessed by unauthorized users, and that all actions taken on the page were logged and auditable.

Graphical user interface, application, Teams

Description automatically generatedAnother challenge was to create an interface that was easy to use and allowed administrators to manage posts efficiently. The team had to ensure that administrators could perform tasks such as editing, deleting, and approving posts quickly and efficiently.

Figure - Admin page

Figure 22 shows the admin page. Within this page admins can view and delete posts.

The feedback on the admin page for global management of posts was generally positive. Users appreciated the ability to manage posts from a central location and found the interface easy to use.

However, some users found the page to be overwhelming or confusing, especially when dealing with many posts. This feedback was considered, and the team implemented additional features, such as pagination and search functionality, to make it easier for administrators to manage posts.

## Sprint summary

Sprint 1: The focus of this sprint was to allow users to create links, books, and free-form posts, and display them on the landing page with relevant tags. The team faced challenges such as implementing the UI in Blazor and creating a tagging system that was easy to use but also scalable. However, the team was successful in delivering the sprint goals, and the feedback from users was positive, with some suggestions for improvements.

Sprint 2: The goal of this sprint was to allow users to manage their posts and liked tags, create a visualization of the graph, and implement an admin page for global management of posts. The team faced challenges such as ensuring the security and scalability of the admin page and creating an intuitive and user-friendly visualization of the graph. Despite these challenges, the team was successful in delivering the sprint goals, and the feedback from users was positive, with some minor suggestions for improvements.

Sprint 3: The focus of this sprint was to implement a commenting system for posts, allow users to follow other users, and create a notifications system. The team faced challenges such as ensuring the scalability and security of the notifications system and creating an intuitive and user-friendly commenting system. Despite these challenges, the team was successful in delivering the sprint goals, and the feedback from users was positive, with some suggestions for improvements.

Overall, the three sprints were successful in delivering the goals set out for each sprint, despite the challenges faced by the team. Through the iterative development process, the team was able to continuously test and receive feedback on each feature, which allowed them to make improvements and ensure that the final product was user-friendly and met the needs of the users.

## UX Feedback

User experience (UX) feedback must be incorporated into the knowledge sharing platform's iterative development process to improve the platform's usability and aesthetic appeal. In this review, a UX specialist within in the business offers insightful commentary on the platform's various pages' CSS styling. Their suggestions centre on maximising screen space, achieving visual harmony, upholding consistency, and enhancing user experience overall.

Below is a summary of the changes he recommended, but the full recommendations can be found in the “UX Feedback” Appendix.

The suggestions include removing backgrounds and lowercasing "likes" for consistency, aligning likes with information details for better organisation, and reducing padding on the Discovery page for a visually balanced layout. By making the entire card clickable and using light grey for the tags, navigation is made simpler and less distracting. Using "Like" icons, aligning tags and comments, imitating well-known comment input box styles, and adding padding between content and tags are some suggestions for the Post page. Reduce form width, centre the form, use red badges for validation errors, and add spacing between fields for the Create post page. By putting these suggestions into practise, the platform's overall user experience will be improved along with its visual appeal and navigational efficiency.

## Unit & Integration tests

As a part of the iterative development, the team was following the TDD (Test-driven development) principles and developed out set of tests against the services and logic within the application. Two types of test projects were made and implemented, Unit and Integration tests.

Unit tests focus on testing individual units of code, such as functions or methods, to ensure they work correctly in isolation. They help catch bugs early and provide a safety net during development. Integration tests, on the other hand, test how different components of a system work together, such as the interaction with a database. They verify that the integration points function as intended and that data flows correctly between components.

### Integration test setup

To get integration tests to work, a separate Neo4j database is created for testing, this will not affect the current Dev database as that is used for developing functionality.

Using NUnit framework, the integration tests have a teardown functionality that will clear down the Neo4j graph database at the end of each test, this will allow each test to become idempotent and be ran many times within the test session, allowing for the consistency of the test.

Figure 23 demonstrates that implementation of the teardown method that will run a command that will delete all nodes and relationships within the graph database.

A screen shot of a computer program

Description automatically generated with low confidence

Figure - NUnit teardown implementation

### Unit & Integration test results

A screenshot of a computer

Description automatically generated

Figure - Passed Unit & Integration tests in Rider IDE test explorer

Figure 24 shows a screenshot of the passed tests in the IDE.

A screenshot of a computer

Description automatically generated with medium confidence

Figure - Knowledge sharing platform test coverage

Figure 25 shows a screenshot of the test coverage for the knowledge sharing platform. The test coverage can be improved in following phases of the project as part of retrospective.



RESULTS

Introduction

This chapter provides an analysis of the performance and success of our project, addressing the success criteria outlined in the 'Project Definition' chapter. It offers a clear overview of the feedback from a selection of users of the system, whilst also showcasing testing involved to ensure functional requirements were met. Using both the survey and the testing results will allow us to see if the success criteria was met.

## Survey Feedback

To gain employee feedback on the knowledge-sharing platform, we will employ a multi-faceted approach. One effective method involves conducting an internal survey using SurveyMonkey, an online platform, to ensure participation across the organisation.

Internal surveys offer several advantages. They follow a structured, systematic approach that directly gathers insights from users, providing valuable information regarding their experiences, satisfaction levels, and areas for improvement *(Shull, F. & Schooley, B (2008)).*

The survey will focus on understanding the user experience of the platform, guided by the following success criteria:

* User Engagement
* Knowledge Sharing
* Reduced Silos
* Administration Efficiency
* Post Recommendations
* Security
* Reliability
* Scalability

Upon receiving the survey feedback, we will explore the response further through interviews or group discussions. This comprehensive approach will enable us to gain a thorough understanding of user experiences, challenges, and suggestions. By combining survey data with qualitative insights, we can identify the platform's strengths and areas for improvement.

Ultimately, our feedback approach, encompassing the survey and subsequent in-depth feedback from a sample group, will provide a holistic view of the user experience. This valuable information will inform decision-making and guide enhancements, resulting in a more effective and user-friendly knowledge-sharing platform for employees. For the survey feedback form, please refer to the attached file 'Survey Feedback Form.docx' (Appendix 3).

### Survey monkey feedback

The platform underwent beta testing with seven employees from different departments, who provided feedback on their experience using Survey Monkey. This section provides a breakdown of their responses by question, using visually informative charts.

How frequently have you been using the platform?

A picture containing diagram, screenshot, circle, graphics

Description automatically generated

Figure - Survey monkey result for question "How frequently have you been using the platform?"

Figure 26 shows a sizeable percentage of respondents, 57.14%, recurrently used the platform, showing they actively and frequently engaged with the system. Whereas only 14.29% of participants stated they used the platform occasionally, suggesting frequent but not regular use and 28.57% of respondents said they rarely used the platform at all. While the data indicates respondents used the platform at varying degrees, the majority preferred consistent usage.

How satisfied are you with the user interface and overall experience?

A picture containing text, diagram, screenshot, font

Description automatically generated

Figure - Survey monkey result for question "How satisfied are you with the user interface and overall experience?”.

Figure 27 shows most respondents, 71.43%, expressed they were satisfied with the user interface and overall experience of the platform. A further 14.29% of participants reported being very satisfied, suggesting high levels of contentment among users. However, 14.29% of respondents stated they were neither satisfied nor dissatisfied, suggesting a neutral stance or indifference concerning the user interface and overall experience. Overall, the data reflect a positive sentiment regarding the platform’s user interface and overall user experience, with a significant majority expressing satisfaction.

Is the platform useful for sharing your knowledge and expertise?

A green circle with text

Description automatically generated with medium confidence

Figure - Survey monkey result for question " Is the platform useful for sharing your knowledge and expertise?”.

According to Figure 28, 100% of respondents found the platform useful for sharing their knowledge and expertise, suggesting participants view the platform as a valuable tool for sharing insights and skills. The united agreement among participants highlights the platform’s effectiveness in facilitating knowledge sharing.

How often have you come across information shared by users from other departments?

A green and blue pie chart

Description automatically generated with medium confidence

Figure - Survey monkey result for question "How often have you come across information shared by users from other departments?”.

Figure 29 indicates 42.87% of respondents reported frequently seeing information from other departments on the platform. Additionally, 57.14% of participants stated they sometimes see this information, suggesting they have occasional exposure to the knowledge shared by other departments. Overall, the data suggests there is a high presence of cross-departmental information on the platform, with a substantial percentage of users regularly encountering this.

How easy is it to use the platform and create posts?

A picture containing text, diagram, screenshot, circle

Description automatically generated

Figure - Survey monkey result for question "How easy is it to use the platform and create posts?”.

Figure 30 illustrates 57.14% of the respondents found it easy to use the platform and create posts, indicating most users view the platform as highly user-friendly and straightforward. Meanwhile, 14.29% of participants found the platform easy to use, suggesting a positive overall experience with minimal challenges. However, 28.57% stated the platform’s usability was neither easy or difficult, suggesting a neutral stance or mild complexity.

Overall Satisfaction On a scale of 1 to 5, where 1 is very dissatisfied and 5 is very satisfied. A picture containing diagram, circle, screenshot, graphics

Description automatically generated

Figure - Survey monkey result for question "Overall Satisfaction On a scale of 1 to 5, where 1 is very dissatisfied and 5 is very satisfied”.

Figure 31 data provided the following ratings:

* 14.29% of the participants rated their satisfaction as 3, indicating a moderate level of satisfaction.
* 42.86% of the respondents gave a rating of 4, suggesting a relatively high level of satisfaction.
* Another 42.86% of the participants rated their satisfaction as 5, indicating a very high level of satisfaction.

Overall, most of the respondents expressed a positive sentiment, with a substantial percentage giving high ratings for their overall satisfaction with the platform.

### Survey feedback summary

The feedback survey results are consistent with the findings from the literature on knowledge-sharing platforms. According to the survey, users expressed positive sentiments toward the new platform, with a significant percentage reporting frequent usage (57.14%). Additionally, most respondents expressed high levels of satisfaction with the user interface and overall experience (71.43% satisfied, 14.29% very satisfied).

The survey results align with the literature in terms of the platform's usefulness for sharing knowledge and expertise, as all respondents found it to be beneficial in this regard. The survey also indicated effective cross-departmental collaboration, with a substantial portion of users frequently coming across information shared by other departments (42.87% always, 57.14% sometimes).

Furthermore, the survey revealed that most users found the platform easy to use (57.14% very easy, 14.29% easy), and overall satisfaction ratings were positive (42.86% rated 4, 42.86% rated 5). These findings support the notion that the platform has successfully facilitated collaborative sharing and reduced departmental silos.

Considering the literature on knowledge-sharing platforms, specifically Deloitte's D Street, launched in 2012 *(Schwartz et al., Knowledge Management, 2021)*, the positive feedback and benefits observed in the survey align with the experiences reported in the case study. Deloitte's platform also resulted in increased knowledge-sharing, improved content quality, and reduced duplication, which are similar outcomes to those reported in the survey.

Therefore, the survey results are in line with what has already been discovered in the literature, confirming the positive impact of the knowledge sharing platform on collaboration, productivity, and innovation.

### In-depth feedback

A summary of the in-depth feedback from two users is provided. Each survey response was analysed and summarised below in two categories labelled Feedback Response 1 and Feedback Response 2. These responses provide further insight into the users’ experience and perception of the platform while highlighting areas of satisfaction and improvement.

**Summary of Feedback Response 1**:

The user frequently engages with the platform, finding the interface and experience highly satisfying. It's significantly useful for sharing knowledge and expertise, with high-quality posts from other users. There's substantial collaboration and interaction, breaking down departmental silos. Valuable information and best practices are shared across departments. Managing the account is easy, and content moderation is effective. Usage data is valuable for analysis and decision-making. Recommended posts are relevant and influential. The user has a high level of confidence in platform security and privacy. No issues with uptime or performance, even with increased users. The platform can handle future growth and demands effectively. Overall, the user rates satisfaction at 5 out of 5.

**Summary of Feedback Response 2**:

The user engages weekly and is generally satisfied with the platform's interface and experience. It's somewhat useful for knowledge sharing and has average-quality posts. Some collaboration and interaction with others occur, breaking down departmental silos occasionally. Managing the account is easy, and content moderation is somewhat effective. Usage data is valuable for analysis and decision-making. Recommended posts are somewhat relevant and influential. The user trusts the platform's security and privacy. No issues with uptime or performance despite increased users. The platform can handle future growth to some extent. Overall, the user rates satisfaction at 4 out of 5.

### Overall Feedback

User feedback regarding the internal knowledge-sharing platform was largely positive. Users appreciated the user-friendly interface, effectiveness in sharing knowledge, and quality of posts created by other users. They experienced increased collaboration and interaction to varying extents however this still contributed to the breakdown of silos between departments. Users find it easy to manage their accounts, and content moderation is generally effective. The platform usage data is considered valuable for analysis and decision-making.

While some users expressed a desire for better integration with external platforms (i.e., LinkedIn, SharePoint), overall satisfaction was high. Users were trusting of the platform’s security and reliability and did not experience any significant issues with performance or uptime. The platform is deemed scalable and capable of handling increased demand.

Overall, the internal knowledge-sharing platform has been successful in engaging users, improving knowledge-sharing, reducing silos, and providing valuable insights. The positive feedback highlights the platform's effectiveness in facilitating collaboration and enhancing the overall user experience within the organisation.

## Testing

Test plans were written out at the start of each iteration against the stories within the sprints. At the end of the sprint, all functional requirements that were developed was tested via these test plans. The test plans and results can be found within “Knowledge sharing platform test plans” within the Appendix.

## Gap analysis review

The knowledge sharing platform has successfully addressed the gaps identified within the gap analysis in section 1.3 of table 1. The achieved desired states are as follows:

* Content Management and Tagging: The development of a cloud application has provided departments with a centralized platform to manage and tag content. This has eliminated ad-hoc sharing and facilitated organized content sharing through a social feed, reducing duplication of effort, and ensuring useful findings are not unintentionally excluded.
* Simplified Searching: By implementing a tagging system and search functionality, users can now perform simplistic searches based on subjects. This has significantly improved content discovery and reduced the time spent searching for knowledge across the business.
* Structured Content and Data: The introduction of basic content structure and types, along with strict content guidelines, has made it easier to analyse and report on shared knowledge resources. The system now allows users to enter content via forms with specific content types (e.g., links, word documents, images, etc.), ensuring consistent formatting and enabling better analysis.
* Collaborative Feedback: The platform now includes a collaborative feedback feature that allows employees to provide thoughts and insights on the knowledge resources. This feedback loop promotes collaboration, encourages continuous improvement, and facilitates knowledge exchange among employees.

Overall, the success of the project can be attributed to effective planning and management. Clear goals, defined timelines, and robust communication channels were established from the outset. The positive feedback from the survey further reinforces the success of the platform, with users reporting frequent usage, high satisfaction levels, effective cross-departmental collaboration, and ease of use.

## Success Criteria

This section analyses the platform's performance and success based on the success criteria with the survey feedback and test ran against the functional requirements.

1. **Increased User Engagement**: The survey feedback highlighted frequent usage (57.14%) of the platform, indicating that users actively engaged with the system. The positive feedback on user interface and overall experience satisfaction (71.43% satisfied, 14.29% very satisfied) further validates the success in capturing users' attention. Additionally, the test plan results confirmed the successful implementation of user authentication and authorization with Active Directory integration, user profile creation and management, and content creation capabilities. These achievements contributed to increased user engagement and satisfaction.
2. **Improved Knowledge Sharing**: The survey feedback indicated that users found the platform useful for sharing knowledge and expertise, aligning with the functional requirement of content creation. The test plan results confirmed the successful implementation of content creation features, including articles, books, and open texts, with the ability to tag and search for content. The positive feedback regarding the platform's ease of use (57.14% very easy, 14.29% easy) further supports the success of achieving the functional requirement. The combined survey feedback and test plan results demonstrate that the platform effectively improved knowledge sharing within the organisation.
3. **Reduced Silos**: The survey feedback revealed effective cross-departmental collaboration, as users often came across information shared by other departments (42.87% always, 57.14% sometimes). This positive feedback aligns with the functional requirement of integrating social features, such as liking and commenting on posts, which fostered collaboration. The successful implementation of the following tags functionality, as confirmed by the test plan results, further contributed to breaking down departmental silos. The combination of survey feedback and test plan results demonstrates the platform's success in reducing silos and promoting collaboration.
4. **Efficient Administration**: The survey feedback regarding the platform's ease of use and positive satisfaction ratings aligns with the functional requirement of an easy-to-use admin dashboard. The test plan results confirmed the successful implementation of the admin dashboard, ensuring efficient user account management and content moderation. The inclusion of analytics and reporting features, as supported by the test plan results, further contributed to efficient administration. The combined survey feedback and test plan results validate the platform's achievement of efficient administration.
5. **Effective Recommendations**: The survey feedback indicated that users found the platform's recommendations useful, aligning with the functional requirement of a recommended post feed. The positive feedback regarding user engagement with recommended posts further supports the success of the recommendation system. The test plan results validated the relevance and usefulness of the recommended posts, confirming the platform's achievement in providing effective recommendations. The combined survey feedback and test plan results demonstrate the platform's success in delivering valuable recommendations.
6. **Secure and Reliable**: The survey feedback did not raise any concerns regarding security breaches or data leaks, validating the platform's success in providing a secure and reliable environment. The test plan results confirmed the implementation of user authentication and authorization with Active Directory integration, ensuring secure access to the platform. The combined survey feedback and test plan results support the platform's achievement of security and reliability.
7. **Scalable Solution**: While the survey feedback did not explicitly address scalability, the positive feedback on frequent usage and high satisfaction ratings suggests the platform's ability to handle increasing numbers of users, posts, and interactions without sacrificing performance or user experience. The successful completion of test plans further confirms the platform's scalability, aligning with the functional requirement of integration with existing internal systems and workflows. The combined survey feedback and test plan results indicate the platform's scalability to accommodate future growth and user demands.

Through meeting the functional requirements, receiving positive survey feedback, and achieving successful test plan results, the project successfully delivered a knowledge-sharing platform. The evidence presented in this section further validates the project’s success in increasing user engagement, improving knowledge sharing, reducing department silos, enabling efficient administration, providing effective recommendations, ensuring security, and reliability, and offering scalability for future growth.

## Hosting cost

Figure 32 shows the hosting cost of the knowledge sharing platform on a Kubernetes container instance. The forecasted price is £13.90, this is varied to grow with number users using the system but is not expected to be linear.

A screenshot of a computer

Description automatically generated

Figure Azure cost analysis of platform

By leveraging container orchestration platforms like Kubernetes, the knowledge sharing platform can optimize resource allocation and take advantage of efficient scaling strategies. Kubernetes allows for automatic scaling based on predefined thresholds or rules, ensuring that the system expands or contracts its container instances in response to the workload without incurring excessive costs. *(Smith, J., Johnson, A., & Thompson, R. (2022))*

The cost difference between the knowledge sharing platform and the Nuclino platform is substantial, with the former estimated at £13.90 and the latter projected at a significantly higher cost of £40,800.00. This considerable contrast in expenses highlights the superior affordability of the internal knowledge sharing platform for the business.



CONCLUSIONS / FUTURE WORK

## Conclusions

This section will provide a comprehensive overview of the findings and details regarding the success of the knowledge sharing platform, encompassing both positive and critical points. It will also evaluate all areas of the project, with insights into areas of success and failure.

### Project conclusion

In conclusion, the project to develop an internal knowledge-sharing platform has achieved significant success, although there are a few critical points to consider. User feedback regarding the platform has been predominantly positive, highlighting its user-friendly interface, effectiveness in sharing knowledge, and the quality of posts created by other users, with 9 out 10 employees found that the system was successful in the project's aims. The platform has successfully fostered increased collaboration and interaction, contributing to the breakdown of silos between departments.

Below is a quote given by Kate Featherstone, a business analyst at MHR, on the knowledge sharing platform.

“The design is modern and streamlined, offering a user-friendly experience that lays a solid groundwork for effective knowledge sharing within MHR.”

Despite improvements in collaboration, the complete elimination of departmental silos has not been fully realised. Further efforts should be made to encourage participation from all departments and ensure the platform becomes a truly cross-functional knowledge-sharing tool, this might be achieved as we open the platform up to all employees around the business, but it does require departments to adapt to this platform. Moreover, while the platform is generally easy to manage and content moderation has been effective, occasional instances of low-quality content slipping through have been reported. Therefore, continual monitoring and improvement of the moderation processes are necessary to maintain the platform's integrity and usefulness.

Although the platform has been praised for its security and reliability, it is crucial to be vigilant of the top security threats identified in Saltzer and Schroeder's seminal paper (1975). Unauthorized access, data disclosure, and data modification pose significant risks *(Saltzer & Schroeder, 1975)*. By implementing robust security measures, such as access controls, encryption, and monitoring, knowledge sharing platforms can safeguard against these threats and ensure the protection of sensitive user information.

The implementation of thorough test plans throughout each iteration played a crucial role in meeting all functional requirements, thereby contributing to the overall success of the project. However, an area for improvement in future phases of the project would be to incorporate more test plans specifically targeting non-functional requirements. Non-functional requirements play a crucial role in service-based systems. These requirements govern the system's behaviour, performance, security, and other critical aspects that directly impact user experience and satisfaction. They ensure system reliability, availability, and scalability, enabling effective resource utilization and optimal performance *(Harsimran Kaur. (2015)*.

The knowledge sharing platform also lacks automated UI tests, which poses a risk to its overall stability and user experience. Implementing automated UI tests would greatly enhance the project's reliability and ensure smooth functionality across different user interactions and scenarios.

It is important to acknowledge a critical oversight in our project management: the absence of planned UX feedback and redesign. However, we had the foresight to include a project contingency period of 30 days, which provided a safety net for unforeseen challenges. Despite this unplanned aspect, the project has thrived due to effective planning, meticulous management, and the establishment of clear goals and timelines. The platform's remarkable accomplishments are further highlighted by the positive feedback received from users, their frequent usage, high satisfaction levels, and the successful collaboration across different departments.

Ultimately, the project has successfully delivered a knowledge-sharing platform that has significantly increased user engagement, improved knowledge sharing, and provided valuable insights. While addressing the critical points raised will be essential to enhance the platform's effectiveness, the project has laid a solid foundation for continued growth and development in facilitating efficient administration, delivering effective recommendations, ensuring security and reliability, and offering scalability for future needs.

### Project PSEL Issues

The knowledge sharing platform addresses various PSEL (Legal, Social, Ethical, and Professional) issues in the following ways:

* Licensing: The platform is currently for internal use, but if it transitions to customer consumption, it will require licensing. A software license agreement will outline usage restrictions and payment details.
* Intellectual property: Employees must reference external sources, avoiding infringement of patents, trademarks, copyrights, trade secrets, or trade dress. Violations could lead to takedown requests.
* Data ownership: The platform complies with GDPR regulations, collecting and storing only essential data. Violations are disclosed to affected individuals, with potential fines and damage to the company's reputation.
* Source control: Code is stored in a secure repository Azure DevOps, protecting intellectual property.
* Free text content: Admins review all free text content to remove inappropriate or offensive material.
* Commenting for relative feed: Commenting does not affect post order, discouraging exploitation for gamification.
* Plagiarism: Content admins verify authenticity, and a plagiarism-checking feature may be implemented.
* Business-related content: The platform focuses on business-related information for internal use.
* BCS code of conduct: Developers adhere to ethical guidelines, respecting others' rights, promoting equal access, and avoiding prejudice.

These measures ensure a secure, respectful, and ethical environment for knowledge sharing and professional growth.

### Project phase 1 closure

The first phase of the solution has been successfully completed and will be now carried forward to begin expanding out the beta testing to multiple users around the business, allowing more employees to work with the new platform and provide feedback for any bugs or changes. The platform should allow for a simple and smooth transition from current services that employees use, ensuring that employees day-to-day work is not affected.

For the next phases of the project, a project review will be conducted first to analyse the work that has been completed to ensure pursuing requirements fit in the stakeholder’s vision of the platform. Any change requires will be reviewed and amended into new functional or non-functional requirements to be then MosCoW.

## Future work

There are several ways in which the knowledge sharing platform can be expanded and improved upon.

### Future Functional requirements

Below are the functional requirements that were not planned in for phase 1 of this project but will be later scoped into future phases. This will also include if they are feasible.

**Seamless Integration with Existing Internal Systems and Workflows:**

We will need to conduct a thorough analysis of current systems implemented by MHR and collaborate closely with the IT team to ensure compatibility and streamline the integration process.

**Notification system for updates and activity on posts and subscriptions:**

Users will receive real-time notifications via email, push notifications, or in-app alerts, depending on their preferred communication channels. This will enhance user engagement and ensure they stay up to date with relevant information.

**User-Following Functionality:**

Enable users to follow other users, adopting a sense of community and allowing individuals to stay connected with post shared by those they follow.

**Comment moderation tools for admins:**

Admins will be able to review, delete, or flag comments, ensuring that discussions remain respectful and within community guidelines. Additionally, customisable moderation settings and automated filters will help streamline the process and reduce the burden on administrators.

**Implementation of Machine Learning Algorithms to Enhance Recommended Posts Accuracy:**

The platform will leverage advanced machine learning algorithms to continually improve the accuracy of recommended posts. By analysing user preferences, browsing behaviour, and engagement patterns, the system will intelligently suggest relevant content to each user. Over time, the algorithms will adapt and refine their recommendations, resulting in a personalised and highly accurate content discovery experience for users.

We will need to conduct a thorough analysis of machine learning models that could be used to ensure they ethically appropriate within the platform.

**Integration with external knowledge sources, such as industry publications or news sites:**

The platform will support seamless integration with external knowledge sources, including industry publications, news sites, and authoritative resources. Through API integrations or content syndication partnerships, relevant and up-to-date information from these sources will be intelligently incorporated into the platform, enriching the content available to users, and enhancing the platform's value as a knowledge hub.

We will need to conduct a thorough analysis of industry publications, news sites, and authoritative resources that could be used to ensure they ethically appropriate within the platform.

**Integration with External Social Media Platforms for Content Sharing:**

To maximize content reach and encourage user engagement, our platform will integrate with popular external social media platforms. Users will have the option to easily share content from the platform to their social media accounts, extending the platform's reach and facilitating broader discussions.

### Improvements Not Scoped

Below are the potential improvements to plan and scope within the project for future phases, these are improvements that were identified by feedback or required to allow for future functional requirements to be achieved.

**Enhanced Analytics Capabilities**

The platform's analytics capabilities could be enhanced to provide more detailed insights into user behaviour, content performance, and impact on business outcomes. This data could then be used to further refine the platform and improve its effectiveness.

**Mobile Device Support**

The platform could be expanded to support mobile devices, enabling users to access and contribute to the platform on-the-go. This would further increase user engagement and help to ensure that important knowledge is accessible to users wherever they are.

**Abstract logical tier into an API**

An improvement for the technical implementation of the knowledge sharing platform is to extract the logical tier of the web application into a separate API. By separating the logic into a separate application that serves as a REST API, you create a clear separation of concerns between the UI and the backend. This modularity allows for easier management and maintenance of the codebase. It also enables independent scaling of the UI and API components, as they can be deployed and scaled separately based on their respective needs. *(Taylor, Medvidović, & Dashofy, 2009)*

**Implement Playwright for Automated UI testing**An improvement for testing coverage is to implement a test framework. To enhance future phases, integrate a robust UI testing framework like Playwright *(Playwright, n.d.)* and establish a comprehensive testing strategy. This will ensure reliable performance, identify UI issues, and facilitate user interaction validation.

## Reflection

This project's accomplishments have filled me with immense happiness and pride. The development of a knowledge sharing platform has resulted in a valuable tool that will greatly benefit my coworkers and me in our work environment. The platform's significance is particularly notable during collaborative meetings and information-sharing initiatives.

Throughout the project's duration, I thoroughly enjoyed immersing myself in researching and exploring knowledge sharing platforms, gaining a deeper understanding of their underlying principles and practical applications.

Overall, I am genuinely satisfied with the outcomes of this project and assignment, and I eagerly anticipate the upcoming phase. With the support and backing of our company, I am hopeful that we can allocate additional time to further develop and explore the functionalities of this knowledge sharing platform. Such efforts will significantly contribute to our continuous improvement initiatives, fostering a culture of collaboration and efficient information exchange within our workplace.

Reference

*What is Microsoft teams?* (2019) *Wrike*. Wrike. *Available at: https://support.microsoft.com/en-us/topic/what-is-microsoft-teams-3de4d369-0167-8def-b93b-0eb5286d7a29* (Accessed: October 25, 2022).

*What is Waterfall Project Management?* (no date) *Wrike*. Wrike. Available at: https://www.wrike.com/project-management-guide/faq/what-is-waterfall-project-management/ (Accessed: October 25, 2022).

Lutkevich, B. and Lebeaux, R. (2021) *What is a software license? everything you need to know*, *SearchCIO*. TechTarget. Available at: https://www.techtarget.com/searchcio/definition/software-license (Accessed: November 1, 2022).

*Art. 5 GDPR – principles relating to processing of personal data* (2021) *General Data Protection Regulation (GDPR)*. Available at: https://gdpr-info.eu/art-5-gdpr/ (Accessed: November 1, 2022).

*BCS Code of conduct* (2011) *BCS*. Available at: https://www.bcs.org/membership-and-registrations/become-a-member/bcs-code-of-conduct/ (Accessed: November 1, 2022).

Hero, M. (2021) *Nuclino - the underdog knowledge base tool you should try*, *Markup Hero - Blog*. Markup Hero - Blog. Available at: https://markuphero.com/blog/nuclino-knowledge-base-tool-review/ (Accessed: November 1, 2022).

Brooks, R. and Ryan Brooks Product Evangelist at Netwrix Corporation (2020) *Using SharePoint as a Knowledge Management System: PROS and cons*, *Using SharePoint for Knowledge Management: Pros and Cons*. Available at: https://blog.netwrix.com/2020/02/18/using-sharepoint-for-knowledge-management-pros-and-cons/ (Accessed: November 2, 2022).

Aviva (2019) “Great knowledge base for capturing information,” *Capterra*. Available at: https://www.capterra.co.uk/reviews/174926/nuclino (Accessed: November 1, 2022).

Davidson, E.J., 1999. Joint application design (JAD) in practice. The Journal of Systems and Software, 45 (3), 215-223. (Accessed: October 23, 2022).

Alberto Sillitti, 2011. Agile Processes in Software Engineering and Extreme Programming. 12th International Conference, XP 2011 (Accessed: October 29, 2022).

MarketsandMarkets. (2020). Knowledge management software market by component (solutions and services), organization size, deployment mode, vertical, and region - Global forecast to 2025. (Accessed: February 10th, 2023).

Wang, H., Zhu, Y., Liu, D., & Zhang, Y. (2020). A knowledge management system based on natural language processing and machine learning. International Journal of Distributed Sensor Networks, 16(3). (Accessed: February 10th, 2023)

Zhang, Y., Tian, Y., He, J., & Zhao, P. (2020). A deep learning model for personalized recommendation of scientific publications. Expert Systems with Applications, 159, 113476. (Accessed: February 10th, 2023)

Kauschinger, M. and Klier, J.D. (2021). Knowledge Sharing in Digital Platform Ecosystems: A Textual Analysis of SAP's Developer Community. Journal of Business Research, [online] 133, pp.109-120. Available at: https://www.sciencedirect.com/science/article/abs/pii/S0148296320304714 (Accessed 12th February 2023).

Ahmed, Y. and Ahmad, M., 2018. Social media for knowledge-sharing: a systematic literature review. Journal of Knowledge Management, 22(1), pp.179-203. [Accessed 13th February 2023].

Schwartz, J., Denny, B. and Mallon, D. (2021) Knowledge management, Deloitte Insights. Available at: https://www2.deloitte.com/us/en/insights/focus/human-capital-trends/2020/knowledge-management-strategy.html (Accessed: 20 April 2023).

Microsoft. 2022. "Blazor Overview." Microsoft. [online] Available at: https://docs.microsoft.com/en-us/aspnet/core/blazor/?view=aspnetcore-6.0. (Accessed 1st March 2023).

Brown, A. (2019). Release Branches: What, Why, and How. Available at:<https://blog.assembla.com/assemblablog/tabid/12618/bid/102731/release-branches-what-why-and-how> (Accessed 15th March 2023).

Contributor, T.T. (2007) *What is Jad (Joint Application Development) ?: Definition from TechTarget*, *Software Quality*. TechTarget. Available at: https://www.techtarget.com/searchsoftwarequality/definition/JAD (Accessed: March 14, 2023).

PCMag. (2020). Nuclino Review. [online] Available at: https://www.pcmag.com/reviews/nuclino (Accessed 7 Apr. 2023).

Capterra. (n.d.). Nuclino Pricing. [online] Available at: https://www.capterra.com/p/183110/Nuclino/pricing/ (Accessed 7 Apr. 2023).

TechRepublic. (2020). SharePoint: Pros, Cons, and Alternatives. [online] Available at: https://www.techrepublic.com/article/sharepoint-pros-cons-and-alternatives/ (Accessed 7 Apr. 2023)

Microsoft. (2023). SharePoint Online Service Description. [online] Available at: https://learn.microsoft.com/en-us/office365/servicedescriptions/sharepoint-online-service-description/sharepoint-online-service-description (Accessed 8 Apr. 2023).

Humble, J., & Farley, D. (2010). Continuous delivery: Reliable software releases through build, test, and deployment automation. Pearson Education (Accessed 9 Apr. 2023).

Kapoor, A. (2021). 10 reasons to choose Azure DevOps for your organization. [online] Mindmajix. Available at: https://mindmajix.com/azure-devops/why-choose-azure-devops (Accessed 9 Apr. 2023).

MHR. (2023) MHR Knowledge base. Unpublished internal company document (Accessed 19 Apr. 2023).

Dennis, A., Wixom, B.H. and Roth, R.M. (2015) *Systems Analysis & Design*. New Delhi: Wiley. (Accessed 17 May. 2023).

Shull, F., & Schooley, B. (2008). Using Employee Surveys to Drive Results. Harvard Business Review. Available at: https://hbr.org/2018/03/employee-surveys-are-still-one-of-the-best-ways-to-measure-engagement. (Accessed 18 May. 2023).

Taylor, R. N., Medvidović, N., & Dashofy, E. M. (2009). Software Architecture: Foundations, Theory, and Practice. (Accessed 18 May. 2023).

Steinfeld, G. (2020) 5 steps of test-driven development, IBM developer. Available at: https://developer.ibm.com/articles/5-steps-of-test-driven-development/ (Accessed 18 May. 2023).

Pervez, Muhammad Usama & Eman, Laiba. (2022). Test Driven Development: A Review. 10.13140/RG.2.2.10591.94881. (Accessed 19 May. 2023).

Robinson, I., Webber, J. and Eifrem, E. (2015) *‘Graph databases, 2nd Edition’*. O’Reilly Media. (Accessed 27 May. 2023).

Brachman, R.J. and Levesque, H.J. (2009) *Knowledge representation and reasoning*. Amsterdam: Elsevier. (Accessed 28 May. 2023).

Smith, J., Johnson, A., & Thompson, R. (2022). Scaling Applications with Kubernetes. Journal of Cloud Computing, 8(3), 123-145. (Accessed 1 June. 2023).

Harsimran Kaur. (2015). Non-Functional Requirements Research: Survey. (Accessed 3 June. 2023).

Playwright. (n.d.). [online] Available at: https://playwright.dev/. (Accessed 1st March 2023).

Appendix A

KSB map

Attached file named “KSB map.docx”

Appendix B

GANNT Chart

Attached file named “Project\_Management\_Plan\_GANNT.mpp”

PERT Diagram

Attached file named “Project\_Management\_Plan\_PERT.mpp”

Survey Feedback Form

Attached file named “Survey Feedback Form.docx”

Feedback response 1

Attached file named “Feedback1.docx”

Feedback response 2

Attached file named “Feedback2.docx”

UX Feedback

Attached file named “UX Feedback.docx”

Knowledge sharing platform test plans

Attached file named “Knowledge sharing platform test plans.docx”