

Skill Boost: Personalised Content Recommendations for Student Skill Enhancement

CS310 Final Report

Lucy Inett

30th April 2024

Abstract

This report presents the development and evaluation of the Skill Boost System, a recommendation platform designed to facilitate personalised learning and skill acquisition for Computer Science. The project identified a unique opportunity to enhance users' educational experiences through tailored suggestions. Employing an Agile methodology, the project meticulously navigated through the stages of design, implementation, and evaluation, adhering to a clear set of objectives and requirements informed by extensive research and user needs analysis.

A key focus of the project was the integration of advanced Natural Language Processing (NLP) techniques to accurately interpret user inputs. The use of content-based filtering and collaborative filtering generated meaningful recommendations to the user. Additionally, the system leveraged collaborative filtering to suggest skills based on similarities among users, further personalising the recommendations. Despite facing challenges such as data accessibility limitations, the project innovatively utilised generative AI to create dummy datasets, ensuring progress without compromising on quality.

The evaluation of the Skill Boost System revealed its effectiveness in meeting predefined objectives, demonstrating its potential to significantly impact users' learning processes. The report concludes with reflections on project management practices, a discussion of limitations, and promising directions for future work, underscoring the project's success and its foundational role for subsequent enhancements.

Keywords - Recommendation systems, Natural Language Processing, Web Application, Content-Based filtering, Collaborative filtering.

Acknowledgements

Firstly, I would like to thank my supervisor, Claire Rocks, for her continuous support and guidance throughout this project. I would also like to thank my IBM supervisor, Stuart Walker, for taking the time to help and advise me with this project. Lastly, I would like to thank my friends and family. Thank you to my Mum, Dad, Tom and Uncle Paul for their continuous support and encouragement. A particular thanks to my Aunty Neena who has been my biggest hero and role model, continuously guiding and encouraging me in my pursuit in computer science.

Contents

1	Introduction	8
1.1	Motivation	8
1.2	Scope	9
1.2.1	Target Audience	9
1.2.2	Recommendation Subject	10
1.3	Project Goal	11
1.4	Project Outline	11
2	Research	12
2.1	Existing Solutions	12
2.1.1	Defining Existing Systems	12
2.1.2	Analysing Design	13
2.1.3	Functionality	16
2.1.4	Summary	18
2.2	Recommendation Systems	18
2.2.1	Data Pre-processing	18
2.2.2	Content-Based Filtering	21
2.2.3	Collaborative Filtering	24
2.2.4	Hybrid Models	26
2.3	Summary	27
2.3.1	Techniques chosen from Existing Solution research	27
2.3.2	Techniques chosen from recommendation system research	27
3	System Specifications	29
3.1	Project Objectives	29
3.2	Project Requirements	30
3.3	System Requirements and Completion Criteria	30

3.3.1	Develop a Minimum Lovable Product (MLP)	30
3.3.2	Implement a Recommendation System	31
3.3.3	Utilise Natural Language Processing (NLP)	31
3.3.4	Provide Content-Based Filtering Recommendations	31
3.3.5	Incorporate User Feedback into Recommendations	32
3.3.6	Recommend Courses Based on Similar User Interests	32
4	Methodology	33
4.1	Agile Methodology	33
4.2	Plan	34
4.2.1	Phase 1: Implementing a Basic MVP	35
4.2.2	Phase 2: Add Content-Based Filtering	35
4.2.3	Phase 3: Add Collaborative Filtering	35
4.2.4	Summary	36
4.3	Risk Assessment	36
4.3.1	Identifying Risks and Mitigation Strategies	37
4.3.2	Summary	38
4.4	Summary	38
5	Design and Implementation	39
5.1	System Architecture	39
5.1.1	Monolithic Architecture	41
5.1.2	MVC Design Pattern	41
5.1.3	Summary	42
5.2	System Navigation	43
5.2.1	Getting a recommendation	45
5.2.2	Liking a Skill	47
5.2.3	Similar Users	48

5.3	System	50
5.3.1	Frontend	50
5.3.2	Database Design and Implementation	65
5.3.3	Backend	68
5.3.4	Recommendation System	76
5.4	Summary	86
6	Evaluation	87
6.1	Unit and Integration Testing	87
6.1.1	Test Plan	87
6.1.2	Unit Testing	87
6.1.3	Integration Testing	90
6.2	User Acceptance Testing	92
6.2.1	Survey Design	93
6.2.2	Analysis of the Results	96
6.2.3	Summary	102
6.3	Technical Survey	103
6.3.1	Design and Methodology	103
6.3.2	Analysis of Results	106
6.4	Expert Evaluation	112
6.4.1	Overview	112
6.4.2	Methodology	113
6.4.3	Summary of the Interview	113
6.5	Summary	114
7	Project Management	115
7.1	Risk Management Reflection	115
7.2	Legal and Ethical Considerations	116

7.2.1	Bias Mitigation	116
7.2.2	Generative AI Tools	116
7.2.3	User Consent	116
7.2.4	Data Security	117
7.2.5	Accessibility	117
7.3	Limitations	117
7.3.1	Data Accessibility	117
7.3.2	Time Constraints	117
7.3.3	Reliance on Generative AI	118
7.3.4	Summary	118
7.4	Project Reflection	118
7.5	Summary	119
8	Discussion	120
8.1	Future Work	120
8.1.1	NLP Context Enhancement	120
8.1.2	System Integration Applications	120
8.1.3	Expansion to Other Universities or Companies	121
8.1.4	Transcript Generation for CV Enhancement	121
8.1.5	Progress and Achievement Tracking	121
8.2	Conclusion	121
A	Colour Pallete analysis with different colour blindness.	126
B	K-means clustering analysis on User Profiles for Similar User recommendations for varying K values	130
C	Responses for Usability Survey Question 1	135
D	K-means clustering values on Skills for the Technical Survey	136

E Technical Survey Question Format	140
F ChatGPT Data Generation for Software Skills	147
G ChatGPT Data Generation for Event Skills	154

Acronyms

MVC Model-View-Controller. 39, 41

TAM Technology Acceptance Model. 93, 95

UAT User Acceptance Testing. 92, 93, 102

WCAG Web Content Accessibility Guidelines. 50

Chapter 1

Introduction

1.1 Motivation

In the current job market, the alignment of candidates' skills and experiences with employers' needs is not just desirable but essential for securing employment [1]. A significant challenge businesses face globally is the scarcity of candidates possessing the required skills for available positions.

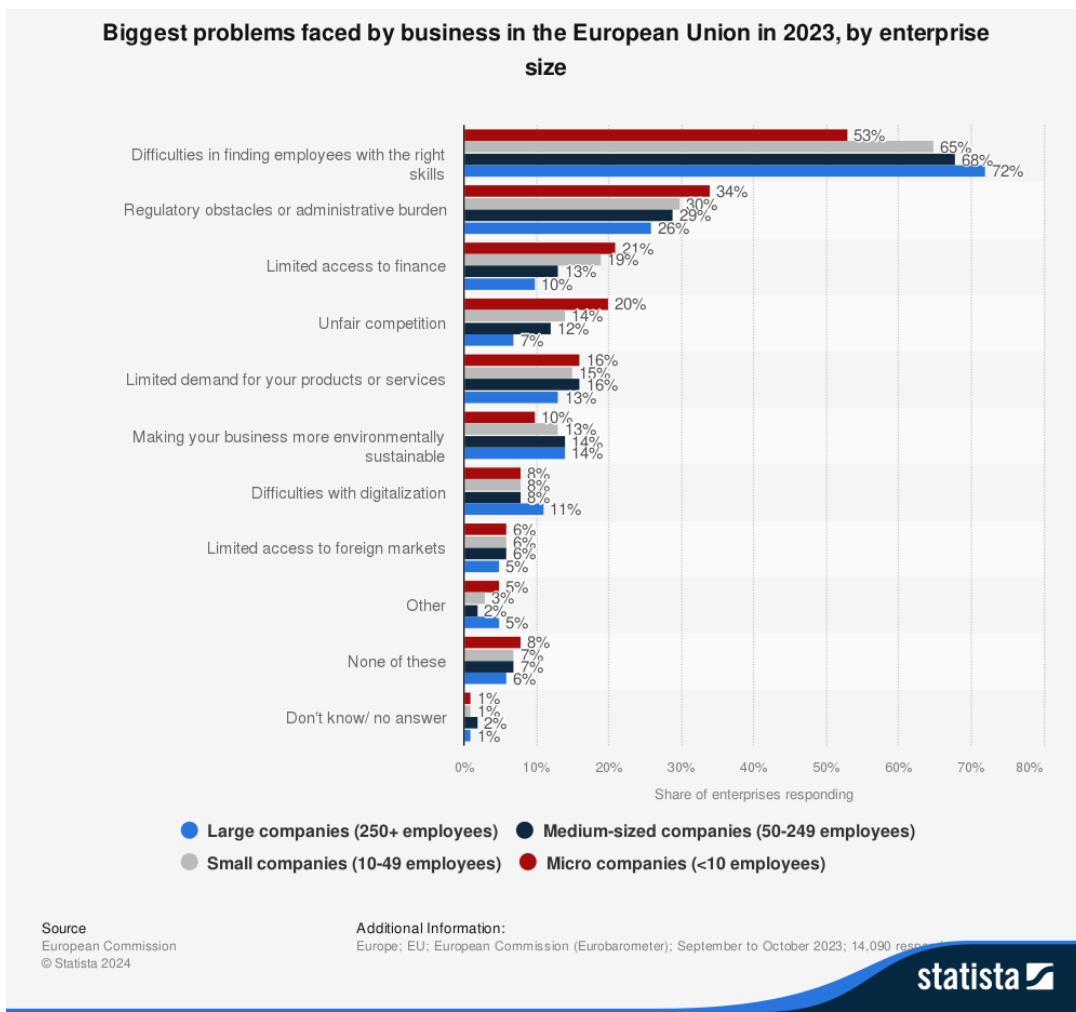


Figure 1.1: Graph illustrating the major challenges faced by businesses in the EU in 2023, highlighting skill shortages as a predominant issue [1].

Figure 1.1 shows the most pressing issue confronting businesses today is the shortage of skilled labour. This not only reduces the growth and competitiveness of companies but also represents a significant

barrier for individuals seeking employment, who often find themselves inadequately equipped with the necessary skills or experience. While there is no shortage of online resources offering free skill enhancement opportunities, navigating these platforms and identifying relevant, quality content can be overwhelming for learners. Among these resources, IBM Skills Build [2] stands out, with users frequently reporting challenges in accessing and utilising the platform effectively [3].

This project is driven by the aspiration to bridge this gap by developing a system that simplifies access to these valuable resources and tailors the learning experience to individual user needs. Beyond the conventional approach of creating a Minimum Viable Product (MVP), this project aims to develop a Minimum Lovable Product (MLP). An MLP delivers a system that resonates with users on an emotional level, fostering genuine affection for the product rather than passive acceptance [4]. By focusing on user engagement and satisfaction from the outset, the project focuses on creating a solution that addresses a critical gap in the job market whilst also enriching the learning journey for individuals seeking to enhance their employability through skill development.

1.2 Scope

To guarantee that the system emerges as a Minimum Lovable Product (MLP), it is crucial to refine the project's scope, especially given the inherent time constraints. Designing a white-label application tailored to a specific use case focuses the efforts whilst ensuring the system's adaptability to various applications. Identifying a precise subject focus and target audience is key to narrowing the scope effectively. This approach means the system will be tailored towards a specific target group as well as improving the capacity of the system to recommend skills within a specific subject domain.

1.2.1 Target Audience

Students appear to be the most appropriate target audience for the system. This demographic is particularly vulnerable to the skills gap as they have the least amount of work experience in their industry. By targeting this group, the project aims to provide a tailored solution that addresses users unique needs for skill enhancement and employability. Warwick University Students will be the most appropriate as there is the opportunity for testing the system on the target audience along with the availability of

department-specific information.

1.2.2 Recommendation Subject

Computer Science is the ideal subject for the system's recommendations. Figure 1.2 shows the pronounced skills gap within the IT industry, demonstrating that Computer Science is an appropriate subject choice for the project.

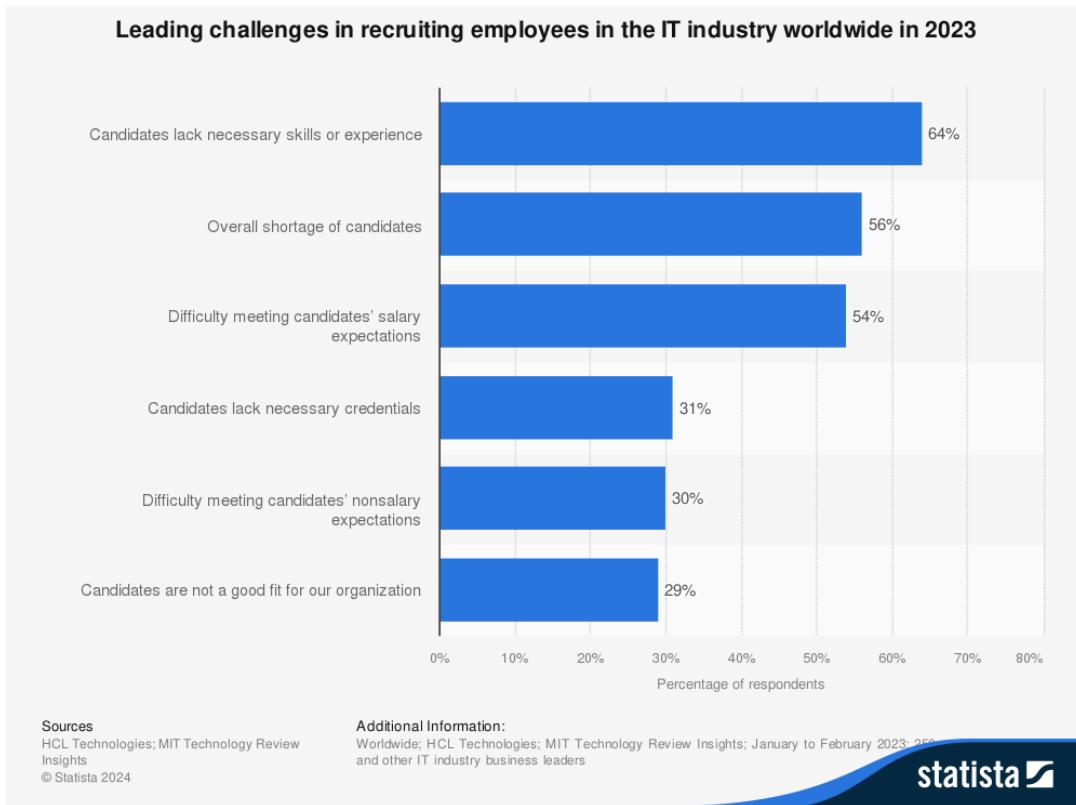


Figure 1.2: Graph illustrating the skills gap within the IT industry, highlighting the critical need for targeted skill development solutions.

This subject focus also capitalises on the developer's expertise in Computer Science as well as ensuring the project has a focus so a system can be created with precision and depth. Since IBM Skills Build specifically offers technology courses, events and software tools, this site will provide the resources that will be recommended by the system. Importantly, while Computer Science is the primary area of focus, the design will ensure the system can be adapted to various subject areas. This flexibility is a cornerstone of the white-label approach, facilitating the creation of an MLP that is both versatile and tailored.

1.3 Project Goal

The overarching goal of this project is to create Skill Boost, a versatile white-label web application that recommends the most relevant IBM Skills Build Computer Science skills (courses, events, and software tools) to University of Warwick students. By clearly defining this goal, the project is positioned to undertake targeted and informed research, which, in conjunction with the underlying motivations, will guide the formulation of specific project objectives.

1.4 Project Outline

This report is structured to provide a comprehensive overview of the Skill Boost project. Chapter 2 examines existing solutions and assesses various recommendation system techniques to identify the most suitable approach. Chapter 3 details the system's objectives and requirements, informed by prior research. The methodology is laid out in Chapter 4, while Chapter 5 delves into the design and implementation of the system, focusing on key functionalities and decision-making rationales. Chapter 6 evaluates the system through testing and user feedback. Project management aspects are discussed in Chapter 7. The report concludes with Chapter 8, summarising limitations, potential future work, personal reflection and overall conclusions on the project's completion and findings.

Chapter 2

Research

This section aims to research existing solutions and techniques, identifying the most appropriate features and techniques to apply to the system. This research will also help to define the objectives and requirements of the project.

2.1 Existing Solutions

The analysis of existing solutions has two aims:

1. Identifying design features targeted towards students.
2. Investigating functionalities provided by online learning platforms.

2.1.1 Defining Existing Systems

Six systems have been chosen to be analysed, three of which are systems designed for students and the remaining three systems provide online resources.

Systems for Students

- UCAS: Offers educational support services, guiding students through university applications [5].
- Unifrog: Assists in comparing university courses, apprenticeships, and further education opportunities [6].
- The Student Room: Provides an online community for UK students to discuss education and lifestyle topics [7].

Systems Offering Online Resources

- Coursera: A vast open online course provider [8].
- Udemy: A platform for learning and teaching online across a wide range of topics [9].

- IBM Skills Build: Offers free digital training in STEM and computer science fields [2].

Evaluating sites like IBM Skills Build is particularly relevant, as it directly provides the types of resources Skill Boost intends to recommend.

2.1.2 Analysing Design

The design of user interfaces significantly influences user engagement and satisfaction. In evaluating the design aspects of existing educational platforms, this project will employ a set of metrics recognised within the field of Human-Computer Interaction (HCI). These metrics offer a structured approach to understanding usability and the overall user experience.

Defining Metrics for Design

Common design aesthetics will be used to evaluate the design of each system. Analysing common design aesthetics will help to evaluate the overall look and feel of each of the systems, aspects of which can be applied to the system being created. The most relevant design aspects being used have been defined and justified below.

- **Colour Scheme:** Assessed for its visual appeal, consistency across the platform, and adherence to aesthetic and minimalist design principles. The colour scheme is essential for accessibility [10], maintaining consistency and making the system look professional.
- **Layout:** Organisation of content and navigational elements, crucial for an efficient and aesthetically pleasing user interface. This approach supports users in focusing on their learning objectives without being overwhelmed by excessive information or complex navigation paths.
- **Tone:** The match between the platform's language and visuals and the real-world context of its users, analysing whether platforms employ a formal, informal, or educational tone and how well this aligns with student expectations. This will help to enhance the user's connection to the content, making learning more relatable and engaging.

- **Affordances:** Indicate how certain elements should be interacted with. This metric ensures that key functionalities are evident, reducing reliance on the user's memory and making navigation more intuitive.
- **Menu:** Assessed on their consistency, recognition ease, and visibility. Effective menu design enables users to effortlessly locate and access different sections, contributing to a frustration-free learning environment.
- **Images:** How they are used to convey information and the system's aesthetics. Images that align with the educational content and user expectations can significantly improve engagement and understanding.
- **Links:** How and where they are used to provide additional information and resources. This is also required as the system will be recommending external resources.

The design analysis will provide nuanced insights into how design choices impact user interactions and satisfaction. By examining these aspects in current systems, the project seeks to extract valuable lessons to guide the design of the system, ensuring it embodies best practices in usability while addressing the unique needs of its intended target audience.

Analysis

Website	Colour Scheme	Layout	Tone	Affordances	Menu	Images	Links
UCAS	Bright, dark, red white and black	Main element is search and articles	Positive, educational	Used for navigation and to highlight specific functionality	Menu but not a search bar to help filter the type of journey each user is on	Lots, used to inform as well as decorative, some moving	Big, bold, blue, takes to specific academic sites when you choose a specific course
The student Room	Simple, blue and white	Easy navigation through forums and resource sections.	Approachable and supportive, community environment	Used to aid navigation	Comprehensive, allows users to organise lots of topics to easily find relevant content	Relatable imagery to make the platform more inviting. Used to be informative as well as decorative	Used clear and descriptive links to guide users to additional resources
Unifrog	Light, simple. White and green	Simple, text and animations	Positive, based at students and parents	No affordances used.	Used as a nav bar to navigate to different pages	Images are animations used only for marketing.	No links are used.
Coursera	Simple, informative. blue and white	Informative, educational. Main element is popular courses.	Formal	No affordances used.	Used to find courses and navigate the site.	Used to demonstrate different courses	No links are used.
Udemy	Plan, simple. purple and white	Main element is popular courses, shown by most popular subjects	Informative and formal	To rate the courses.	Navigate different topics that are popular.	Used to represent each course being recommended.	No links are used.
IBM Skills Build	Basic, simple. blue and white.	Shows lots of different things that the site offers.	Formal, aimed at students and educators	No affordances used.	Organise the type of user that you are.	Shows images of different events, courses and general marketing images.	No links are used.

Table 2.1: Analysis of existing solutions against design metrics.

Design Analysis Summary

Figure 2.1 shows the design analysis for the existing solutions, the analysis revealed some common design trends:

- White and blue colour schemes with an accent colour.
- The site's main functionality is presented in a grid layout.
- Images are used as relatable imagery to promote resources.
- Menus typically appear as a top toggle bar or an expandable side menu, facilitating navigation.

These common trends will be applied to the design of the system's User Interface.

2.1.3 Functionality

The most common website features will be analysed to understand how they are used to obtain information for the user as well as what information the site offers to the user. This analysis will help to understand the different information the users expect the system to require and produce.

Analysis Metrics for Functionality

The following metrics have been identified as the most common features in websites that are believed to be the most appropriate to be used in the system:

- **Search:** Evaluated for the type of information it is collecting from the user, the visibility of the search feature on the site and any if any additional search features, such as filters, are used.
- **User Profiles:** Assessed for the different information available to the user and any additional features available that are not available to an anonymous user.
- **History:** Analysed for the information available and how easy the history is to access.
- **User Feedback:** Evaluating how user feedback is collected and the information that the user can provide feedback on.
- **Search Results:** Assessed for how search results are presented, the information included in the results and how the system presents the relevancy of the results.

The objective of the feature analysis is to identify best practices that can inform the system design, ensuring the system offers an interface that adheres to the needs and preferences of its target users. Through this approach, the project can create a platform that is both intuitive and satisfying for learners.

Analysis

Website	Search	User profiles	History	User Feedback	Search results
UCAS	Easily filter, essential, first thing on page.	Able to see progress, liked courses and your personal stats.	Able to see search history and courses that I have liked and my progress with applying to the courses	Able to like and save courses.	Organised, easily filtered, specifically based on search results or filtered input
The student Room	Robust, essential to find specific threads, advice and content.	View personal information.	Track previous search history and user activity	Able to like posts and comment	Organised and relevant search results,
Unifrog	Search by choosing options from a list to see what courses match your skills and interests.	Can view progress and forms filled in.	No user history functionality available.	No user feedback functionality available	Show percentage match with personality and skills with available courses and a-levels
Coursera	Hidden, used for if you know a specific course or subject you want to learn about.	Can view user history and previous courses.	Easy to see activity	Save courses	Simple, closest match to title that matches to search term
Udemy	Specific courses and topics, not the focus of the website.	Can see history and progress on each courses.	See previous search history and previous reviews and ranking	Rank courses	Most relevant course title to the search input
IBM Skills Build	No search functionality available	Can see courses progress and achievements.	See what courses have started and progress	No user feedback functionality available.	No search functionality available

Table 2.2: Functionality assessment of existing solutions.

Functionality Analysis Summary

Figure 2.2 shows the functionality analysis for the existing solutions, the analysis revealed some common functionality trends:

- Search was the main functionality on the screen, offering users to enter free text into a text box, some included filtering on their search.
- User profile allowed users to track their progress on resources or view saved resources.
- User feedback was provided through saving, liking, or ranking resources.
- Search results commonly gave an ordered list of results, with the most relevant result being in the top left of the options.

These common trends will be included in the functionality available in the system, ensuring that it is designed to meet users' expectations.

2.1.4 Summary

This examination of existing solutions offers crucial insights into design trends and functional features relevant to Skill Boost's target audience. By identifying effective strategies, this analysis provides a solid foundation for designing Skill Boost to meet the specific needs of students seeking computer science skills, ensuring relevance and user expectations are met.

2.2 Recommendation Systems

Recommendation systems are tools that filter information to provide users with personalised recommendations of items. The main aim of recommendation systems is to reduce the amount of time required for users to find relevant information on the internet [11].

Recommendation systems require the data to be represented in a specific manner, once the data has been pre-processed various recommendation steps can be applied. These steps and processes are outlined below:

1. **Data pre-processing:** Represent the data item and user profiles to be used by the recommendation system. [12].
2. Apply appropriate recommendation techniques, there are three different categories of techniques:
 - **Content-based filtering:** Uses item attributes to match users to items to users based on the user's profile [12].
 - **Collaborative filtering:** Uses the rankings of other users to find the most suitable items [12].
 - **Hybrid models:** Uses ideas from both Collaborative filtering and Content-based filtering[11].

The outlined steps and their corresponding techniques will be described, evaluated and compared to identify the most appropriate approach to use for the system.

2.2.1 Data Pre-processing

Data pre-processing requires a representation of the users (user profile) as well as a representation of each data item being recommended (item profile). These profiles need to be consistent across the system for a

fair comparison of attributes.

These profiles can be created from different attributes such as rankings, metadata or descriptions. Feature extraction is a common technique where the descriptions of various items are extracted [12]. The most popular approach to feature extraction is extracting keywords from the underlying data [12]. This is because descriptions are the most natural representations for describing items, it also ensures consistency in representation when comparing items. There are several common feature extraction techniques, such as Natural Language Processing and Deep Learning [12].

Natural Language Processing (NLP)

Natural Language Processing (NLP) extracts meaning and sentiment from text [13], with keyword extraction being the most common NLP technique used in the context of feature extraction [12]. Several different NLP techniques can be used for keyword extraction.

Key Techniques in NLP for Keyword Extraction:

- **Named Entity Recognition (NER):** Extracts key entities from text, enriching the item's feature set for more nuanced recommendations [14]. Topic dictionaries are commonly used which include words and phrases associated with a specific entity [15].
- **Part of Speech tagging (POS):** Adds tags to words in a document to identify the category of different words such as “Noun” or “Verb” [16].
- **Term Frequency - Inverse Document Frequency (TF-IDF):** Highlights the importance of unique terms within item descriptions, this technique is a widely used statistical method that measures how important a term is within a document [17]
- **Bag of Words (BoW):** Provides a baseline representation of item content by counting word occurrences, supporting item comparison based on textual features [17].

Processing Steps in NLP:

- **Stop-word removal:** Removes words that are not specific or relevant to an item, but are common words in the English vocabulary such as “a”, “the” and “of” [12].

- **Tokenisation:** Splitting the text into different words (tokens) that can be analysed individually [16].
- **Phrase Extraction:** Detecting words that occur together in documents. For example, in “Internet of Things”, the constituent words have different meanings to the phrase, as well as containing a stop-word. This step is imperative to ensure that important keywords are not lost, such as in stop-word removal. Dictionaries can be used for phrase extraction, to identify keyword phrases that exist in the text [12].

Once these steps are executed, the keyword profile that represents the text can be converted into a vector-space representation [12], which is a numerical representation of the keyword profile. One common representation includes representing a document as a bag of words (keywords) and their frequencies [12]. However, for short text descriptions, this technique is unsuitable as word frequencies will be low. Therefore, word embedding is another technique that can be used, converting each word into a vector from a vector space with a fixed dimension, where similar vectors are similar words in a similar context [16], therefore sentiment and the meaning of words can be captured in word embeddings.

Deep Learning

Deep learning architectures, inspired by the human brain’s neural network, comprise interconnected nodes or neurons that process and transmit signals. There are several key deep learning architectures, each suited to specific types of data and tasks, including feature extraction in recommendation systems [12]:

- **Multilayer Perceptrons (MLP):** MLPs are foundational feed-forward neural networks characterised by their depth, consisting of three or more layers. They are versatile, and capable of approximating solutions for a wide range of problems, from regression to classification, making them instrumental in identifying intricate patterns within user-item interaction data for recommendation systems [17].
- **Autoencoders (AE’s):** AE’s are unsupervised learning models designed for dimensionality reduction, data compression, and denoising. In recommendation systems, AE’s can effectively capture the latent factors of user preferences and item features, enhancing the system’s ability to recommend personalised content [17].

- **Convolutional Neural Networks (CNN):** CNNs excel in processing visual information, utilising layers of convolution, pooling, and fully connected layers to capture hierarchical features in images. CNNs can extract visual features from items, significantly improving the quality of recommendations in systems where visual appeal is crucial [17].
- **Recurrent Neural Networks (RNN):** RNNs, including Long Short-Term Memory (LSTM) networks, are suited to handling sequential data, making them ideal for recommendations based on time-series data or sequential interaction patterns. They remember past information, enabling the model to predict future interactions or preferences based on historical data [17].

Each of these deep learning architectures brings unique strengths to feature extraction in recommendation systems, from the general applicability of MLPs to the specialised capabilities of CNNs for visual data and RNNs for sequential data. By leveraging these models, recommendation systems can achieve higher accuracy, better understand user preferences, and ultimately, deliver more personalised and relevant recommendations.

Having discussed data pre-processing, an analysis and evaluation of different recommendation techniques can be conducted.

2.2.2 Content-Based Filtering

Content-based filtering leverages the detailed attributes of items, utilising both structured and unstructured data to generate personalised recommendations [17]. This approach compares item profiles to find the most suitable match for a given user profile, classifying whether each item is relevant to a user or not [12]. Content-based filtering employs a variety of sophisticated techniques to achieve this [17]:

Nearest Neighbour Classifier

Nearest Neighbour classifiers determine how similar users or items are to each other. Various measures assess how closely items match a user's preferences or how similar they are to items the user has previously liked [12].

- **Cosine Similarity:** Calculates the angle between two non-zero vectors. This technique is widely

used for comparing item feature vectors, enabling the system to identify items with content closely aligned to the user's preferences [12], with a smaller angle meaning the profiles are most similar. The formula is given by:

$$s(\mathbf{A}, \mathbf{B}) = \frac{\sum_{i=1}^n A_i B_i}{\sqrt{\sum_{i=1}^n A_i^2} \cdot \sqrt{\sum_{i=1}^n B_i^2}}$$

where \mathbf{A} and \mathbf{B} are two vectors.

- **Euclidean Distance:** Measures the direct distance between two feature vectors [12]. This distance is often used when the magnitude of the vectors is important to the analysis. The formula is given by:

$$d(\mathbf{p}, \mathbf{q}) = \sqrt{\sum_{i=1}^n (q_i - p_i)^2}$$

where \mathbf{p} and \mathbf{q} are two points in Euclidean n-space.

- **Manhattan Distance** Calculates the sum of the absolute differences of coordinates. Manhattan distance is particularly useful in high-dimensional spaces where the Euclidean distance can become inflated due to the curse of dimensionality. The formula is:

$$d(\mathbf{p}, \mathbf{q}) = \sum_{i=1}^n |q_i - p_i|$$

Each of these metrics offers a different perspective on similarity, with the choice of metric depending on the specific requirements of the recommendation system [12]. For example, while cosine similarity is adept at identifying directional similarity regardless of magnitude, Euclidean and Manhattan distances provide a sense of spatial or 'physical' proximity between points in the feature space.

Bayes Classifier

A Bayes Classifier is a straightforward probabilistic classifier which operates on the principle of applying Bayes' theorem with a strong (naive) assumption of independence between features. This classifier is termed "naive" because it assumes that the presence (or absence) of a specific characteristic of a class is independent of the presence (or absence) of any other characteristic, given the class variable. Despite this simplification, naive Bayes classifiers often perform significantly better in scenarios where the independence

assumption is roughly true.

The core mechanism of the Bayes classifier can be summarised as follows:

- **Bayes' Theorem:** The Bayes classifier is underpinned by Bayes' theorem, which describes the probability of an event, based on prior knowledge of conditions that might be related to the event [12]. Mathematically, it is expressed as:

$$P(A|B) = \frac{P(B|A) \cdot P(A)}{P(B)}$$

where $P(A|B)$ is the probability of hypothesis A given the data B , $P(B|A)$ is the probability of data B given that hypothesis A is true, $P(A)$ is the prior probability of A , and $P(B)$ is the probability of data B .

- **Classification Problem Setup:** Given a set of features (or attributes) of an object, and the goal is to assign it to one of several classes based on these features. For example, in email spam detection, features might include the presence of certain words, with the classes being "spam" and "not spam."
- **Training Phase:** The model learns the probability of each feature being associated with each class. The assumption of independence between features significantly simplifies the computation.
- **Prediction Phase:** The classifier utilises the feature probabilities learned during training to calculate the probability of the instance belonging to each class. The instance is then classified into the class with the highest probability.

Despite its simplicity and naive assumption, the Bayes classifier has shown remarkable success in many real-world applications, especially in text classification tasks such as spam detection and document categorisation [18].

Rule-based Classifier

Rule-based classifiers apply predefined logic and rules to classify items or recommend them based on their match with a user's profile or preferences. One prevalent form of rule-based classifiers in recommendation

systems is associative classifiers, these classifiers are designed to leverage the relationships between items (or between users and items) to make predictions or recommendations.

Associative classifiers define rules where the antecedents (conditions) and consequents (outcomes) are based on the attributes of items or ratings by users [12]. A rule's support indicates the proportion of instances satisfying both the antecedent and the consequent, while its confidence measures the likelihood of the consequent being true when the antecedent is satisfied.

A significant advantage of rule-based systems is their interpretability. For any recommended item, the system can provide insights into why it was recommended by referring to the keywords in the antecedents of the rules that were triggered. This feature can enhance user trust and engagement by making the recommendation process transparent.

Summary

Content-based recommendation systems offer significant advantages, such as mitigating the data sparsity problem and the ability to recommend new items, solving the new item cold-start issue. They also provide transparent explanations for recommendations, enhancing user trust[17].

However, there are challenges in generating accurate recommendations for new users due to the new user problem, a tendency towards overspecialisation in recommendations, and varying effectiveness across different content types. While particularly well-suited for text-based items, content-based systems may require more complex feature extraction techniques for items like images or music [17].

2.2.3 Collaborative Filtering

Collaborative filtering (CF) is a technique in recommendation systems, distinguished by its ability to recommend items based on the preferences of similar users. It primarily operates by predicting user preferences for items the users have not yet interacted with, by identifying patterns and similarities within the data.

Types of collaborative filtering

There are two main types of collaborative filtering, User-based and Item -based [11]. The techniques applied to the different types of collaborative filtering techniques are the same but the items being compared are different:

- **User Based Collaborative Filtering:** Focuses on finding users similar to the target user, recommending items those similar users have liked or interacted with.
- **Item Based Collaborative Filtering:** Identifies items similar to those already liked or rated highly by the target user, based on the preferences of other users.

Collaborative Filtering Techniques

Matrix Techniques: Matrix factorisation techniques decompose the user-item interaction matrix into lower-dimensional matrices, revealing latent factors associated with users and items, thereby improving the prediction accuracy of collaborative filtering. Key techniques include:

- SVD (Singular Value Decomposition): A dimensionality reduction technique that factorises the user-item matrix into three matrices, highlighting the latent features of users and items [18].
- ALS (Alternating Least Squares): Aims to find the best factorization by minimizing the squared differences between the observed ratings and the product of two lower-dimensional matrices, iteratively optimizing one while keeping the other fixed [18].
- PCA (Principle Component Analysis): Another dimensionality reduction technique that helps in understanding the underlying structure of the data, identifying patterns, and reducing dimensions while preserving as much variance as possible.

Association Techniques: This approach involves calculating the similarity between two users or two items, akin to content-based filtering. However, the focus here is on the relationships between users or items rather than between an item and a user's profile.

Neighbourhood Models: Neighbourhood models recommend items by identifying a subset of users or items that are similar to the target user or item. This can be achieved through various means:

- **K-means Clustering:** Segments users or items into clusters based on their preferences or characteristics, facilitating recommendations from the same cluster [18].
- **Fixed Number of Neighbors (NN):** Recommends based on a predefined number of closest users (user-based) or most similar items (item-based).
- **Similarity Threshold:** Utilises a minimum similarity threshold to select the neighbours for making recommendations [18].

Collaborative filtering (CF) excels in domains where item content is not easily quantifiable, such as movies and music, allowing for discoveries and recommendations based on user behaviour rather than item attributes. It's particularly good at identifying and leveraging complex patterns of user preferences [18]. However, CF techniques face challenges, including the cold-start problem for new users or items, data sparsity in large matrices, scalability with growing data, and synonymy, where similar items are perceived as distinct due to different naming conventions.

2.2.4 Hybrid Models

Hybrid Filtering Techniques

Hybrid filtering techniques integrate various recommendation strategies to enhance system performance and mitigate the limitations of singular recommendation approaches [18]. By amalgamating different algorithms, these techniques aim to provide more accurate and effective recommendations, leveraging the strengths of one approach to counterbalance the weaknesses of another. There are several methods of hybridization, each with its distinct application and benefits:

Weighted Hybridization: Combines results from multiple recommenders, weighting each by relevance or accuracy to produce a final recommendation [18].

Switching Hybridisation: Dynamically select the recommendation technique best suited for the current context, removing the limitations of any single method [18].

Cascade Hybridisation: Iteratively refines recommendations, starting with a broad list that is progressively narrowed down by subsequent techniques [18].

Feature-Augmentation: Integrates additional data from one system into another, often adding minimal but highly informative features for the recommendation system [18].

Summary: Hybrid filtering represents a sophisticated evolution in recommendation systems, addressing common challenges such as the cold-start problem, data sparsity, and scalability. By integrating multiple recommendation techniques, hybrid systems offer a more versatile, robust, and user-centric approach to filtering, effectively capturing the complexity of user preferences and item attributes [18].

2.3 Summary

The research conducted provides a comprehensive foundation for determining the most suitable methods for developing a web application aimed at recommending learning resources. This evaluation discusses the rationale behind selecting specific methods for feature extraction, recommendation techniques, and the hybrid approach, underscoring their appropriateness for the Skill Build platform.

2.3.1 Techniques chosen from Existing Solution research

The existing solutions research demonstrated that the most appropriate system is a web application, as it can easily be used and accessed. User input through search also appeared the most common technique, therefore to obtain a recommendation users will search for a topic they wish to learn more about. The ability to provide feedback to resources also appeared important, therefore the user will be able to like a recommendation.

2.3.2 Techniques chosen from recommendation system research

Following from the recommendation research, the following techniques for each recommendation stage have been chosen to be the most suitable for the system.

Techniques chosen for Data Pre-Processing:

Given the application's reliance on text user input, NLP is the most suitable technique to employ. NLP allows the system to interpret and transform unstructured text into a format that the recommendation system can understand and analyse. This approach is vital for accurately capturing the user's learning interests from their textual descriptions.

Dictionary-based Named Entity Recognition (NER) will be the NLP technique implemented as it is suitable for the anticipated short user input statements. This ensures that even brief user descriptions can yield meaningful insights, guiding the recommendation process effectively.

Techniques chosen for Recommendations:

Given its ability to leverage the strengths of different techniques, a hybrid model has emerged as the most appropriate approach. This means that the system will be able to create personalised recommendations based on textual content whilst also incorporating user feedback to refine these suggestions over time.

- **Content-Based Filtering Using Cosine Similarity:** Cosine similarity emerges as the appropriate content-based filtering technique, ensuring recommendations are directly aligned with the user's expressed interests.
- **Collaborative Filtering with User Feedback and K-Means Clustering:** User feedback will be used to adjust recommendation scores. Furthermore, k-means clustering will categorise users into distinct groups based on their preferences, enabling the system to offer recommendations that have resonated with similar users.

Overall, this research provided valuable insights into the appropriate techniques that are to be used in the system. The chosen techniques provide a foundation for helping to define the core aims and objectives of the system.

Chapter 3

System Specifications

Defining the project's motivation, and goal and conducting relevant research has outlined the objectives for the system. These objectives can be broken down into requirements that can be measured. This section outlines the different objectives and system requirements that define the project, which will guide any decisions made during the project.

3.1 Project Objectives

Objectives are marked as **Core** or **Extended**; core objectives are prioritised and highlight the system's main focus. The systems **CORE** Objectives:

1. **Develop a Minimum Lovable Product (MLP):** Create a web application that is functional and appealing to students through its functionality and aesthetic appeal.
2. **Implement a Recommendation System:** Design and develop a system capable of recommending relevant computer science skills (courses, events, and software) from IBM Skills Build specifically tailored for University of Warwick students.
3. **Utilise Natural Language Processing (NLP):** Employ NLP techniques to analyse and understand user text input, ensuring that recommendations are based on a nuanced understanding of user queries.
4. **Provide Content-Based Filtering Recommendations:** Use content-based filtering to offer personalised recommendations by matching the features of the items (skills) with the user's text input preferences.

The systems **EXTENDED** Objectives:

- **Incorporate User Feedback into Recommendations:** Design the system to learn from user feedback, improving the accuracy and relevance of recommendations over time.

- **Recommend Courses Based on Similar User Interests:** Implement collaborative filtering techniques to suggest courses and resources liked by similar users, enhancing personalisation.

These objectives collectively form a strategic blueprint that guides all subsequent decisions and developments, ensuring the final system is robust, user-friendly, and effectively meets the educational needs of its target audience.

3.2 Project Requirements

The project objectives outline broad goals, each broken into specific requirements detailing necessary criteria and functionalities for achievement [19]. Functional requirements (FR) describe what the system does, while non-functional requirements (NFR) focus on system performance [19]. Measurable and verifiable completion criteria for each requirement are established and preferred over Specific, Measurable, Achievable, Relevant, and Time-bound (SMART) objectives. This approach avoids fixed timeframes, emphasising functional completion and quality, essential for the project's success. Requirements relating to each objective are defined below.

3.3 System Requirements and Completion Criteria

3.3.1 Develop a Minimum Lovable Product (MLP)

- **FR1: User Account Creation and Management**

- The user can register for an account.
- The user can log into their existing account.

- **FR2: Recommendation Interface Web App**

- The user can enter a description of a topic they want to learn in a free text box.
- The user can filter specific types of skills using a filter button.
- The system displays the type of skill with the recommendation.

- **NFR1: The application must be user-friendly, with intuitive navigation and accessible design**

– New users can quickly understand how to use the application, supported by help resources.

3.3.2 Implement a Recommendation System

- **FR1: Personalised Recommendations**

– The system analyses user input and preferences to generate personalised recommendations.

- **FR2: User Feedback on Recommendations**

– The user can provide feedback on the recommendations received.

- **NFR1: The system generates recommendations quickly.**

– Recommendations are generated and displayed within 5 seconds of receiving user input.

3.3.3 Utilise Natural Language Processing (NLP)

- **FR1: Process Natural Language Queries**

– The system processes natural language queries to extract keywords and phrases for analysis.

- **NFR1: The NLP engine processes inputs efficiently.**

– The NLP engine processes user inputs efficiently, maintaining a low response time.

3.3.4 Provide Content-Based Filtering Recommendations

- **FR1: Match User Profiles with Item Attributes**

– The system matches user profiles with item attributes based on cosine similarity measures to generate recommendations.

- **FR2: Update Recommendations Based on Interactions**

– Recommendations are scored based on user feedback obtained from previously suggested items.

- **NFR1: Each user should receive a unique set of recommendations tailored to their input.**
 - Each user receives a unique set of recommendations tailored to their user profile.

3.3.5 Incorporate User Feedback into Recommendations

- **FR1: Allow Users to Rate or Give Feedback**
 - The user can 'like' or 'unlike' a recommendation.
- **FR2: Adjust Recommendations Based on Feedback**
 - The system adjusts future recommendations based on user feedback patterns by updating the score of the recommendation.

3.3.6 Recommend Courses Based on Similar User Interests

- **FR1: Identify Similarities Among User Profiles**
 - The system uses k-means clustering on user profiles to identify similarities among users.
- **FR2: Suggest Items Liked by Similar Users**
 - Items liked by users in the same cluster are suggested to the user.
- **NFR2: Dynamic Learning**
 - The system periodically re-calculates the k-means clustering to update the clusters.

Chapter 4

Methodology

This chapter defines the methodology being used for the project as well as outlining a plan that aligns with the methodology's concepts. Assessment of any potential risks will also be evaluated that, highlighting any mitigation's that can be implemented to ensure seamless project delivery.

4.1 Agile Methodology

This project will utilise an Agile approach, featuring iterative and collaborative two-week sprints to enhance the recommendation system. It begins with thorough planning, design, and research, establishing a foundational project skeleton. The initial sprint deploys a library-based system to collect early user feedback. Subsequent sprints focus on developing a custom system with refined algorithms, enhanced functionality, and personalised recommendations. Each sprint includes iterative testing and evaluation to validate improvements and address issues.

Opting for the Agile methodology in this project, particularly over Waterfall and Lean approaches, aligns with the objective of developing a Minimum Lovable Product (MLP). While the sequential nature of the Waterfall model might limit rapid iteration due to its rigid structure, and Lean intensely focuses on reducing waste, potentially at the cost of iterative creativity, Agile offers a balanced and flexible framework essential for the iterative development and incorporation of user feedback critical to MLP creation [19].

Agile's iterative development process allows for constructing a foundational system that can be progressively enhanced with features that users adore, rather than just fulfilling the bare minimum functionality. This strategy is crucial for an MLP, as it ensures the product evolves in direct response to actual user needs and preferences, fostering a user-centric design philosophy from the beginning [4].

The Agile methodology, with its emphasis on regular sprints and feedback loops, facilitates continuous evaluation and refinement of the product [19]. Agile's adaptability enables the project to prioritise the development of features that significantly boost user satisfaction and engagement, which are key components of an MLP.

4.2 Plan

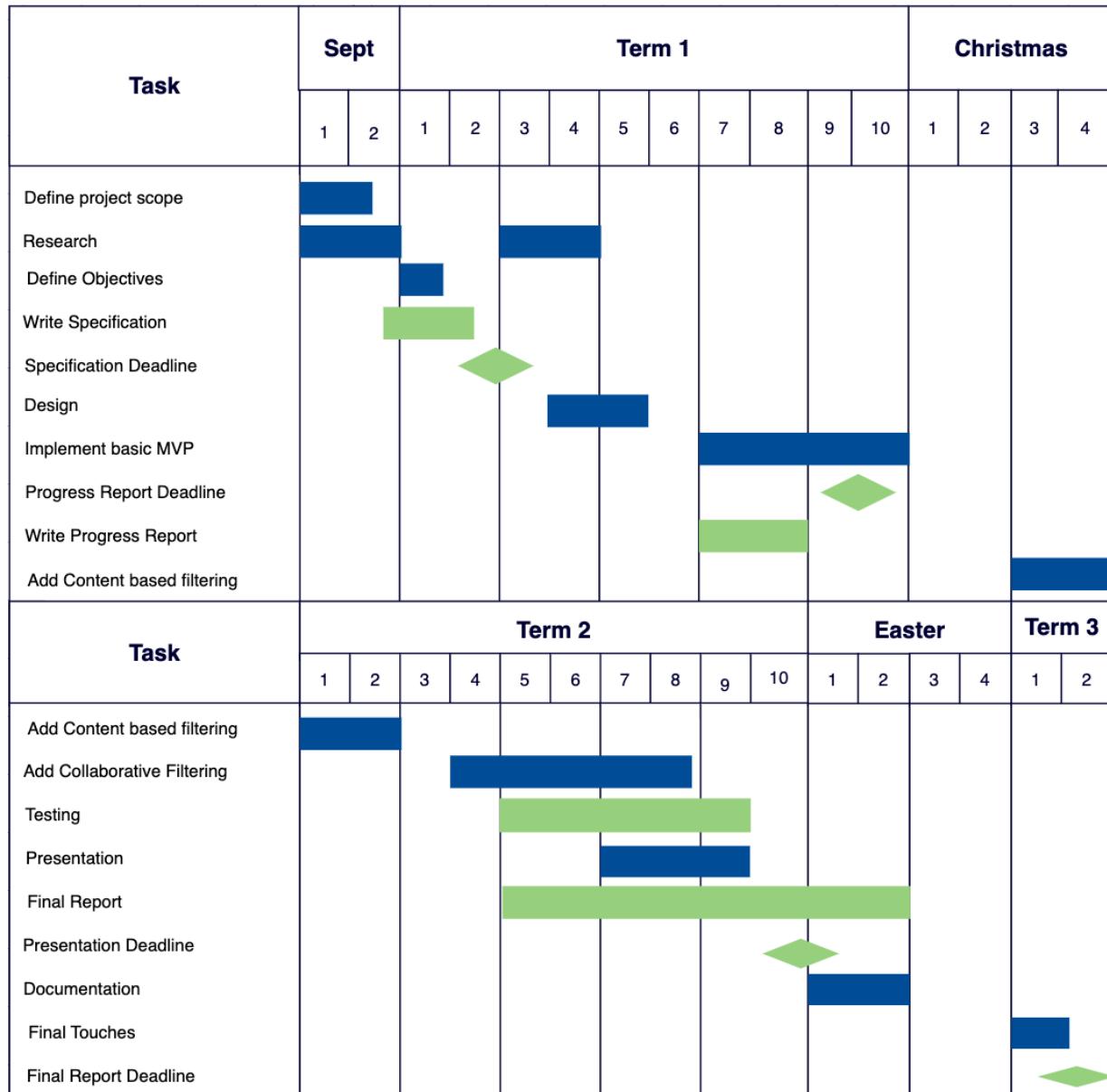


Figure 4.1: Gantt chart illustrating the project timeline, highlighting the key development phases and documentation tasks.

Figure 4.1 shows the Gantt chart for the project, tasks are colour-coded for clarity, with green indicating documentation tasks and blue denoting project development tasks. This visual timeline facilitates a clear understanding of the project's progression and how each task aligns with the Agile methodology's iterative approach.

4.2.1 Phase 1: Implementing a Basic MVP

The initial phase focuses on establishing a Minimum Viable Product (MVP) that meets Objectives 2 (Implement a Recommendation System) and 3 (Utilise Natural Language Processing). This phase involves:

- Creating the foundational recommendation system that performs basic matching of user input keywords to relevant courses, events, and software.
- Implementing Natural Language Processing (NLP) capabilities to produce keywords from the user input.

This foundation lays the groundwork for subsequent phases, ensuring the system's core functionality is in place and operable.

4.2.2 Phase 2: Add Content-Based Filtering

The second phase expands the system's capabilities by introducing content-based filtering, aligning with Objectives 1 (Develop an MLP), 2, and 4 (Provide Content-Based Filtering Recommendations). Key developments include:

- Integrating cosine similarity algorithms to refine the recommendation engine, enabling more nuanced matching based on content attributes rather than frequency of keyword matches.
- Develop user functionality, allowing for a more personalised interaction with the system, including user preference settings and saved searches.

This phase aims to elevate the user experience by making the recommendations more tailored and relevant to individual users.

4.2.3 Phase 3: Add Collaborative Filtering

The final phase introduces collaborative filtering, further aligning with Objectives 1, 5 (Incorporate User Feedback into Recommendations), and 6 (Recommend Courses Based on Similar User Interests). Developments in this phase include:

- Implementing mechanisms for collecting and analyzing user feedback to continuously refine recommendation accuracy.
- Utilising k-means clustering on user profiles to identify and recommend resources liked by similar users.

This phase emphasises leveraging user insights and preferences to enrich the recommendation process.

4.2.4 Summary

Each phase is deliberately planned to ensure that the development not only builds upon the achievements of the previous phases but also moves the project closer to meeting all stated objectives. By the end of Phase 3, the project aims to have fully realised a comprehensive, user-centric recommendation system.

4.3 Risk Assessment

Recognising potential risks and developing effective mitigation strategies is crucial. Proactively identifying risks ensures strategies can be implemented that minimise project delays, cost overruns, and quality issues. Below is a detailed assessment of identified risks and their corresponding mitigation strategies, emphasising the importance and rationale behind each approach.

4.3.1 Identifying Risks and Mitigation Strategies

Risk	Likelihood	Mitigation Strategy
Product is unfinished	High	Implement an iterative development approach, ensuring each iteration delivers a functional product increment.
The newest recommendation system iteration fails.	High	Implement version control for rollback on failure; conduct thorough testing before deployment.
Software-related device crashes	Medium	Implement comprehensive testing across devices; maintain code quality and updates through version control.
The backend infrastructure is unsuitable for future scalability.	Medium	Design initial backend with scalability in mind, incorporating modular and extensible architecture.
Data Access Denied	Low	Prepare backup strategies like using dummy datasets that accurately reflect real data characteristics.

Table 4.1: Assessment of potential project risks with corresponding mitigation strategies.

Product is Unfinished: This is a high risk due to the project's ambitious scope and tight timelines. The iterative development approach is selected as a mitigation strategy, allowing for continuous delivery of workable product increments. This approach ensures that there is always a working version of the product and enables adjustments to the project scope based on time constraints, ensuring the most critical features are developed first.

Newest Recommendation System Iteration Fails: This is a high risk due to the inherent complexity of developing advanced recommendation algorithms. Using version control systems and conducting thorough testing before each deployment can mitigate this risk. Version control allows rollbacks to previously stable versions if new updates fail, ensuring the system's reliability. Thorough testing, including unit tests and user acceptance testing, helps catch and fix issues before they affect the end users.

Software-related Device Crashes: This risk is medium as there could be crashes of devices. Version control ensures that there is an external, backed-up version of the system that is up to date. Regularly updating the backup is essential to save essential changes that have been made.

Backend Infrastructure Unsuitable for Future Scalability: As user numbers grow, there's a medium risk that the initial backend setup may not handle the increased load. Designing the backend with scalability in mind from the outset addresses this concern. Using a modular and extensible architecture allows for easier updates and scaling, ensuring the system can grow to meet user demand.

Data Access Denied: This risk is considered low but could significantly impact the project if external data sources become unavailable. Preparing backup strategies, such as developing dummy datasets that closely mimic real data, ensures that development and testing can proceed without delays, even if access to external data is restricted.

4.3.2 Summary

Each identified risk is closely tied to the project's success. The chosen mitigation strategies reflect a proactive approach to project management, emphasising the importance of flexibility, quality assurance, and preparedness. Careful risk planning ensures that the system is developed on time, within scope, and meets high-quality standards.

4.4 Summary

Overall this chapter has provided the foundational plan and methodology that the project is going to follow to ensure that all of the objectives and requirements are met within the given time frame. Defining the plan and methodology helps to inform design and development as there is a clear idea when each aspect of the system needs to be done.

Chapter 5

Design and Implementation

This section reviews the system's design and implementation, aiming to provide a clear understanding of how each subsystem was deliberately designed, constructed, and integrated to meet the project's objectives. The discussion includes exploring the system's architecture and layout, supplemented by UML diagrams, to illustrate how different parts effectively work together. The design approach for each subsystem is based on thorough research, ensuring that decisions are not only strategic but also appropriate for the project's needs. The primary goal is to offer a detailed insight into the system's functionality, connecting design choices to the project's objectives and initial research to underscore the reasoning behind these methodical decisions. This illustrates how the system's structure and capabilities collectively satisfy the specified requirements.

5.1 System Architecture

The system has been designed using a monolithic architecture and Model-View-Controller (MVC) design pattern.

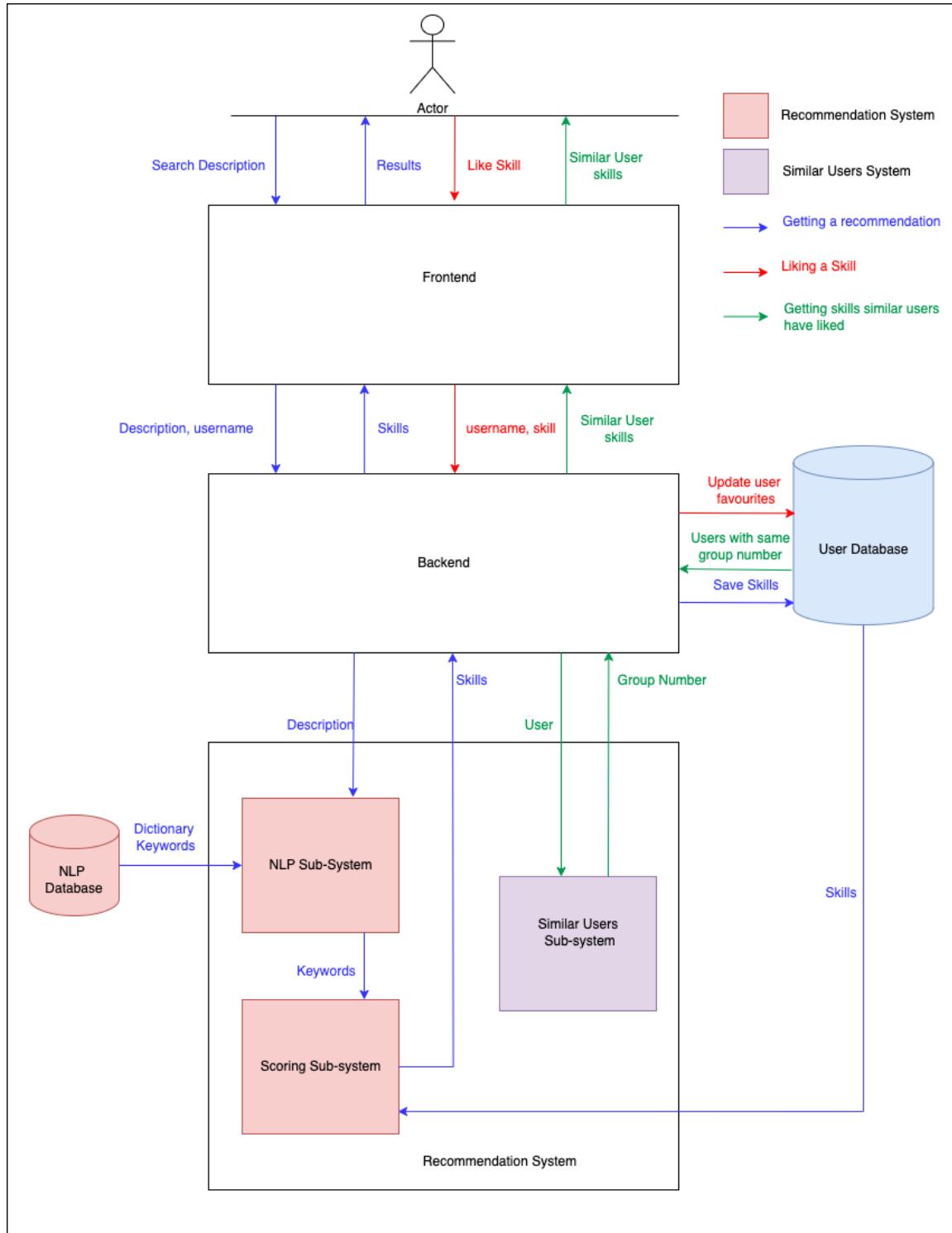


Figure 5.1: Skill Boost System Diagram.

5.1.1 Monolithic Architecture

A monolithic architecture is suitable for scenarios where development simplicity and operational efficiency are prioritised [20]. This approach, characterised by a singular codebase, facilitates a straightforward development, testing, and deployment regimen, particularly advantageous for small to medium-scale projects. It simplifies debugging and operational tasks by bypassing the complexities inherent in distributed systems, such as network latency and inter-service communication challenges.

Monolithic architectures allow for swift iterations and adjustments. They are particularly suitable for projects where the scope and required scalability align with the architecture's capabilities, ensuring the system remains robust and manageable within its intended operational parameters [20].

Thus, for this project, where the complexity and scale align with the benefits offered by a unified codebase, monolithic architecture emerges as a compelling choice. It ensures the application remains accessible, maintainable, and efficiently operable, aligning with best practices in software architecture for applications with well-defined boundaries and scalability expectations.

5.1.2 MVC Design Pattern

Figure 5.1 shows the system design, following a Model-View-Controller (MVC) design pattern. This design pattern serves as the blueprint for developing a web application that is robust, interactive and adaptable to evolving user requirements and technology advancements.

MVC Components

- **Frontend (View):** Where the user interactions occur, communicating user actions to the backend and displaying the results returned from the backend.
- **Backend (Controller):** Acts upon the user inputs, processes them and then dictates how the data is passed between the frontend and the recommendation system. The backend also interacts with the User database, to update, retrieve and store information and user actions.
- **Recommendation System (Model):** Along with the User database, the recommendation system handles the application's data logic. The system responds to requests from the backend to either

produce a recommendation or assign a user a cluster by interacting with the database and sending back the computed results.

- **NLP Sub-System (Part of Model):** The NLP database and subsystem contribute towards the business logic and data normalisation by converting the user input into a keyword representation before it is used by the recommendation system.
- **Similar Users Sub-system (Part of Model):** This subsystem processes user data and history to assign a user to a user group, which is used by the Model to enhance the recommendation logic.

Rationale for MVC Architecture

MVC was chosen for its modularity and compatibility with Agile methods, ideal for complex NLP and recommendation processing by isolating logic from the UI and control flow, and facilitating scalable backend management. Despite potential complexities and redundancy in MVC, its structure remains robust for the system, with strategies like code review and refactoring [19].

Comparatively, MVC outperforms Layered and Repository Architectures for this project. The repository architecture, with its centralised data handling, limits UI flexibility, and the strict layering of the Layered architecture could introduce latency not suitable for dynamic interactions required by the recommendation system [19].

5.1.3 Summary

Incorporating a monolithic architecture into the web application, alongside the MVC design pattern, establishes a foundation for managing structured data, complex business logic, and responsive user interfaces. This approach ensures high interactivity, scalability, and modularity, essential for the system's demands. The monolithic framework, complemented by the MVC architecture, enables the model, view, and controller components to be developed, tested, and scaled within a unified codebase. This integration facilitates a streamlined Agile development process, catering specifically to the project's requirements. This offers a pragmatic solution, leveraging the strengths of MVC for component independence and monolithic architecture for simplicity and cohesion, ensuring the system's effectiveness in delivering a seamless user

experience.

5.2 System Navigation

This section focuses on the users interaction with the system, showing how the functionalities outlined in the previous section are performed by the user, specifying how the different systems interact to create the desired functionality.

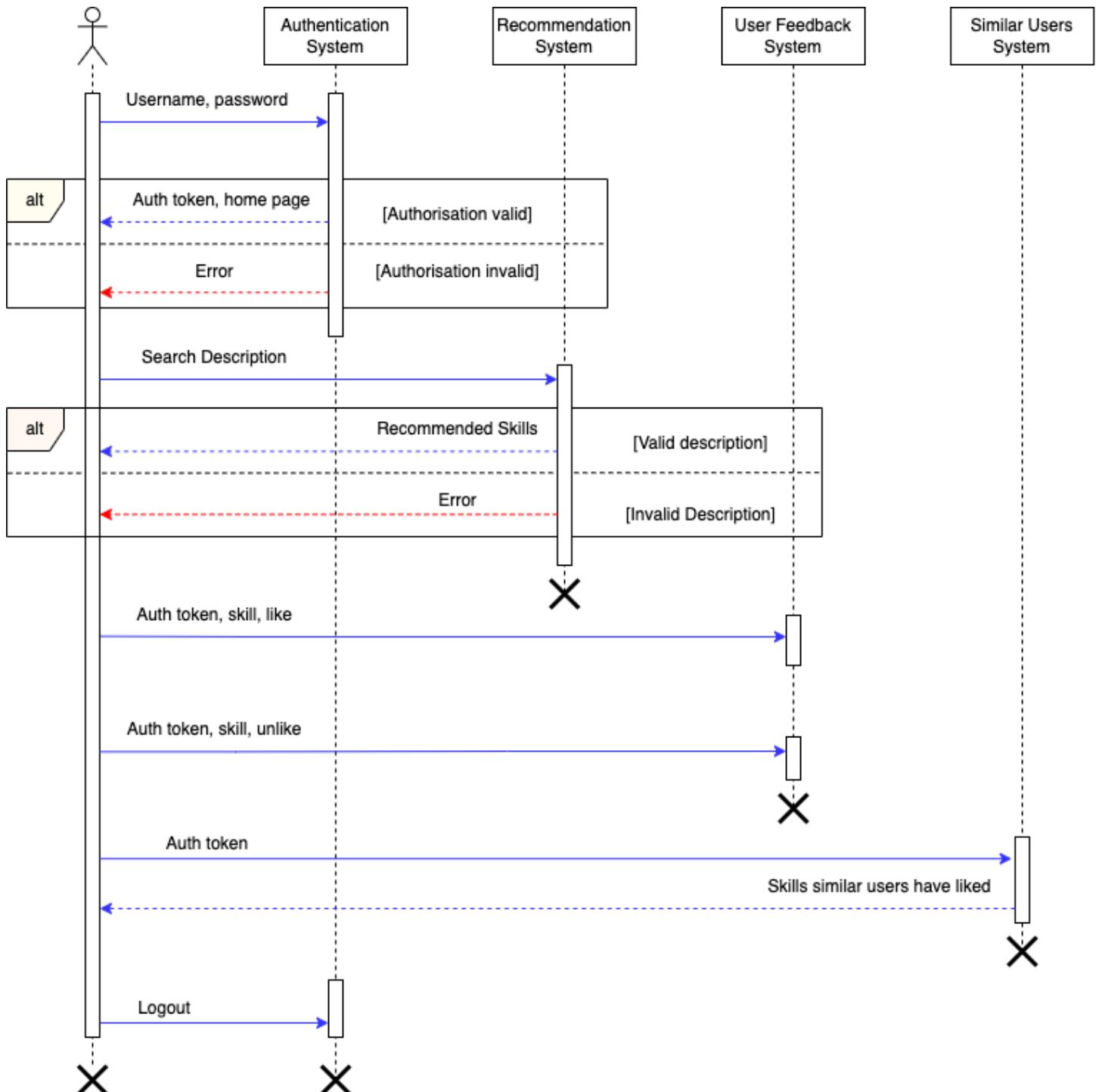


Figure 5.2: UML Sequence diagram showing the main functionalities in the system and how the different subsystems interact.

Figure 5.2 shows the sequence diagram, illustrating the dynamic interactions between different subsystems and components in response to user actions. By providing a clear visualisation of system interactions,

the sequence diagram aids in the design, implementation, and testing phases, ensuring that the final system not only meets the initial design specifications but also adapts to the requirements. The most important functionalities outlined in the sequence diagram are described below, breaking down the fundamental tasks required by each component.

5.2.1 Getting a recommendation

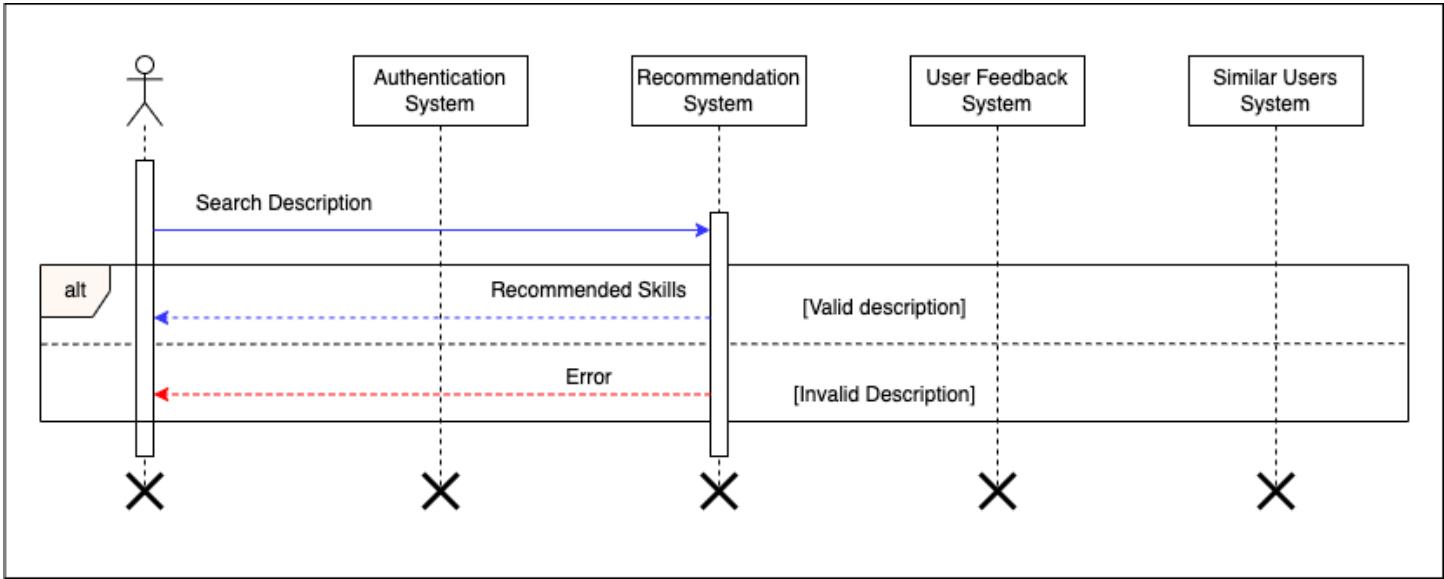


Figure 5.3: Subsection of the UML Sequence diagram in figure 5.2 showing the user requesting and receiving a recommendation.

Figure 5.3 is a section of Figure 5.2, illustrating the process of a user requesting and receiving a recommendation. This functionality is designed to be accessible without requiring users to log in, demonstrating the system's capabilities to new users and encouraging trial use. Registered users, however, benefit from additional functionality, such as being able to like skills that they have been recommended. The roles of each system component in facilitating this feature and their contributions are outlined below.

- **Frontend:** Engages the user by requesting a description of the topic they wish to learn about. Users can further refine their search with filters for the types of skills they are interested in. Invalid inputs trigger an error message clarifying the issue, whereas valid inputs lead to a results page providing relevant skills. Each listed skill includes its title, type, description, and a link to the resource.

- **Backend:** Manages the recommendation request. For anonymous users, requests are forwarded directly to the recommendation system without logging. For logged-in users, the system logs the request, updates the user's profile history, and, after processing by the recommendation system, saves the received recommendations. It also increments a counter for each recommended skill in the top 10, indicating how often it has been highly recommended.
- **Recommendation System:** Utilises the NLP Sub-system to convert the user's text input into a keyword profile, which is then matched against the keyword profiles of skills in the database to identify the most relevant skills. A more detailed explanation of the recommendation system's workings is provided in Section 5.3.4.

This process underlines the system's focus on accessibility for new users and enhanced personalisation for registered users, creating a seamless and user-friendly experience.

5.2.2 Liking a Skill

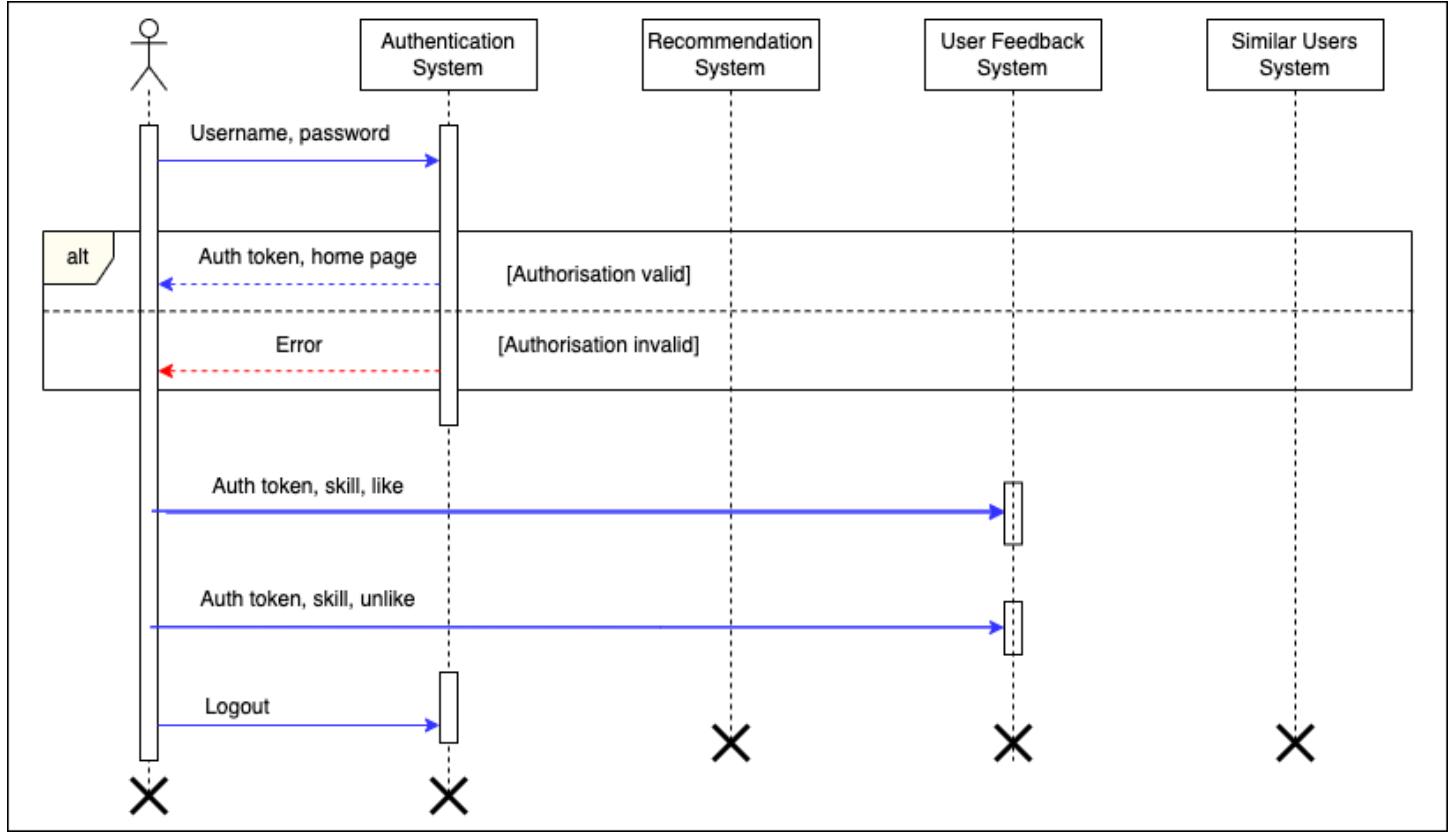


Figure 5.4: Subsection of the UML Sequence diagram in figure 5.2 showing the user liking a skill.

Figure 5.4 is a section of 5.2, illustrating the process of a user liking a skill. This functionality enhances user engagement by allowing logged-in users to like or unlike skills recommended to them. This feature introduces different profile levels within the system, reserving certain functionalities for registered users and allowing them the flexibility to modify their preferences. User feedback also shows user preference, which can be used to help inform the recommendation system, this is discussed in section 5.3.4.

- **Frontend:** When users provide feedback on a recommended skill, the system dynamically reflects this action by changing the icon's appearance, providing a clear, immediate visual cue of the action's processing. This feature, coupled with the ability to review liked skills on the User Dashboard, significantly enriches the user experience.
- **Backend:** The backend initiates the like/unlike process upon user interaction. This action is cap-

tured by the Controller, which then communicates with the Service layer to update the user's preferences and the skill's metadata accordingly. If a skill is unliked, the Service layer recalculates the user's profile based on the revised list of liked skills, with the Repository layer ensuring these changes are accurately reflected in the database. This efficient division of labor across the MVC architecture components ensures the recommendation system remains optimally focused on its primary function of generating personalised suggestions.

This feature represents a crucial aspect of the system's user-focused design, providing an interactive and personalised experience to the user. This functionality enhances user engagement and ensures the system's architecture remains aligned with best practices, delivering a responsive and intuitive platform, where user actions directly influence the refinement of their experience.

5.2.3 Similar Users

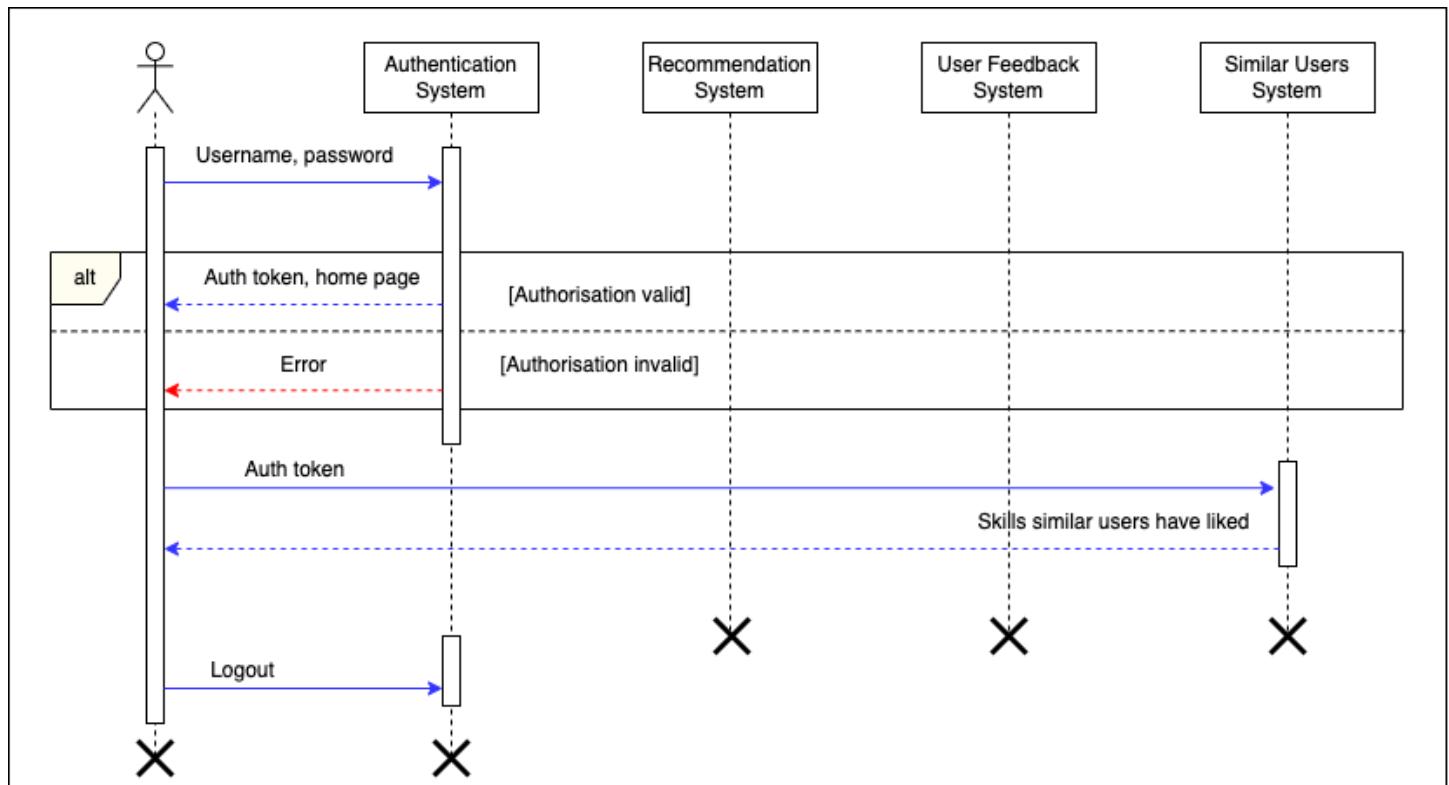


Figure 5.5: Subsection of the UML Sequence diagram in Figure 5.2 showing the user skills that similar users have liked.

Figure 5.5 is a section of Figure 5.2, illustrating users being recommended skills that similar users have liked. This feature is exclusively available to logged-in users, as it requires a user profile composed of keywords associated with the skills the user has liked. As outlined in the previous subsection, only a logged-in user can like skills and possess a user profile; therefore, only they can receive recommendations based on similar users' preferences.

This collaborative filtering recommendation feature enhances system usability and encourages deeper interaction. For a user to have a profile, and receive a personalised recommendations, they must actively engage with the system by liking skills. The roles of each system component in providing this functionality are detailed below:

- **Frontend:** Seamlessly displays various skills on the User Dashboard. Skills are iterated over and presented similarly to other recommendations to maintain design consistency, adhering to core usability principles [21]. If a user has not yet liked any skills, an informational prompt is shown, encouraging them to engage with skills to unlock recommendations based on similar users' preferences.
- **Backend:** Activates when a user accesses their dashboard, initially determining the user's group based on their profile. If a user has not been assigned to a group but has an existing profile, this profile is sent to the recommendation system for group assignment. Upon receiving the user's group, the backend identifies users within the same group, retrieves the skills they have liked, and forwards these to the frontend for display.
- **Recommendation System:** Handles incoming user keyword profiles from users without an assigned group. Since these profiles consist of NLP-generated keywords from liked skills, no further NLP processing is required. The system utilises a k-means clustering model (detailed in Section 5.3.4) to assign cluster groups. The assigned cluster, representing the user's group, is relayed to the backend.

This approach underlines the system's user-focused design, facilitating a more connected and personalised user experience.

Summary

The exploration of system navigation, underpinned by detailed UML sequence diagrams, has provided valuable insights into the user-system interaction dynamics. These diagrams have highlighted how users engage with the system's functionalities and have helped bridge the gap between conceptual design and practical implementation. By offering clear visual interactions between various system components, UML diagrams have facilitated a deeper understanding of the system's operational flow.

5.3 System

This section discusses the thorough design and implementation of each sub-system, focusing on the design decisions, technology and phased implementation.

5.3.1 Frontend

From the objectives in Section 3, there are specific objectives that relate to the design of the frontend, each of which will be discussed and exemplified through design wireframes. When designing a user interface the design must be focused around the user experience and usability of the system, particularly ensuring that accessibility standards and fundamental usability concepts are adhered to [22]. Therefore, the frontend will be designed to adhere to the Web Content Accessibility Guidelines (WCAG) [10] as well as Nielsen's Usability heuristics [21], adhering to these standards and guidelines ensures that the system is accessible and usable [22].

Colour Scheme

Consistency is one of the core usability heuristics [21], therefore it is essential to ensure that there is a cohesive colour scheme for the site. An accessible colour scheme is essential, to account for users with different needs such as colour blindness. Therefore, the colour scheme will be analysed to ensure there is enough contrast between the different colours, ensuring that information is clear to all users [23].

The investigation into prevailing design trends in existing sites, as detailed in Section 2.1, highlighted a widespread preference for minimalist colour schemes, predominantly incorporating shades of dark blue and white, accented by contrasting colours for visual depth. Coolors [24], a website that provides aesthetic,

accessible colour palettes, was used to help create and assess the colour scheme. Coolors provided insight into appropriate text colours against different backgrounds and enabled the assessment of the palette's suitability to users with various forms of colour blindness, including Achromatopsia. This comprehensive analysis is presented in Appendix A, with a summarised depiction provided below:

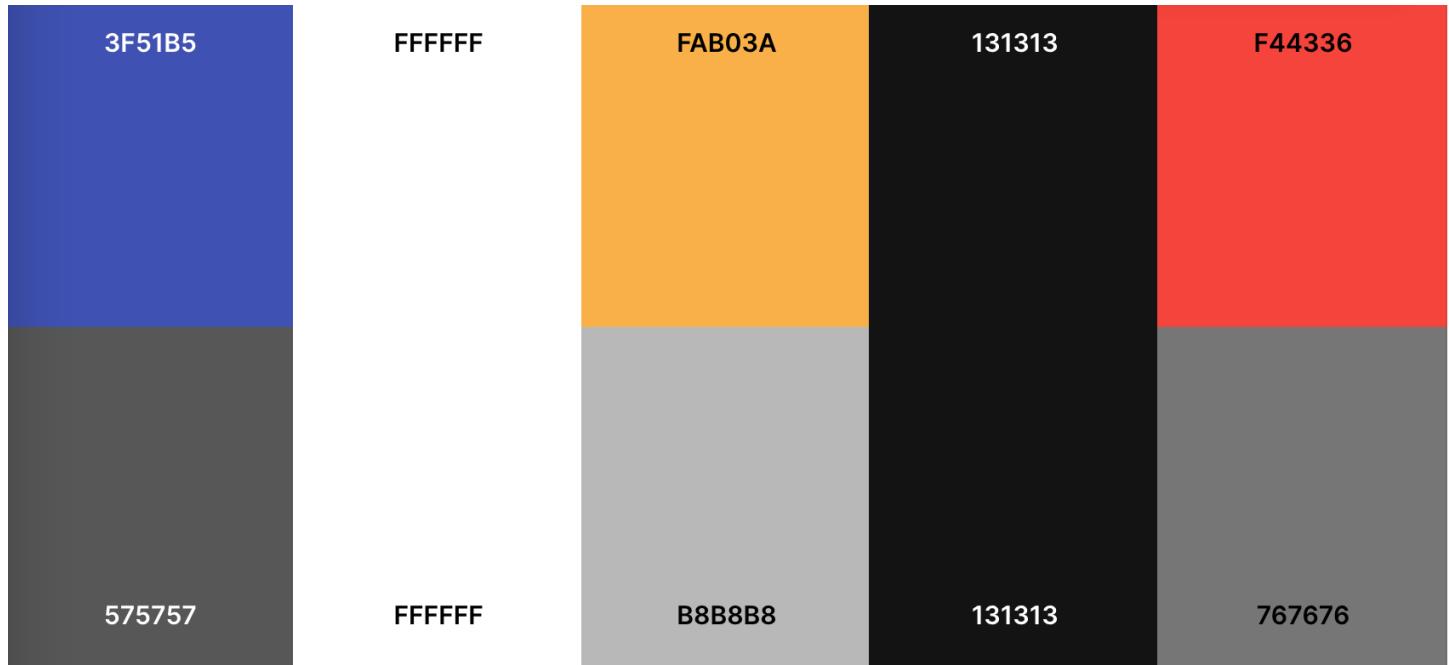


Figure 5.6: Colour palette assessment for Achromatopsia, illustrating both the selected palette and its perception by individuals with Achromatopsia.

From the analysis using the Coolors tool [24], the following colour scheme was chosen.

- **White (#FFFFFF):** Employed as the primary background colour, mirroring the common choice in the examined solutions. It doubles as a text colour when set against darker backdrops.
- **Blue (#3F51B5):** Serves as the focal colour to captivate user interest and enhance the website's visual appeal. This colour was frequently observed as a text colour in the analysed solutions, but will also be used in buttons and icons due to its high contrast to the white background.
- **Yellow (#FAB03A):** Acts as the complementary hue, enriching the website's aesthetics. The incorporation of an accent colour emerged as a common design practice.
- **Black (#131313):** Designated as the alternative text colour, it ensures optimal readability by

standing out sharply against the white background. This choice adheres to conventional text colour standards.

- **Red (#F44336):** Identified for its critical role in user interface design, red is commonly utilised to signify errors and denote interactive elements, such as 'liking' an item, resonating with established web design affordances [21].

This palette aligns with the aesthetic preferences identified in existing solutions, aiming to deliver a visually coherent and user-friendly interface. This approach ensures that the website is aligned with contemporary design preferences and accessible to a broad audience.

Branding Elements: Logo and Website Title

Following the establishment of the system colour scheme, these visual principles were applied to create the website's logo and title, encapsulating the site's ethos and enhancing its professional appearance.



(a) Skill Boost Title design

(b) Skill Boost Logo

Figure 5.7: Skill Boost branding created and drawn using Procreate [25].

The choice of Skill Boost as the system title and the incorporation of a rocket as the logo is symbolic. "Skill Boost" reflects the system's aim to elevate and accelerate users' learning and mastery of new skills, suggesting a dynamic and empowering journey. The rocket, a universal symbol of rapid ascent and breakthroughs, mirrors this ambition, encapsulating the swift progress and high-reaching goals that users can achieve through this platform.

Colour-wise, the logo integrates Blue (#3F51B5) and Yellow (#FAB03A) from the chosen palette,

promoting accessibility and user-centric design. The background of White (#FFFFFF) ensures contrast and readability.

Incorporating a professionally designed title and logo significantly contributes to the aesthetic appeal and usability of the system. It ensures that users are greeted with an aesthetic interface that is memorable. This deliberate fusion of aesthetics with functionality creates a professional appearance, enhancing usability by creating a positive and productive user experience.

Defining Web Pages

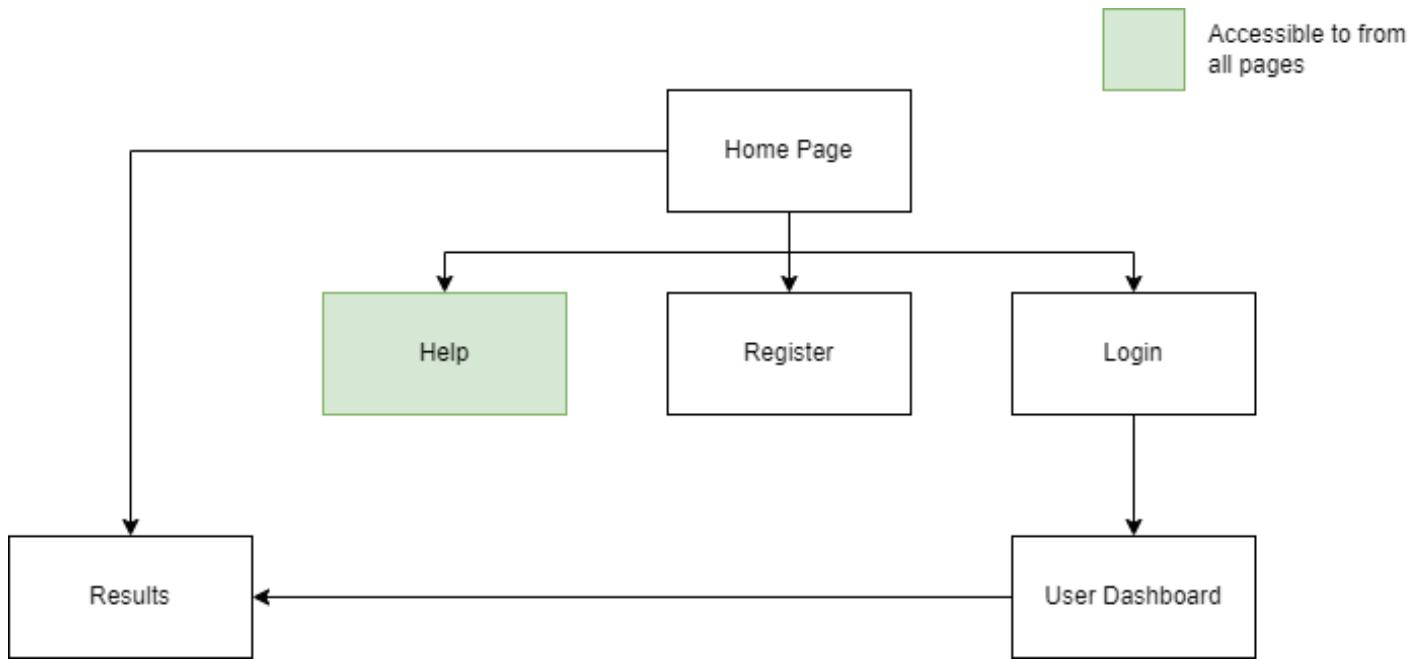


Figure 5.8: Diagram showing the navigation for the different pages on the system.

Figure 5.8 shows a state diagram that specifies the web pages constituting the frontend and the potential user navigation pathways. This diagram helps identify the essential pages and navigation functionality to be implemented, ensuring fluid transitions between different states of the application. The state diagram also outlines the different pages that need to be designed, with each corresponding page design depicted below in the form of wireframes.

Layout A consistent layout across all pages is essential for reinforcing user familiarity with the system's interface [21].

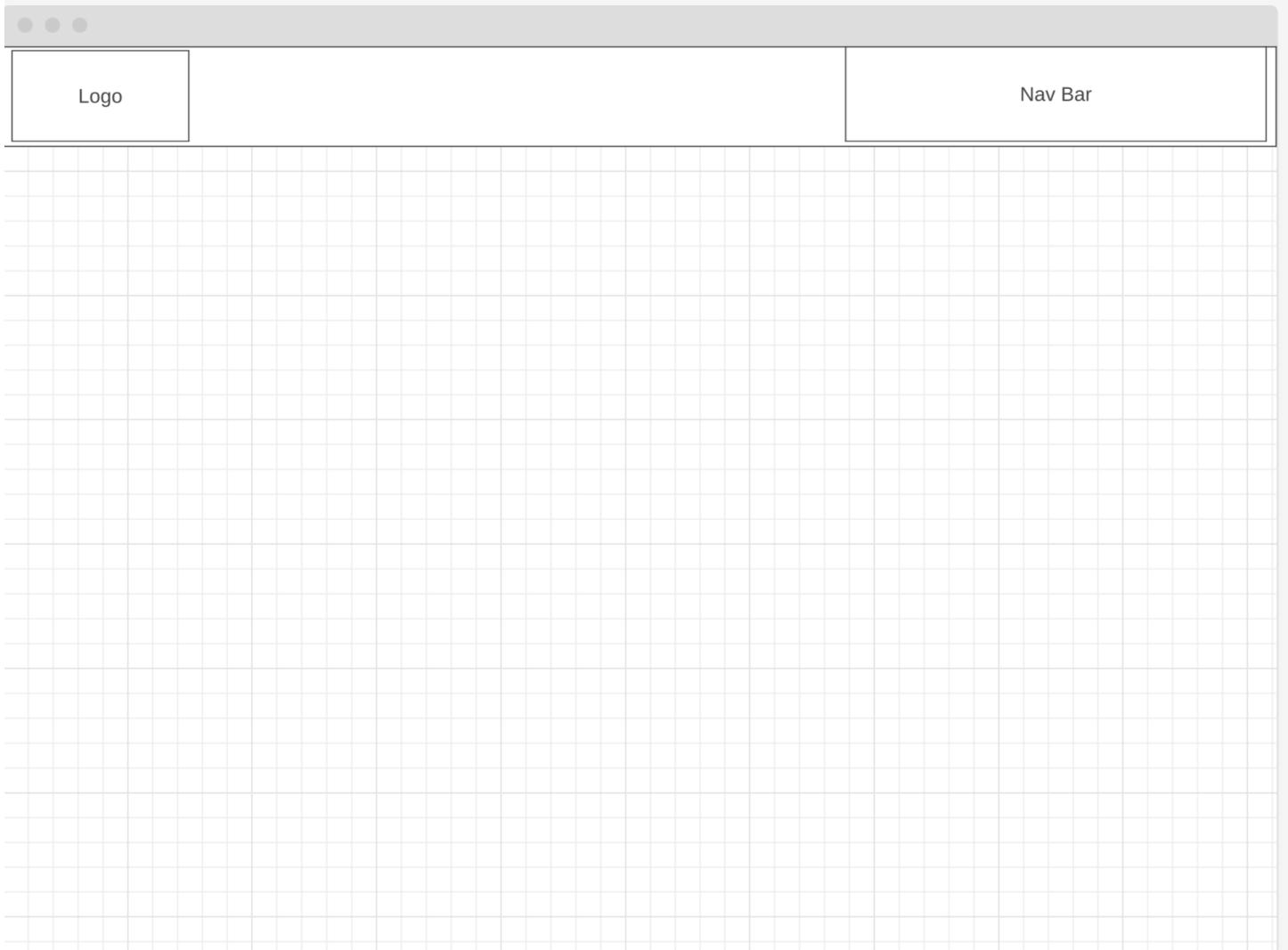


Figure 5.9: Wireframe design for the layout of each page created using a wireframe tool [26]

Figure 5.9 shows a wireframe that demonstrates the basic layout that each page will adopt. Featuring a toolbar with the logo on the top left and a navigation bar on the top right, this consistent element across pages aids user navigation and system recall. This static toolbar, preferred over an expandable side menu for its persistent visibility and consistency, ensures a uniform interface throughout the user's journey. Page-specific icons and titles within the navigation bar employ familiar affordances to intuitively signal their functionality [22], changing according to the user's current page but retaining the same layout and design principles.

Home Page Figure 5.10 shows the design of the homepage, the search bar is in the centre to draw the user to the search bar to encourage them to use the recommendation features. This is the first page that the user is presented with when they access the site. This page gives information on the system as well as allows the user to enter a topic they wish to receive recommendations for. As discussed in the previous section, both anonymous users and logged-in users should be able to use the recommendation feature of the system, and since it is the main feature of the system, the functionality will be presented to the user on the Home page.

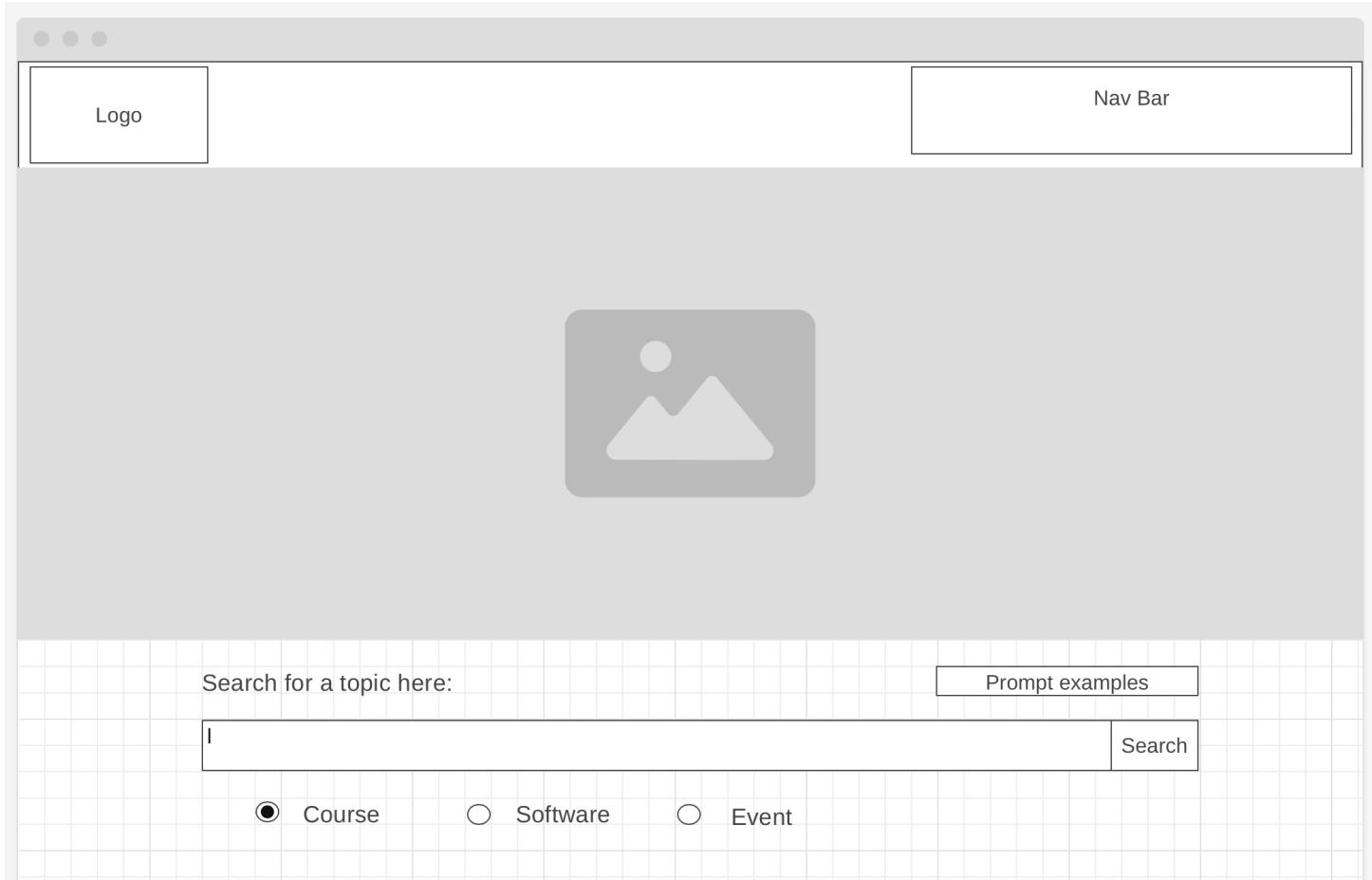


Figure 5.10: Wireframe design for the homepage created using a wireframe tool [26]

Results Page Presenting results in a clear, concise manner is crucial. Consistent display and a minimalist design approach ensure an intuitive user experience. Figure 5.11 shows the designed Results page with a grid layout, showcasing the skills with all necessary information for ease of user interpretation, adhering to cognitive load principles [27]. Existing solutions research demonstrated that recommendations included

an image to draw the user's attention and break up the text information, therefore it has also been incorporated into the design.

The grid layout suggests to the user that the items are ordered from left to right [22], whilst also adding to the aesthetic design. The users must not be overwhelmed with all of the options being recommended, users are only able to process $7 +/- 2$ items at a time [27], therefore only displaying 6 items on the screen at one time makes looking and interpreting the results easier for the user.



Figure 5.11: Wireframe design for the results page created using a wireframe tool [26]

Help Page Figure 5.12 shows the design for the Help page. A help page is essential, providing users with guidance on system navigation and usage—a core usability heuristic that enhances user support and

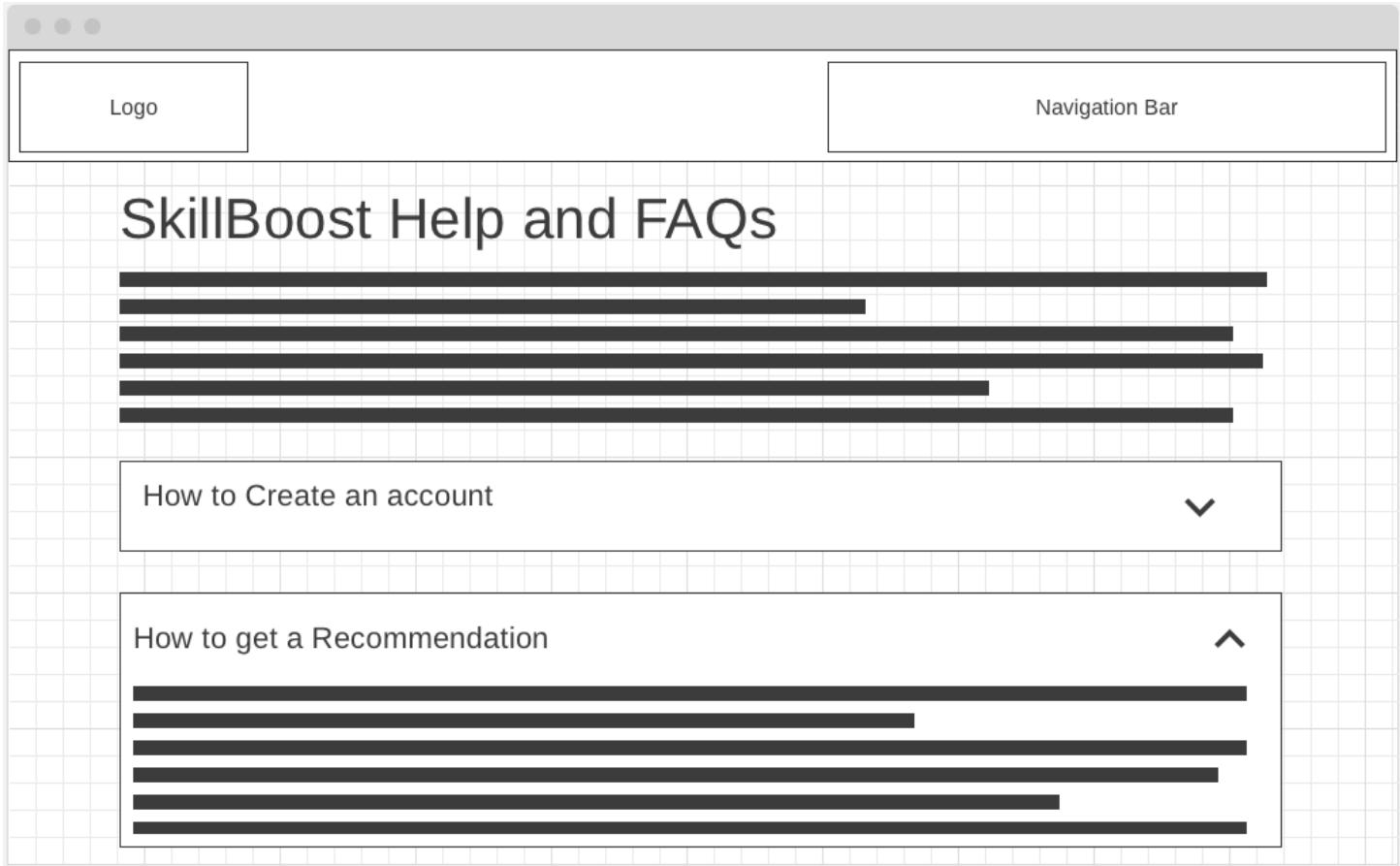


Figure 5.12: Wireframe design for the help page created using a wireframe tool [26]

system understanding [21].

Registration and Login Page Figure 5.13 shows the design for the Registration page and Figure 5.14 shows the design for the Login page. These pages are essential as the system provides users with the ability to have a user profile, therefore the ability to register for an account and login to the account is essential. They are discussed collectively due to their shared user input dynamics and the need for consistent design. These pages are designed with uniform input fields and submission buttons to maintain consistency and predictability. Users are required to provide a unique username, email for verification, and a password entered twice for confirmation. A clear indication of the submission of invalid details is provided via a red error message, aligning with the colour scheme associated with alerts, as previously discussed.

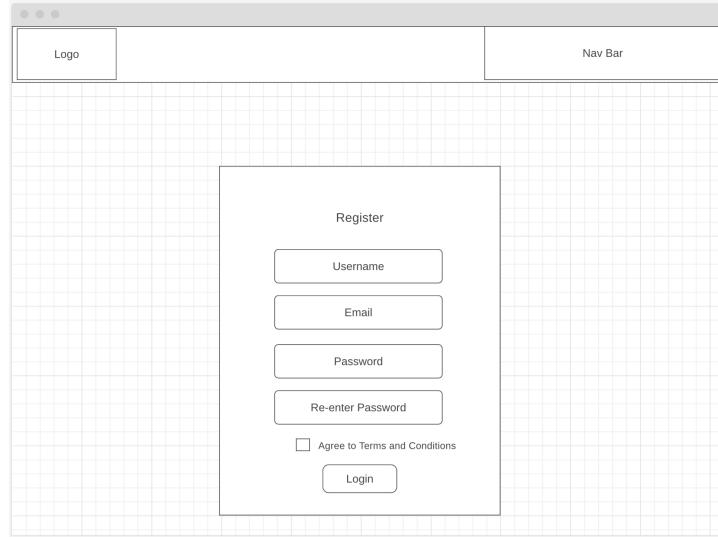


Figure 5.13: Wireframe design for the registration page created using a wireframe tool [26]

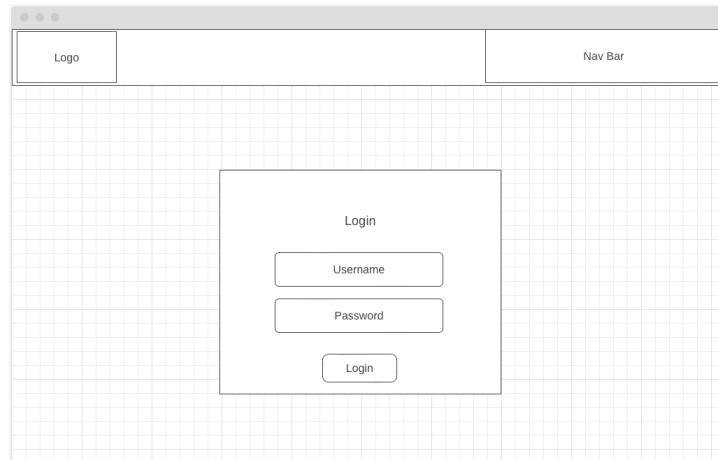


Figure 5.14: Wireframe design for the login page created using a wireframe tool [26]

User Dashboard Page Figure 5.15 shows the design for the User Dashboard page, this page acts as a personal command centre, allowing users to quickly navigate to features like new recommendations, favourites, and history. This page consolidates key functionalities, allowing users to manage their profiles and interact with the system fluidly. The layout uses a grid to provide clarity and direct access to the main features of the site. Users can start a new search, view and manage liked skills, and revisit past activities, all from one place. The 'unlike' option for skills gives users control over their preferences, aligning with usability best practices [21]. This dashboard simplifies the user's journey through the site, reflecting a design that prioritises efficiency and a streamlined experience.

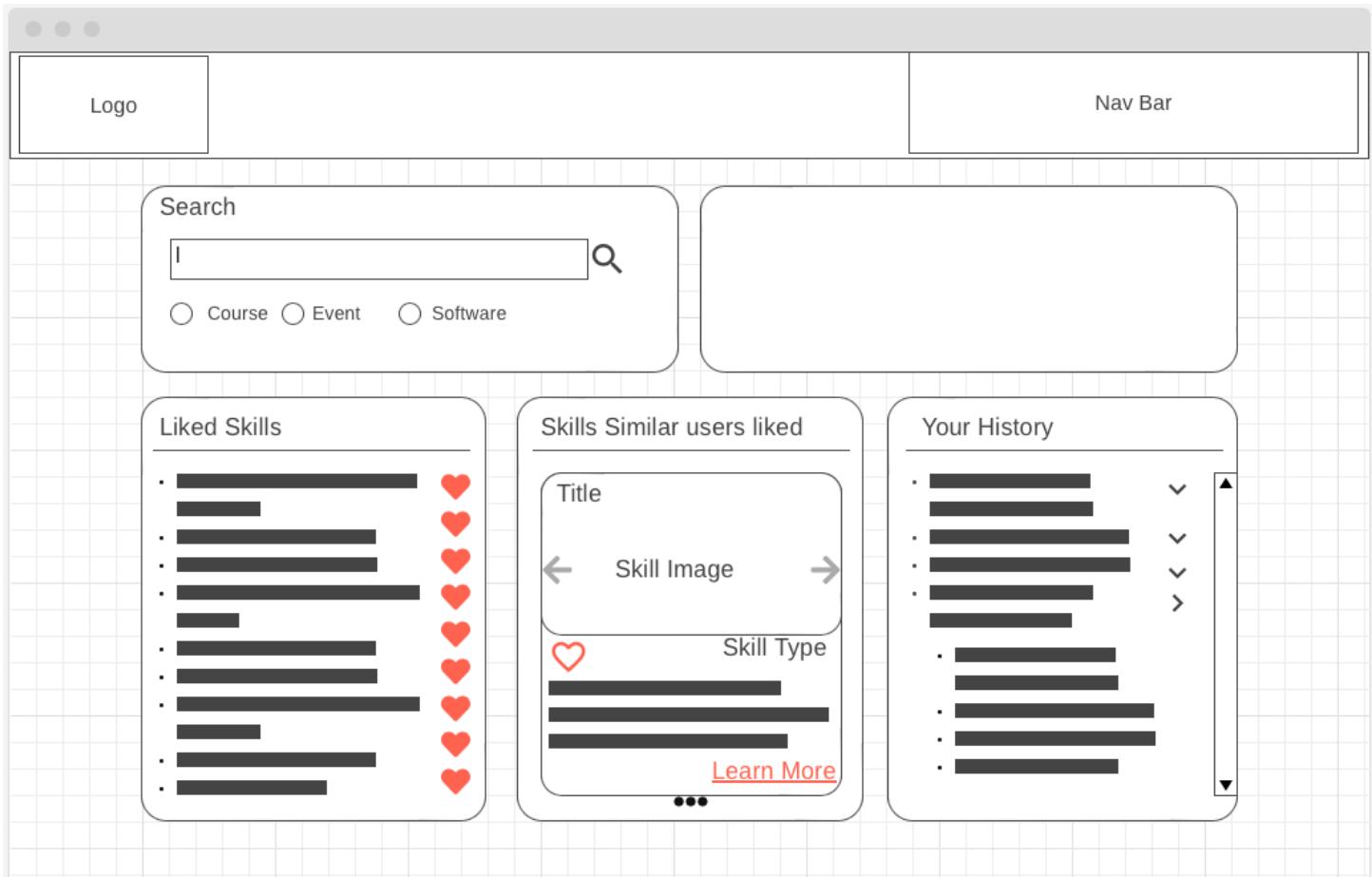


Figure 5.15: Wireframe design for the User Dashboard created using a wireframe tool [26]

Technologies Used

React for Interactive UIs: React [28], a declarative and efficient JavaScript library will be the programming language used for frontend development. Its component-based architecture allows for modular and reusable code, leading to a more organised and maintainable codebase. React's virtual DOM (Document Object Model) system provides a performance boost by minimising direct manipulation of the DOM and batch updating changes efficiently. This results in a fluid and responsive user interface, which is essential for maintaining user engagement in a web application. The use of React also facilitates the creation of stateful components, enabling the application to respond to user inputs and actions dynamically, ensuring a seamless experience.

Material UI for Design Consistency: Material UI [29], a popular React UI framework, will be integrated to streamline the design process. It provides a set of pre-designed components that follow Material Design principles, ensuring consistency across the application's interface. The use of Material UI components helps maintain a uniform look and feel while significantly speeding up the development process, as the time typically dedicated to crafting custom components is reduced. With features like customisable themes, Material UI enabled the incorporation of the defined colour scheme, maintaining brand consistency across all elements.

Axios for Backend Communication: Axios [30], a HTTP client, will be used for handling asynchronous requests to the backend. It simplifies the process of sending and receiving data from the backend API, processing the JSON responses efficiently. With Axios, user authentication, data retrieval for the recommendation logic, and the 'like' system can be implemented, all with robust error handling and a clean syntax that improve the readability and maintainability of the code.

Together, these technologies form a cohesive technology stack that help to create a scalable web application. React's dynamic capabilities, combined with Material UI's design uniformity and Axios's streamlined HTTP requests, provide a solid foundation for developing a frontend that is both engaging and consistent with modern web standards.

Phased Implementation of Frontend Pages

Creating the designs and establishing the technologies to use to create the frontend, the system was implemented according to the Agile plan discussed in Section 4.2. This plan included the iterative development of the system, building the complexity of the system through each iteration.

Phase 1 As discussed in section 4.2, the first iteration of development involved creating an MVP therefore, the essential pages were implemented first. Figure 5.16 shows the implemented Home page, Figure 5.17 shows the implementation of the first iteration of the Results page and Figure 5.18 shows the implemented Help page.

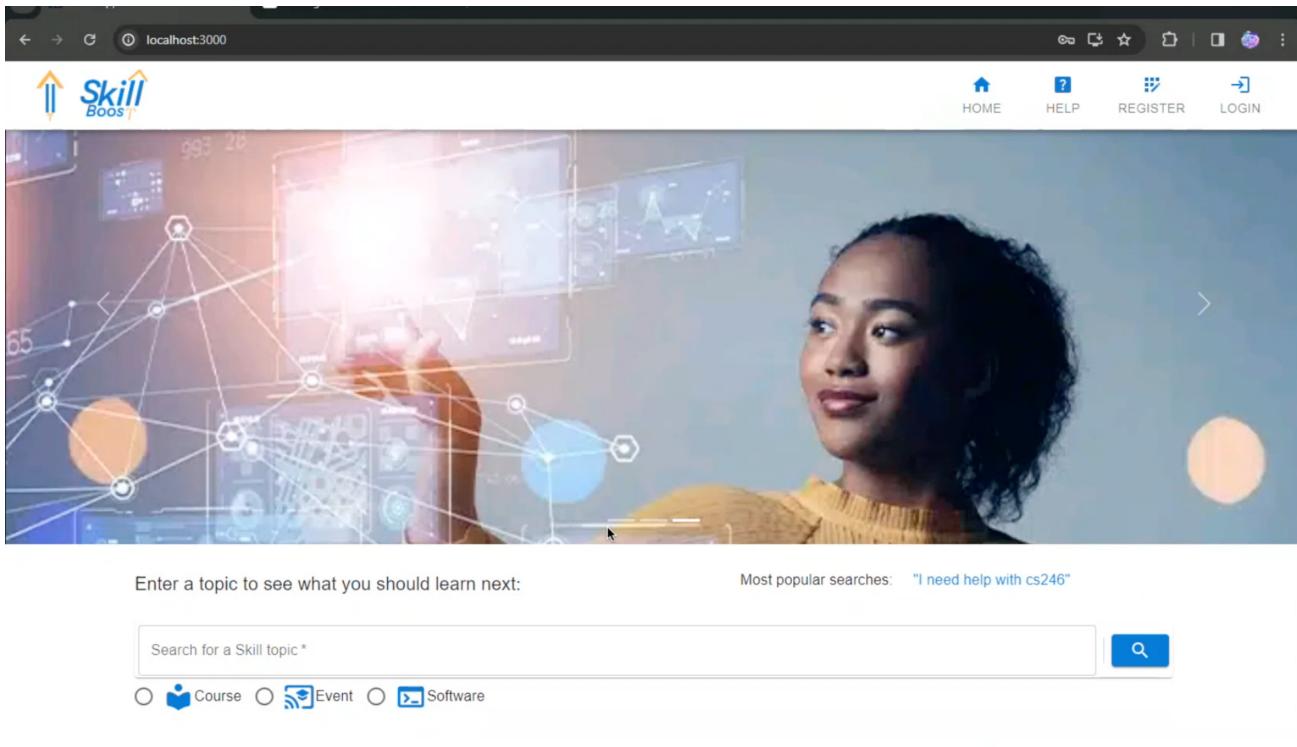


Figure 5.16: Screenshot of the Home Page

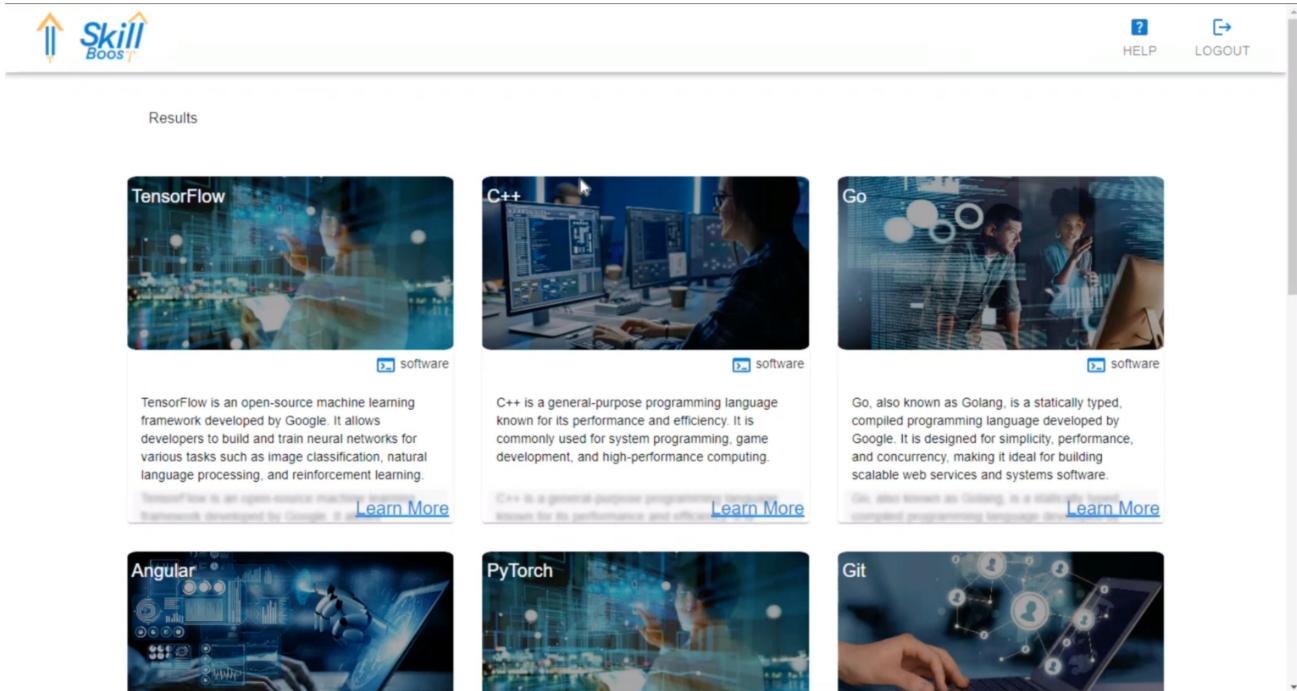


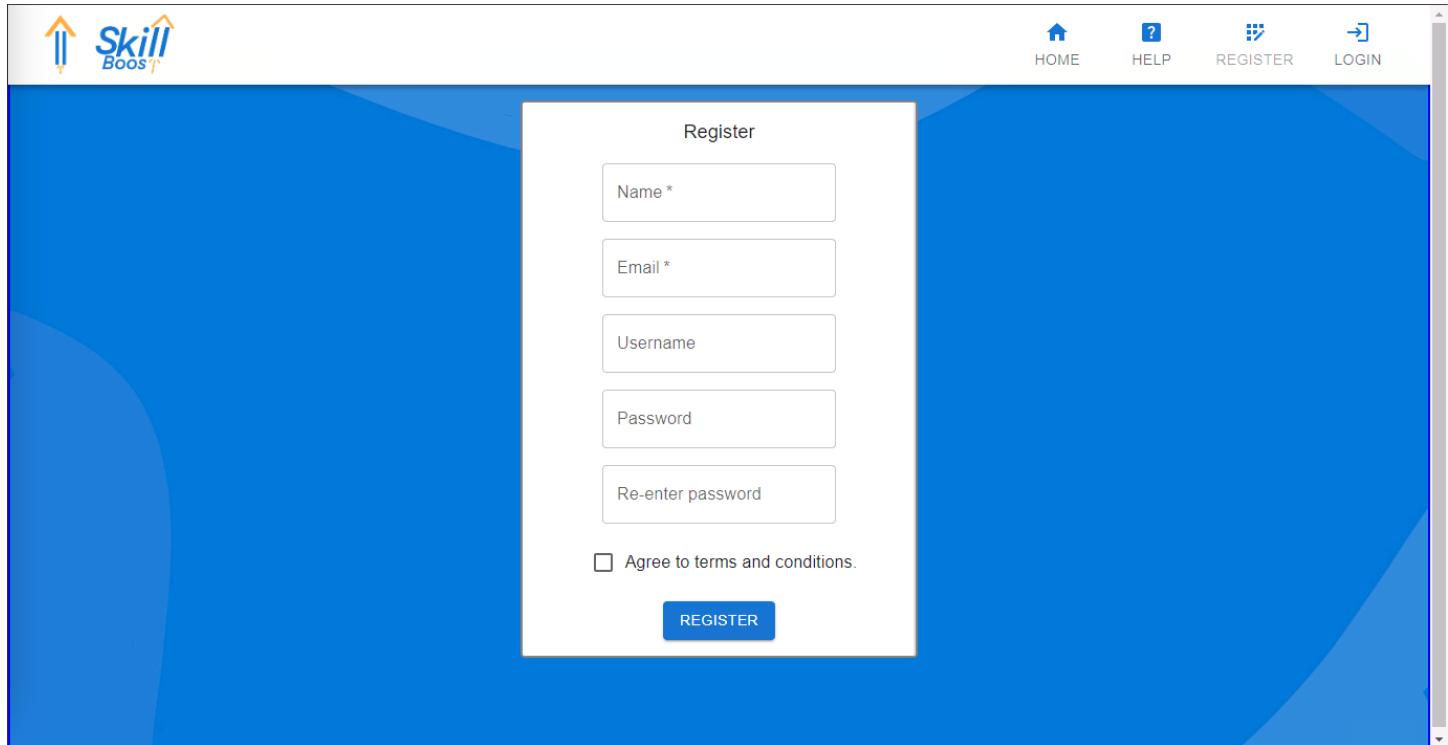
Figure 5.17: Screenshot of the initial Results Page

The screenshot shows a web browser window for the URL `localhost:3000/help`. The page title is "SkillBoost Help & FAQs". The header includes the SkillBoost logo, navigation links for HOME, HELP, REGISTER, and LOGIN, and standard browser controls. The main content area is titled "SkillBoost Help & FAQs" and contains three expandable sections: "Creating an Account", "Creating a Recommendation", and "Liking a Skill". The "Creating a Recommendation" section is currently expanded, showing instructions: "After logging in, navigate to the 'Recommendations' section. Here, you can create a new recommendation by entering a topic or area you're interested in. SkillBoost will then generate a list of skills and resources tailored to your interests." The "Liking a Skill" section also contains instructions: "In your recommendation results, you can like a skill by clicking the 'Like' button associated with each skill. Liking skills helps SkillBoost better understand your preferences and improve future recommendations."

Figure 5.18: Screenshot of the Help Page

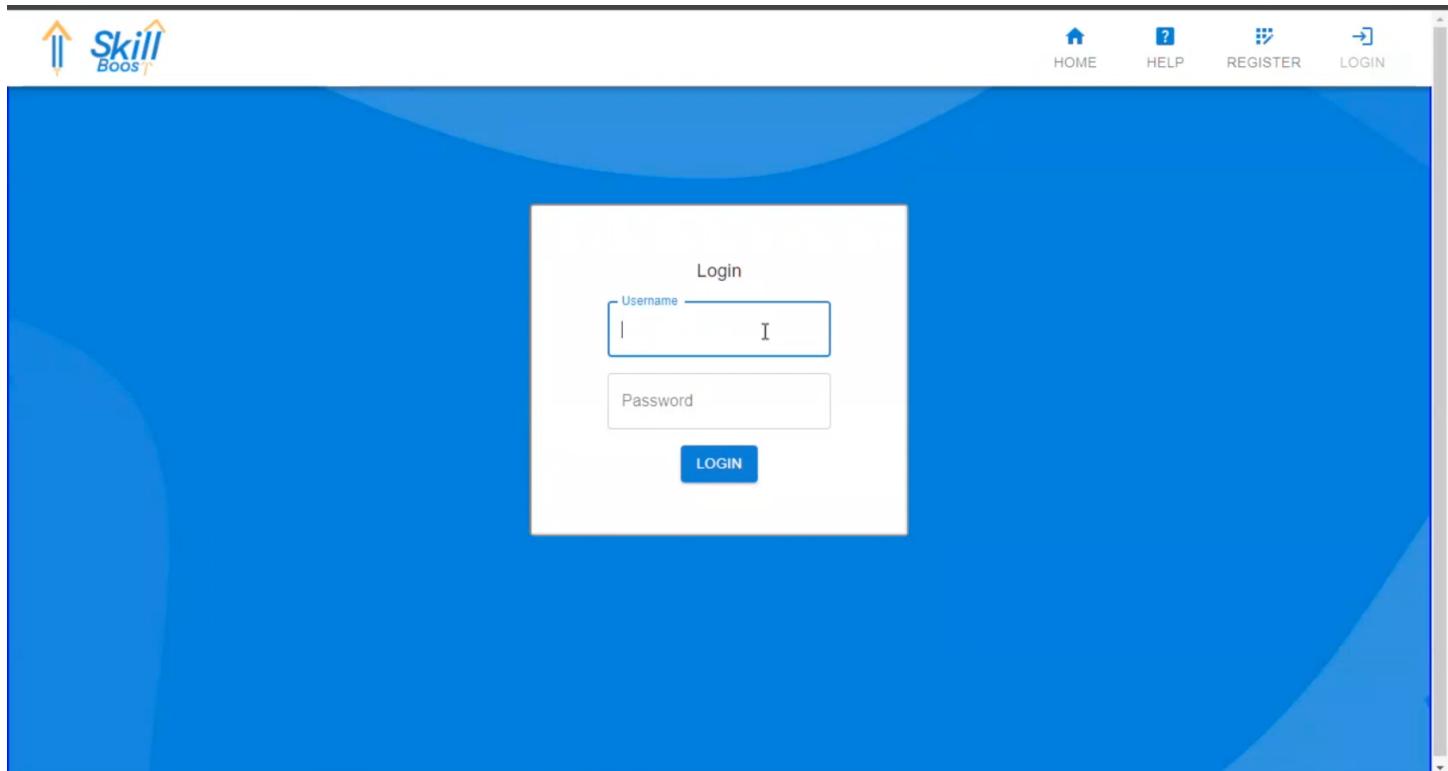
The implementation of these pages correlates very closely to each of their designs. Having a detailed wireframe for each page allowed easy and efficient implementation of the frontend.

Phase 2 The next iteration of the plan involved adding users and improving the recommendation system, therefore the associated pages for this iteration were those that concerned basic user functionality. Figure 5.19 shows the implemented Registration page and Figure 5.20 shows the implemented Login page.



The screenshot shows the registration page for the SkillBoost website. The page has a blue header with the SkillBoost logo on the left and navigation links for HOME, HELP, REGISTER, and LOGIN on the right. The main content area is titled "Register" and contains five input fields: "Name *", "Email *", "Username", "Password", and "Re-enter password". Below these fields is a checkbox labeled "Agree to terms and conditions." and a blue "REGISTER" button.

Figure 5.19: Screenshot of the Registration Page



The screenshot shows the login page for the SkillBoost website. The page has a blue header with the SkillBoost logo on the left and navigation links for HOME, HELP, REGISTER, and LOGIN on the right. The main content area is titled "Login" and contains two input fields: "Username" and "Password". Below these fields is a blue "LOGIN" button.

Figure 5.20: Screenshot of the Login Page

Phase 3 The final iteration involved adding user feedback and a user dashboard which includes recommending the user similar skills that other users have liked, this involved updating the results page to add feedback functionality as well as creating the user dashboard phase. Figure 5.21 shows the implemented User Dashboard page and Figure 5.22 shows the final results page, having added the ability to like a skill.

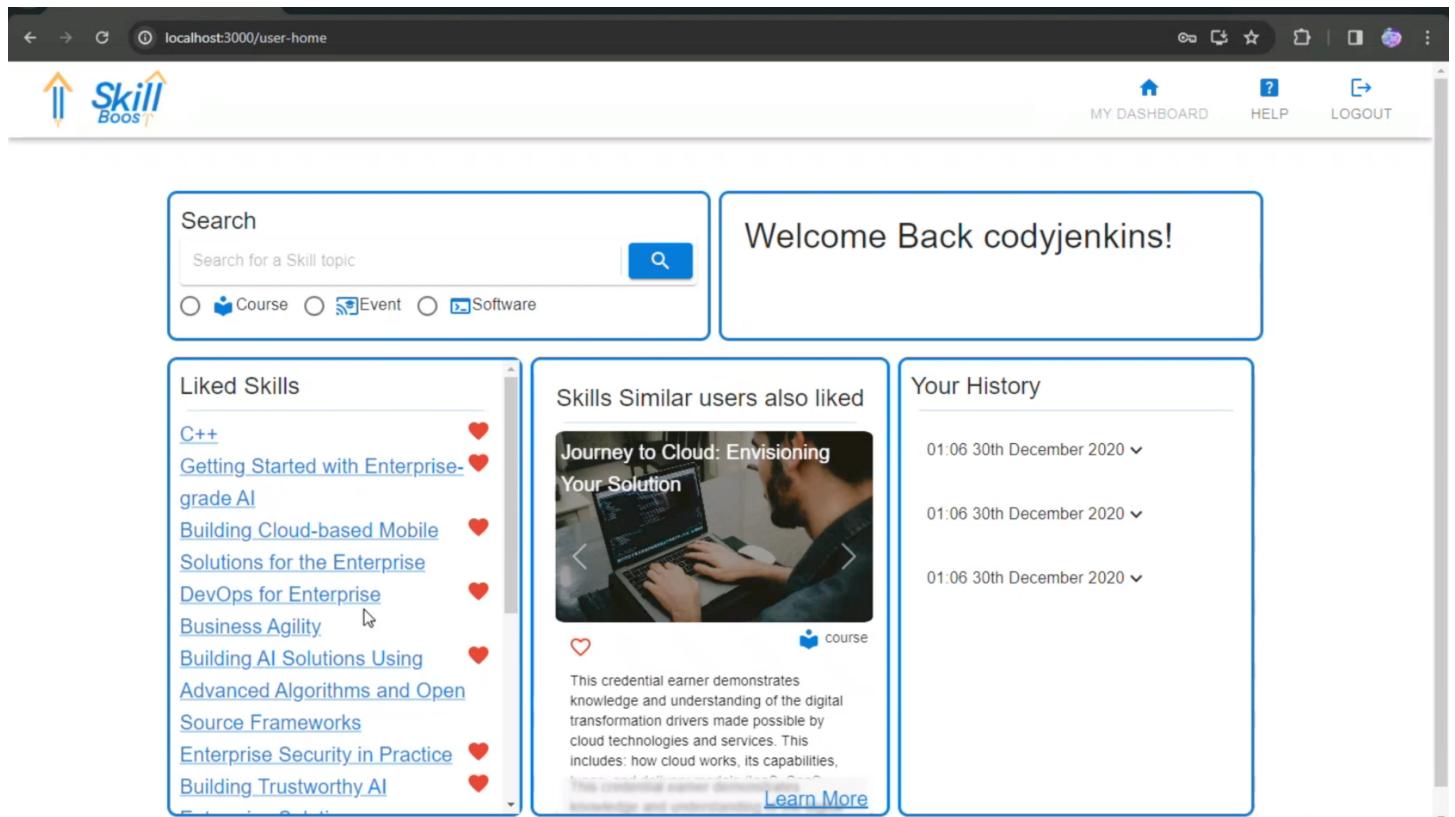


Figure 5.21: Screenshot of the User Dashboard

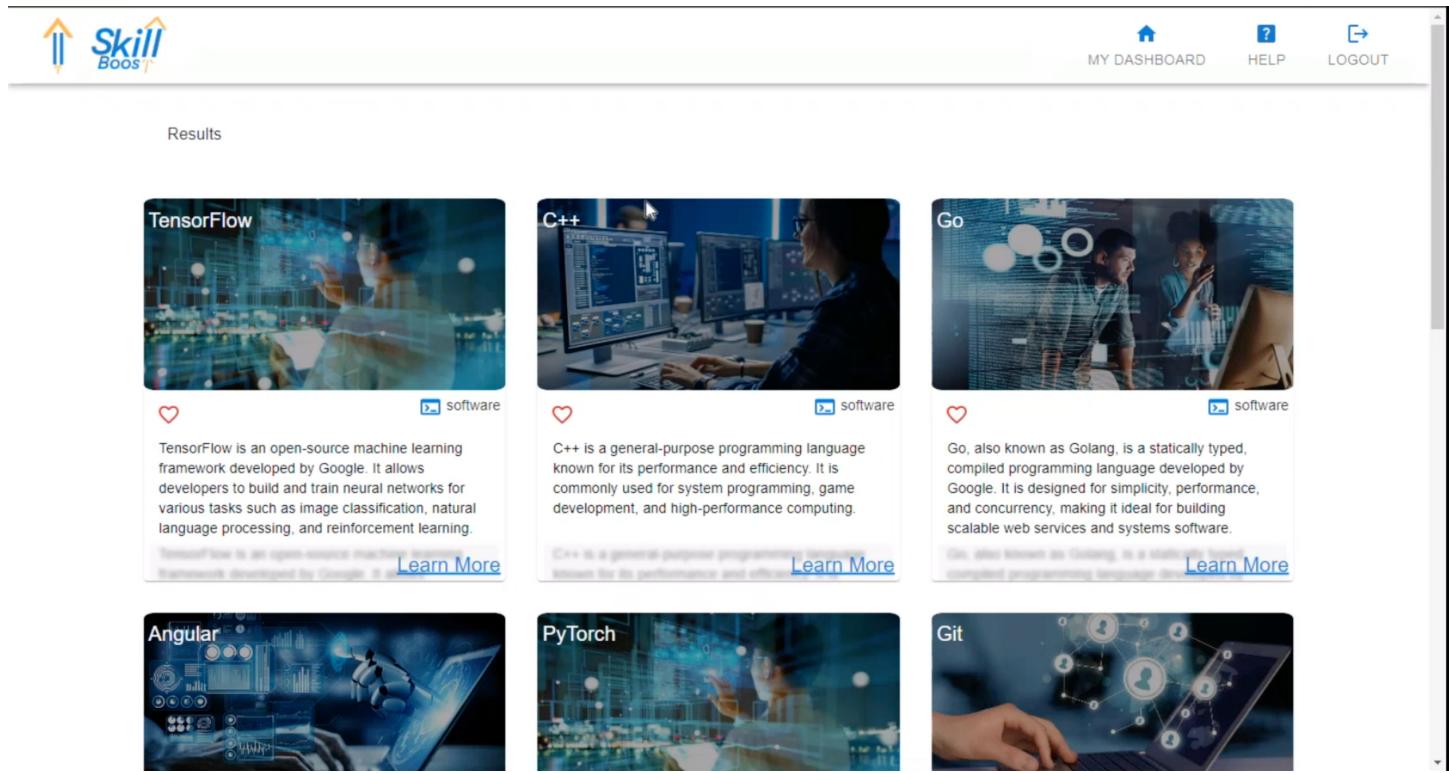


Figure 5.22: Screenshot of Results page with user feedback

Summary

The meticulous planning and design translated into a frontend that is functional and visually appealing, contributing to a cohesive and user-friendly experience on the platform.

5.3.2 Database Design and Implementation

This section discusses the dual-database architecture employed by the system. The primary database is used for the main system functionality, whereas the supplementary Natural Language Processing (NLP) database is used solely by the NLP system. Notably, the NLP component, essential for the recommendation system, could alternatively be realised through an external library, dictionary or lexicon.

MongoDB

The preference for MongoDB [31], a non-relational database system, over traditional relational databases, was informed by specific data structure considerations. Particularly within the NLP tokens dictionary, many terms lack abbreviations or alternative phrases, facilitating instances where attributes may be non-

existent. Unlike relational databases, where such attributes would require definition, often as null, MongoDB allows for their omission, thereby enhancing data storage efficiency and system flexibility. Moreover, MongoDB introduces the concept of “collections” instead of tables and “documents” for entries or rows, with object IDs serving as referential links to other collections, similar to foreign keys in relational models. MongoDB Compass was employed for database management, offering a user-friendly interface for database interactions.

Main Database

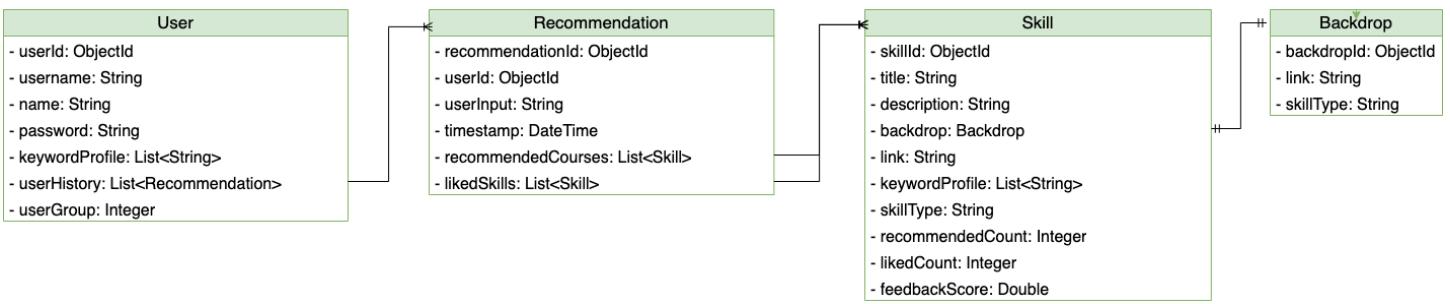


Figure 5.23: Database schema for the main system database.

Figure 5.23 shows the schema for the database, the structure comprises several pivotal collections:

Skill Collection: Encompasses IBM Skills Build items available for recommendation, featuring:

- **id** : A unique identifier for the skill.
- **title**: The skill’s name, as displayed to the user.
- **description** : Detailed information about the skill.
- **link**: The URL directing to the skill resource.
- **keyword profile**: A pre-processed amalgamation of the description and title via NLP, optimised for search efficiency.
- **skill_type**: Categorises the skill as a Course, Event, or Software.
- **recommendationCount** and **likedCount** : Metrics tracking the skill’s popularity and approval.

- **feedbackScore**: A calculated ratio reflecting the skill's reception, designed to expedite processing.

User Collection: Represents system users, incorporating:

- **id** : A unique identifier.
- **username**: For user login, uniquely identifying each user.
- **name** and **password** : Personal user information, with the password securely hashed.
- **keywordProfile**: An aggregation of keywords from the keyword profile of liked skills.
- **userHistory**: A log of recommendation IDs associated with the user.
- **userGroup**: A cluster assignment facilitating user similarity analysis.

Recommendation Collection Details each system-generated recommendation, including:

- **id** and **userId**: Unique identifiers for the recommendation and its associated user, respectively.
- **userInput**: The query submitted by the user.
- **timestamp** : The recommendation's creation time.
- **recommendedSkills**: An ordered list of skills suggested to the user.
- **likedSkills** : Tracks user preferences for specific recommendations.

NLP Database

Obtained from the Wikipedia Glossary of Computer Science terms, the NLP database facilitates enhanced token analysis through:

- **topicPhrase**, **abbreviation**, and **aka** : Enabling comprehensive term identification and replacement.
- **type**: Distinguishes between phrases and individual keywords to optimise the tokenisation process.

Incorporating Personalisation

Tailored towards Warwick University students, the system integrates specific knowledge into the NLP database, ensuring recommendations are contextually relevant. This personalisation enables the system to recognise and adapt to university-specific terminology, significantly enhancing user experience.

Summary

The database architecture is meticulously designed for scalability and adaptability, with a framework that, while presently tailored to a specific context, possesses the versatility to accommodate various scenarios, such as adaptation to other universities or resource categories. MongoDB's dynamic schema significantly contributes to this flexibility, offering a robust solution for the recommendation system's diverse and evolving data requirements.

5.3.3 Backend

This section discusses the backend system's design and implementation, focusing on the architectural decisions, functionalities, and technologies that underpin the system's capability to process information, manage database interactions, and interface with other system components effectively.

Structure

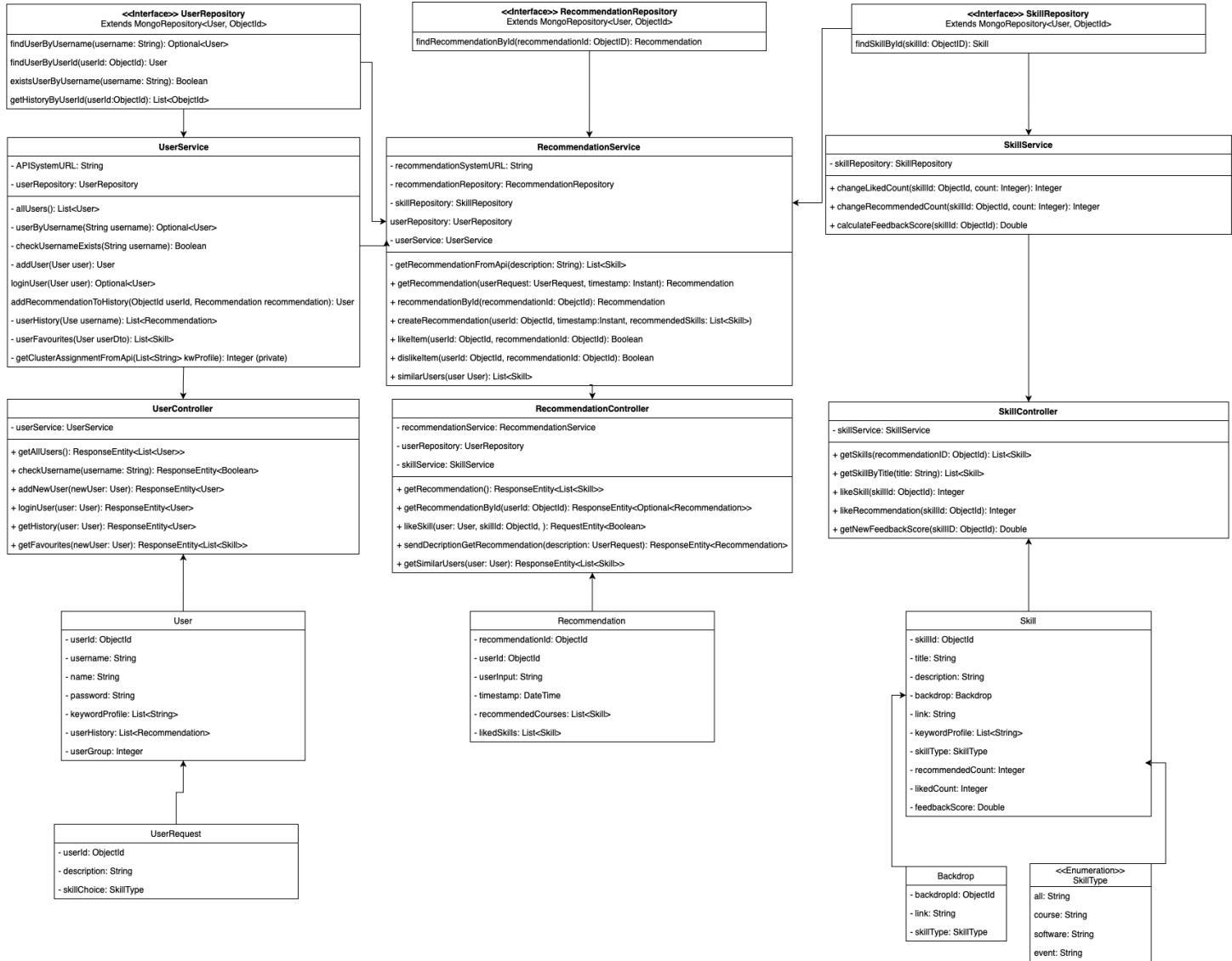


Figure 5.24: Class Diagram for the backend Domain-Driven Design Structure

Figure 5.24 shows the system class diagram, illustrating the Domain-Driven Design (DDD) approach adopted for the backend, comprising Controller, Service, and Repository layers for each domain class (Skill, Recommendation, User). This pattern is pivotal for applications with complex business logic, facilitating clear representation and separation of concerns [32]. Mirroring the MongoDB collections, this architecture ensures seamless database interactions.

Domain Driven Design Pattern Designing separate Controller, Service, and Repository layers for each domain class, ensures that each aspect of the business logic is encapsulated and managed cohesively, making the system more intuitive for development. This approach also enhances the modularity of the system, allowing each domain to evolve independently. With a dedicated set of layers for each class, it becomes easier to maintain and scale the application. For instance, if new functionalities need to be added to the Skill domain, they can be developed, tested, and deployed without affecting the Recommendation or User domains. This isolation reduces the risk of introducing bugs into unrelated parts of the system during updates [32].

Each domain class can have its complex logic encapsulated in its Service layer, promoting a rich domain model that closely represents business operations. Having a separate Repository for each domain class allows for flexibility in how data is persisted. This structure makes it easier to customise the data layer as needed, without creating dependencies across domains [32].

This method enhances the system's maintainability, scalability, and testability and ensures that the domain logic is well-represented and aligned with the system's requirements.

Functionality

The backend's primary roles include managing skill likes and generating recommendations based on similar users' preferences. The logic behind these functionalities is crucial for engaging user experiences. Section 5.2 justified why the functionality is implemented in the backend however, the specific logic of how each functionality is completed is discussed here.

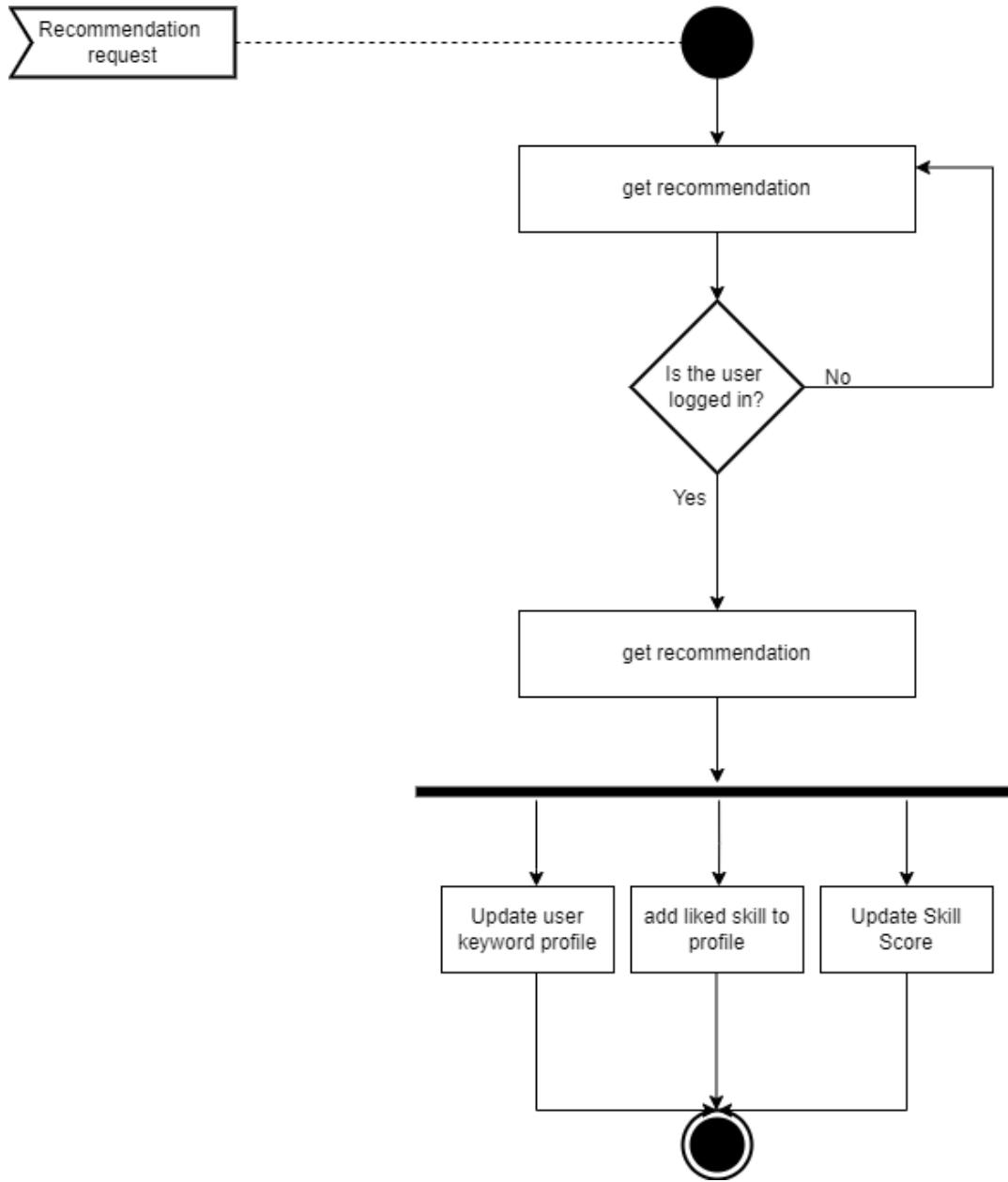


Figure 5.25: Activity Diagram for liking a skill

Liking a skill Figure 5.25 demonstrates the sequence of steps required for a skill to be liked and unliked. This is triggered when the Recommendation Controller receives a like skill post request which includes the user, the ID of the skill being liked, the skills associated recommendation ID, and a boolean attribute that indicates whether the skill is being liked or unliked.

For a skill being liked, the ID of the skill is added to the Recommendations liked skills attribute, which contains the IDs of all the skills the user has liked for that specific recommendation. The Skill is obtained

from the database using its ID and the keyword profile for the skill is then obtained then distinct keywords are added to the User's keyword profile and it is updated. The counter for the number of times the skill has been liked also needs to be incremented and updated. Once these steps are completed, the backend sends the updated user object to the frontend, so that the user profile can be updated accordingly.

The sequence to unlike a skill is similar apart from updating the user's keyword profile. Once the skill ID has been removed from the user's favourites, the keyword profile needs to be re-calculated from all of the skills in the favourites. This is because only distinct keywords are added, therefore the keyword profile for the skill being unliked cannot simply be removed as one of the keywords may have already existed because of another skill's keyword profile.

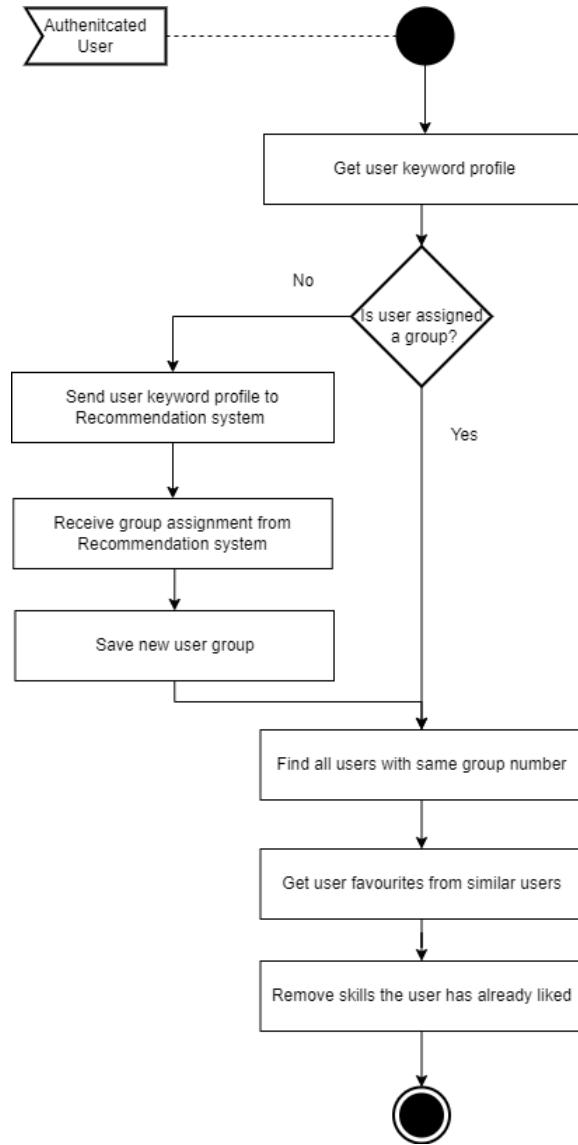


Figure 5.26: Activity diagram for finding skills similar users have liked.

Figure 5.26 demonstrates the steps required to get the list of skills for a specific user that has been liked by similar users. This method is triggered when the Recommendation Controller receives a similar user post request along with the user ID.

The recommendation service then gets the correct user from the database using the user's ID. One of the user attributes is a user group which is used to get similar users. A user may not have been assigned a group, as described in Section 5.2. If the user does not have an assigned group, the recommendation service sends the user's keyword profile to the recommendation service and receives a group number for

the user. All users with the same group are considered similar users, the details of which will be covered in Section 5.2.3. Then, the backend gets all of the users that have the same user group and gets the user favourites from each of these similar users. A distinct list of all of the user's favourites is then created and any skills that the user has already liked are removed. The list of skills is then returned to the frontend to be displayed to the user.

Technologies used

Java: This programming language was selected for its robustness, object-oriented capabilities, and extensive ecosystem, making it an ideal choice for developing complex backend systems [33]. Its compatibility with Spring Boot and Maven provide a cohesive development environment for building a scalable web application.

Spring Boot: This paradigm accelerates development by simplifying boilerplate code, configuration, and dependency management [34]. Its integration with Java and Maven supports a streamlined workflow for creating microservices and RESTful APIs, crucial for the system's architecture.

Maven: A project management tool used for building and managing any Java-based project. Maven enhances project management by automating builds and dependency management [35]. Its compatibility with Spring Boot simplifies the configuration and deployment of the backend, ensuring consistency and reliability across development cycles.

Phased Implementation of Backend Pages

The backend was developed iteratively, aligning with the agile methodology. Each phase focused on incrementally adding functionality, starting with the MVP and expanding to include user feedback and similarity-based recommendations. This approach ensured that the backend remained adaptable and scalable throughout the development process. Figure 5.27 shows the backend class diagram with details on the implementation phases for each method and class.

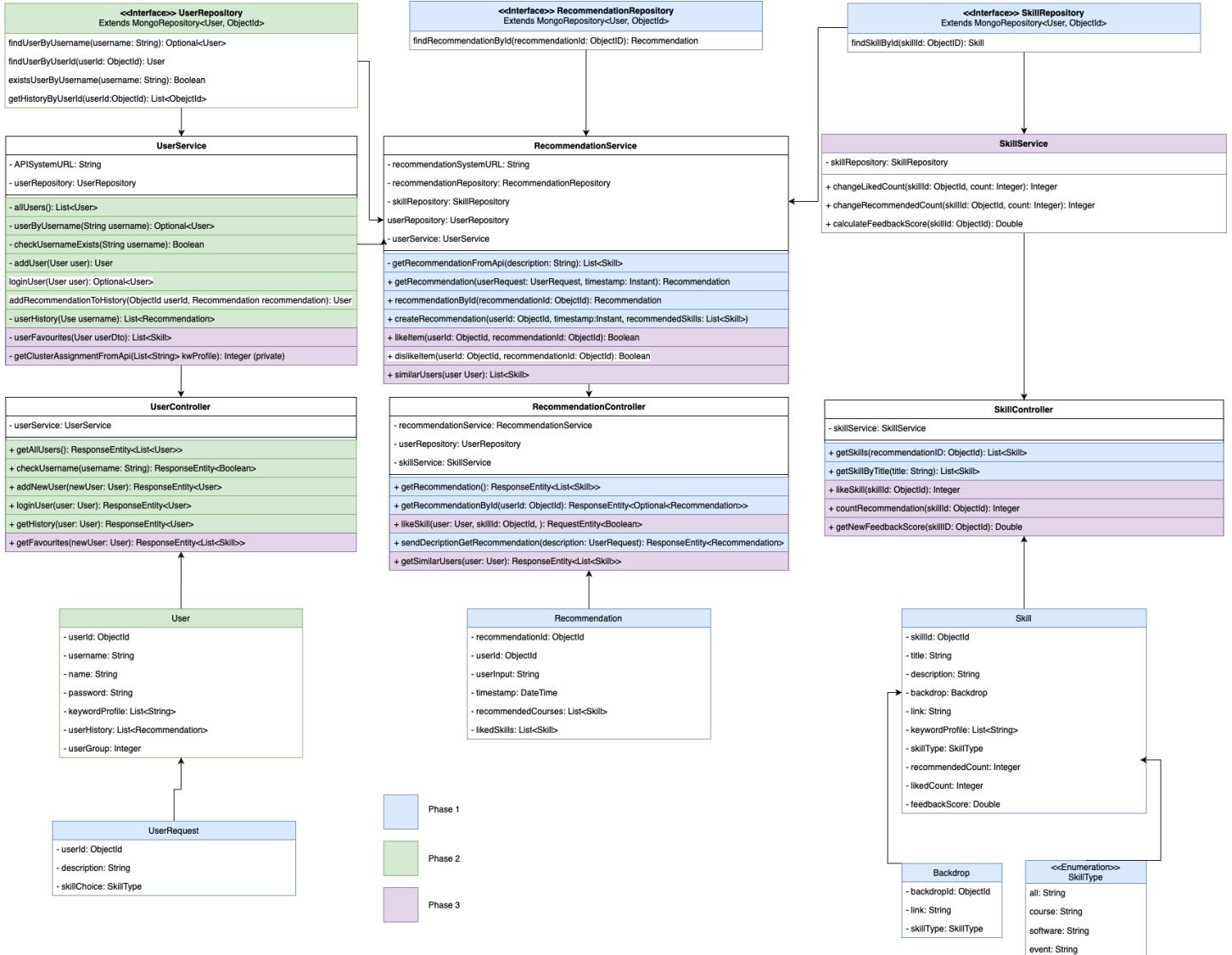


Figure 5.27: Annotated Class Diagram demonstrating the phase each method or class is to be implemented in

Phase 1 This phase involved creating the basic MVP, therefore the backend structure was developed, along with all the required database functionality, and connection with the recommendation system.

Phase 2 This phase involved users, therefore the user domain was created, along with its database functionality. Along with adding appropriate attributes, such as userId, being included in the recommendations.

Phase 3 This phase involved adding user feedback and similar users, therefore the liking skill, and similar skills functionality discussed previously was also included.

Summary

The development of the backend was characterised by careful planning, adherence to Domain-Driven Design principles, and the strategic use of Java, Spring Boot, and Maven. This foundation, combined with an agile, phased implementation, facilitated a backend system that is scalable, maintainable, and capable of supporting complex business logic and user interactions.

5.3.4 Recommendation System

This section will discuss the design and implementation of the recommendation system, which comprises two primary components: the NLP sub-system and the prediction sub-system. An overview of how these components interact within the system is provided here, with detailed discussions on each sub-system to follow in subsequent sections.

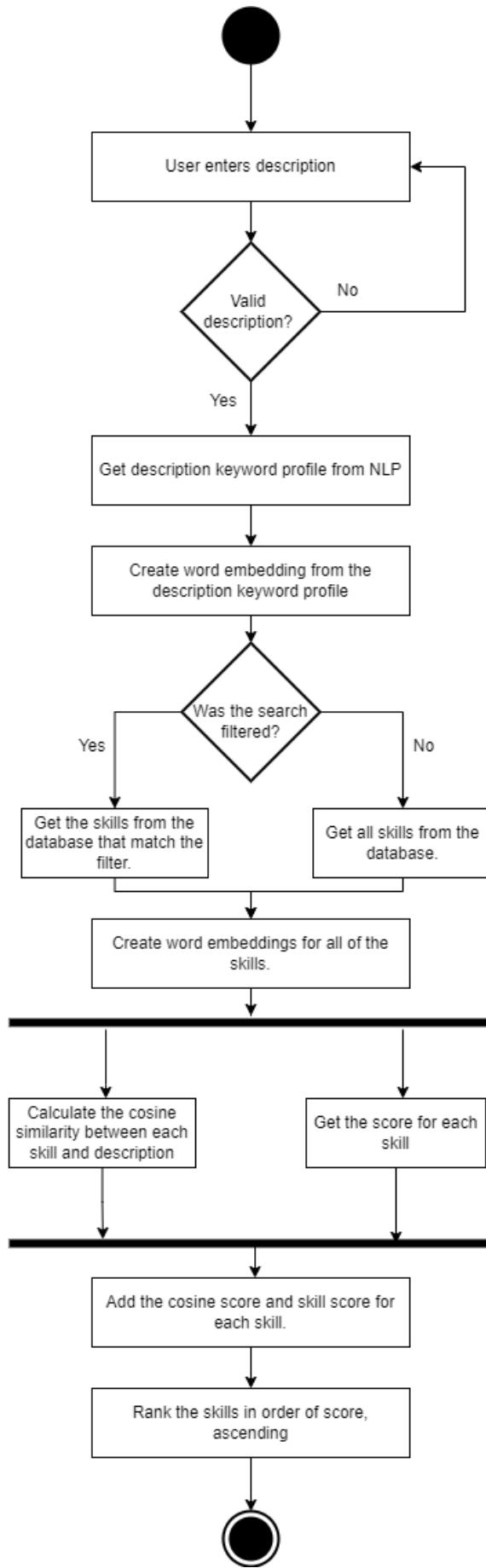


Figure 5.28: Activity diagram for the recommendation system.

The workflow of the recommendation system is as follows:

1. A description provided by the user is processed by the NLP system, transforming it into a keyword profile that represents the user's input. This process is elaborated in the NLP section.
2. The keyword profile is converted into a word embedding vector, preparing it for use by the prediction system.
3. Relevant skills are retrieved from the main database, based on the user's filters.
4. Word embeddings for the keyword profiles of each skill are generated, ensuring a uniform representation with the user's description for fair comparison.
5. The prediction system computes the cosine similarity between the user's description vector and each skill vector. Details on this process are provided in the prediction system section.
6. Feedback scores for each skill are incorporated into their respective cosine similarity scores.
7. Finally, the scores are ranked to generate the list of recommended skills.

Technologies Used

Python: Selected for its strengths in NLP and recommendation systems, underpinned by a vast ecosystem of libraries. Known for its simplicity, Python facilitates the efficient development and upkeep of complex NLP solutions. Deploying Python-based components externally ensures system performance optimisation and adherence to the separation of concerns principle, keeping the backend free from the strain of intensive computational tasks.

NLTK: The NLTK library [36] is employed for its comprehensive stop-word removal functionality, which is crucial for eliminating low-information words from textual data. This step significantly improves keyword extraction accuracy, a fundamental stage in preparing data for deeper NLP analysis within the recommendation engine.

Gensim Word Embedding Model : The Gensim Word2vec model [37] was chosen for its efficiency in creating word embeddings, which quantify semantic relationships between words. This technology is

essential for matching user queries with relevant content, enhancing the system's ability to provide accurate recommendations based on synonyms and related terms.

The adoption of Python, NLTK, and Gensim signifies a deliberate strategy to leverage established technologies in crafting an advanced recommendation system. These technologies collectively bolster the system's ability to accurately process and analyse textual data, enabling the provision of tailored content recommendations while ensuring scalability and operational efficiency.

The following sections will detail the design and implementation of each of the sub-systems.

NLP Sub-System

This subsection details the Natural Language Processing (NLP) subsystem's design and functionality, which is essential for converting user input descriptions into keyword profiles. These profiles are crafted to highlight terms related to computer science, using dictionary-based Named Entity Recognition (NER) techniques, a methodology discussed extensively in Section 2.

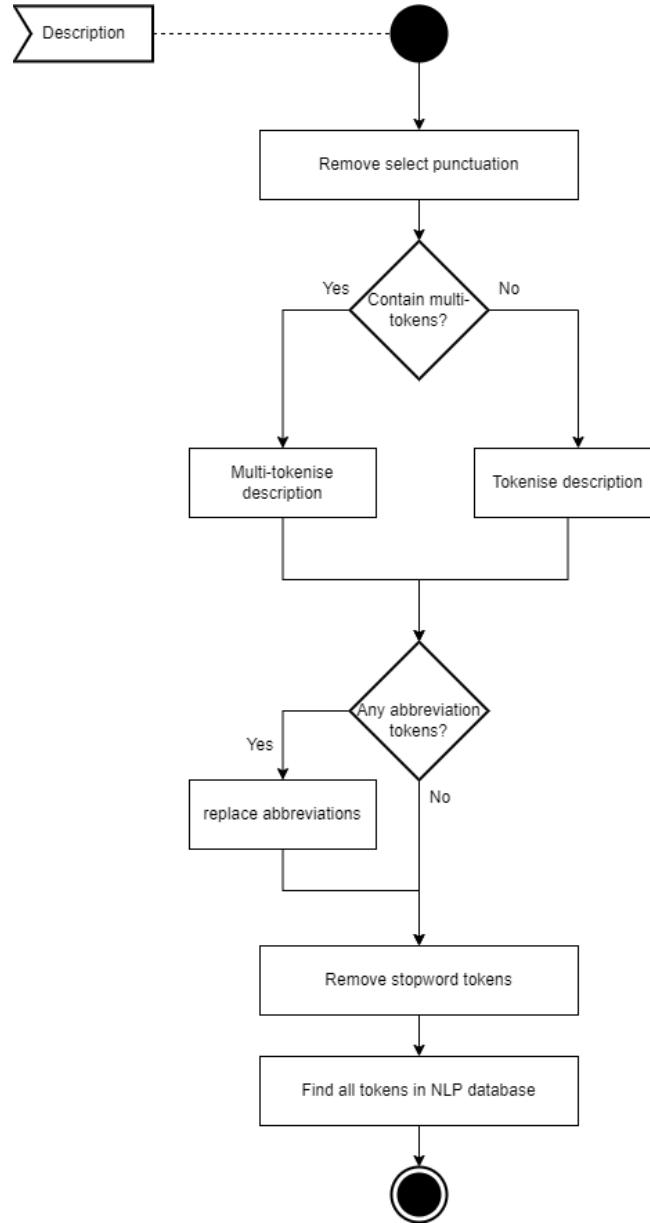


Figure 5.29: Activity Diagram Illustrating NLP Process

As seen in Figure 5.29, the system utilises a dictionary-based NER strategy to identify and extract computer science-related terms from the input text. This approach compares input against a comprehensive dictionary of relevant terms [15], creating a distilled keyword representation of the input. A pivotal aspect of deploying NLP in domain-specific contexts is the necessity to tailor conventional NLP pre-processing steps such as tokenisation, punctuation handling, and stop-word removal—to meet unique domain challenges. For example, computer science texts frequently contain key terms with embedded punctuation ("C++",

”B+ trees”), requiring a nuanced approach to pre-processing:

- **Punctuation Removal:** Pre-processing of the data found that the punctuation used within the computer science terms were #, + and . Therefore these were excluded from the punctuation being removed from the text.
- **Text normalisation:** Ensuring consistency by changing all words to lowercase to match the dictionary terms.
- **Multi Keyword Extraction:** Essential for recognising terms that should not be broken down into separate tokens, ensuring multi-token terms like “ Web Development” are accurately captured as one token. Conducting multi-keyword extraction before stop-word removal is critical, as some tokens containing stop-words may be integral to understanding the text’s context. For example “Internet of Things” is a common computer science term, but contains the stop word “of”.
- **Tokenisation:** Once all of the multi-tokens have been identified, the rest of the text can be tokenised into individual words.
- **Stop word removal:** All stop word tokens can now be removed, since all of the multi-tokens have been identified.
- **Keyword Extraction (finding tokens in the database):** Each token is checked to see if it exists in the NLP dictionary database, each dictionary item also includes known abbreviations or common alternative names “aka”, the system must adeptly handle terminology variations and abbreviations, ensuring consistency across the generated keyword profiles.

It is typical to adapt these NLP techniques to fit the specific context in which the system is deployed. Tailoring the process allows for the nuances of domain-specific language to be accurately represented and processed. This adaptive approach ensures that NLP techniques carefully adjusted to enhance the relevance and accuracy of the keyword profiles generated, reflecting a deep understanding of the domain’s linguistic characteristics.

Overall The NLP subsystem is a testament to the thoughtful application of NLP to domain-specific challenges, showcasing the careful adaptation of techniques to enhance system accuracy and user relevance. Through the integration of a dictionary-based NER and customized preprocessing steps, the subsystem effectively bridges user inputs with meaningful computer science concepts, paving the way for precise recommendations. This approach highlights the project's innovative engagement with NLP, demonstrating the importance of contextual adaptation in deploying NLP solutions effectively.

Prediction Sub-system

Figure 5.30 shows the activity diagram for the prediction system, a crucial component that directly follows the processes outlined in the latter part of the recommendation system's activity diagram.

Upon receiving a keyword profile from the NLP sub-system, a word embedding is generated for this profile. Word embeddings provide a semantic representation of words, capturing the nuances of their meanings based on usage context. Similarly, word embeddings are created for each skill's keyword profile within the database, ensuring that both user queries and skills are represented in a comparable format.

To generate a singular, composite score from the multidimensional word embeddings, Principal Component Analysis (PCA) is employed. PCA reduces the dimensions of the embedding, distilling its essence into a single score that retains the semantic significance of the original profile. This dimensionality reduction is essential for simplifying subsequent calculations while preserving the integrity of the semantic analysis.

Cosine Similarity Calculation: Similarity is calculated using the cosine method, calculating the angle between the word embedding vectors of the user's input and the skills. A score nearing 1 signifies a small angle, denoting a high resemblance between the user's input and skill.

$$\text{Cosine Similarity} = \frac{\vec{A} \cdot \vec{B}}{\|\vec{A}\| \|\vec{B}\|} \quad (5.1)$$

Incorporating Skill Feedback Score: A feedback score, reflecting user interactions, further refines the recommendation:

$$\text{Skill Feedback Score} = \alpha \frac{\text{Likes}}{\text{Top 10 Recommendations}} \quad (5.2)$$

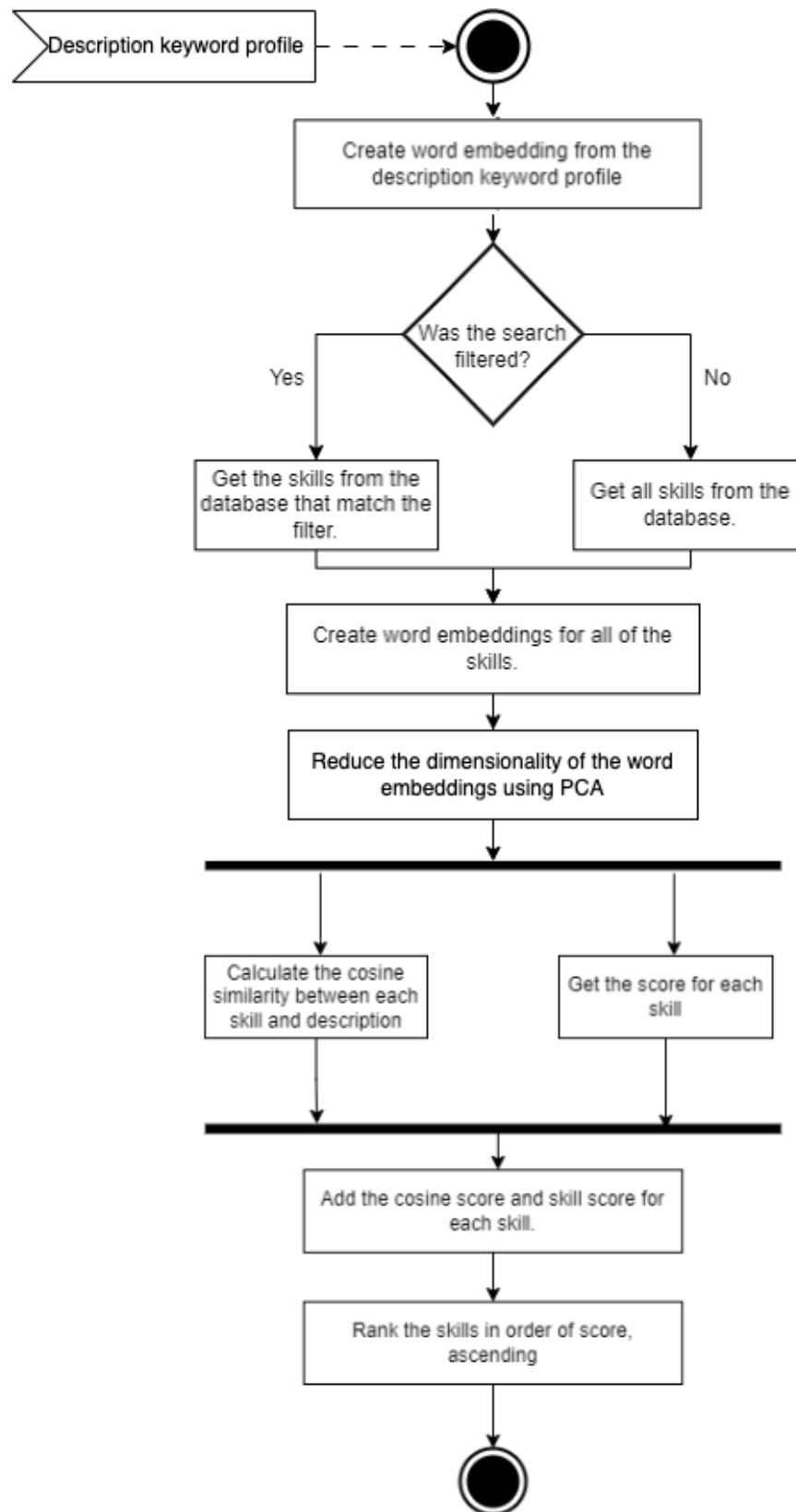


Figure 5.30: Activity diagram illustrating the prediction process for the recommendation system

An alpha value of 0.1 is selected, ensuring this score subtly influences the overall recommendation, in line with the hybrid model's ethos of refining outcomes based on user feedback.

Discussion on Limitations and Design Choices: Consideration was given to Laplace smoothing; however, its application was found to be incompatible with the score range. Similarly, the exclusion of dislikes from the feedback calculation was deliberate, aimed at maintaining score positivity and coherence within the established scoring framework.

Overall this approach encompasses both content-based filtering (cosine similarity) and collaborative filtering (feedback score) to produce recommendations that are relevant and change based on the user's preference and feedback, which are key requirements for the system.

Assigning Similar Users

This subsection discusses the approach for grouping users based on similarities in their interaction histories. Section 5.3.3 discussed what the system does with users with the same group however, this specific subsection addresses the process of assigning users to clusters when the user is yet to be assigned.

A collaborative filtering approach is used to create recommendations from user similarities, utilising k-means clustering to assign user groups.

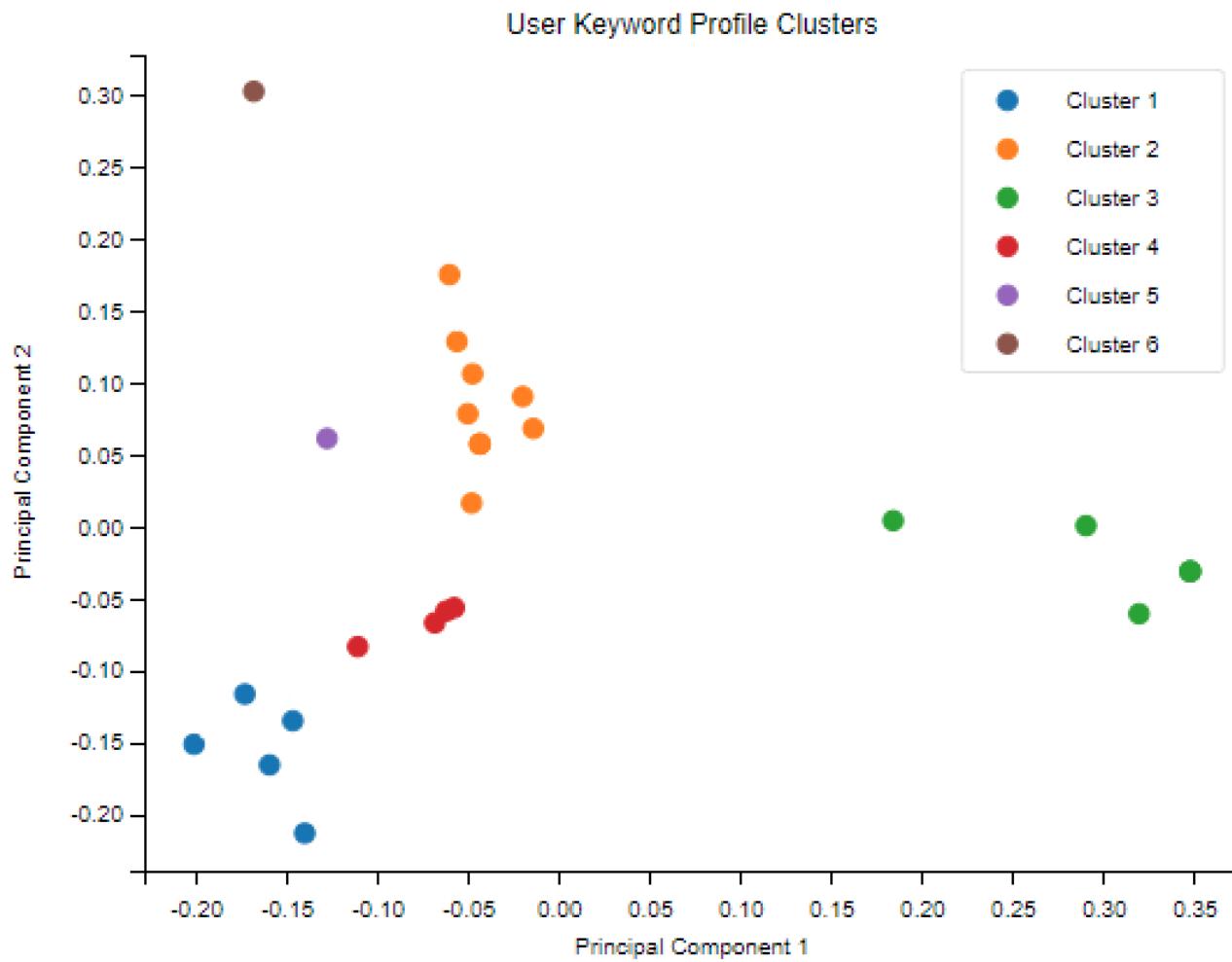


Figure 5.31: K-means clustering on the user profile with $k = 6$

Figure 5.31 shows the application of k-means clustering on each user profile. This method segments users into clusters by analysing their user profile, a collection of keywords from all of the skills the user has liked, thereby enabling the system to suggest items favoured by users within identical groups. To generate a dataset to be used to employ this technique the Python Faker library [38] was used to create a set of users with fictional, yet plausible, histories.

The k-means algorithm's implementation and subsequent testing were executed in Python, utilising matplotlib (plt) [39] for the visualisation of clustering outcomes. Analysis was conducted on the most appropriate value for k , experimental analysis revealed that setting $k = 6$ achieves an optimal compromise

for cluster granularity. Appendix B includes the analysis for other values of k . Following the model's training phase, the model was serialised ('pickled') to expedite the loading process and facilitate the swift assignment of new users to clusters based on their keyword profiles.

5.4 Summary

This chapter presents a holistic view of the recommendation system's design and implementation, covering the system's architecture, user interface, backend processes, database management, and the intricacies of the recommendation engine. This approach ensures the delivery of a robust, user-centric recommendation system that is both scalable and efficient.

Chapter 6

Evaluation

This chapter explores the evaluation methods used upon completing the project's implementation, focusing on essential testing and various evaluation strategies. The objective of testing is to ensure that the system operates as intended, confirming its reliability and functional integrity. Additionally, the evaluation process aims to verify that the system provides relevant recommendations and complies with the requirements outlined in Section 3, whilst also meeting the user's expectations. The evaluation seeks to meet user expectations, serving as a critical measure of the system's effectiveness and performance, with testing carried out at the appropriate stage relevant to the sprint or iteration.

6.1 Unit and Integration Testing

Unit and integration testing are crucial components of the project, aligning with the system's requirements and operational criteria.

6.1.1 Test Plan

The test plan is designed to be iterative, reflecting agile practices. This approach ensures tests are relevant and timely, corresponding directly to the latest developments and functionalities within the project.

6.1.2 Unit Testing

Unit tests are conducted to ensure individual components operate correctly in isolation, using specific libraries tailored to the project's diverse technology stack:

Frontend

Table 6.1 shows the test plan for the frontend unit tests, these tests ensure that the UI behaves as expected under various user interactions.

Test Case ID	Description	Expected Result	Pass/Fail	Comments
Login Functionality				
1.1	Login with valid username and password	User successfully logs in and is redirected to the main page.	Pass	Verify successful login message is displayed.
1.2	Login with invalid username (negative test)	Error message indicating invalid username is displayed.	Pass	Verify the error message is clear and specific.
1.3	Login with invalid password (negative test)	Error message indicating invalid password is displayed.	Pass	Verify the error message is clear and specific.
1.4	Login with empty username and password (negative test)	Error message indicating missing credentials is displayed.	Pass	Verify the error message is clear and specific.
1.5	Logout functionality	User is successfully logged out and redirected to the login page.	Pass	Verify successful logout message is displayed (optional).
Topic Search Functionality				
2.1	Search for a broad computer science topic ("machine learning")	Relevant recommendations for courses, software, and events are displayed.	Pass	Verify a variety of recommendations are displayed.
2.2	Search for a specific computer science topic ("natural language processing")	Relevant recommendations for courses, software, and events are displayed.	Pass	Verify recommendations are more focused on the specific topic.
2.3	Search for a non-existent computer science topic ("oranges")	Informative message indicating no results found is displayed.	Pass	Verify the message suggests alternative search terms (optional).
2.4	Search for a topic with special characters (e.g., "ai")	Relevant recommendations are displayed based on the matched keywords.	Pass	Verify abbreviations are replaced correctly.
2.5	Search suggestions appear as user types	Search suggestions relevant to the user's input are displayed in a dropdown menu.	Pass	Verify suggestions are helpful and accurate.
Recommendation Display				
3.1	Recommendations displayed after a search	Courses, software, and events are displayed in a clear and organized manner.	Pass	Verify each recommendation section (courses, software, events) is present.
3.2	Relevance of recommendations	Recommendations are relevant to the searched topic.	Pass	Verify titles and descriptions of recommendations reflect the search topic.
3.3	Recommendation details	Each recommendation includes a clear title, description, and link.	Pass	Verify links open the relevant webpage when clicked.
3.4	Recommendation link functionality	Clicking on a recommendation link opens the relevant webpage in a new tab/window.	Pass	Verify the link opens the correct webpage without errors.
User Profile and History Access				
4.1	User profile display	User profile displays user history, similar users and liked skills.	Pass	Verify the displayed information is accurate and up-to-date.
4.2	Access to recommendation history	User can access their past searches and recommendations.	Pass	Verify the history is organized and easy to navigate.
4.3	Accuracy of recommendation history	History accurately reflects past searches and recommendations.	Pass	Verify timestamps and details of recommendations are correct.
4.4	Navigation back to main search page	User can navigate back to the main search page from their profile.	Pass	Verify clear navigation buttons or links are present.
Liked Skill Functionality				
5.1	"Like" functionality	User can "like" recommended skills from the search results.	Pass	Verify a like button or icon is present for each skill.
5.2	"Dislike" functionality	User can "dislike" a skill they have already liked	Pass	Verify a like button changes and the skill is removed from the liked skills section.
5.3	Saving liked skills	Liked skills are saved in the user's profile.	Pass	Verify liked skills are persistent and accessible after logout (if applicable).
5.4	View liked skills	User can view their liked skills in a dedicated section.	Pass	Verify the liked skills section is clearly labelled and easy to find.
5.5	Similar user liked skills	System displays "liked skills" from similar users.	Pass	Verify displayed skills are relevant to the user's interests.
5.6	Relevance of similar user liked skills	Similar user liked skills are related to the user's search history and liked skills.	Pass	Verify a clear connection exists between displayed skills and user's profile.

Table 6.1: Frontend sub-system unit test plan.

Backend

Table 6.2 shows the unit test plan for the backend. This utilises JUnit [40] for unit testing, focusing on the reliability of server-side logic and data manipulation.

Test Case ID	Description	Expected Result	Pass/Fail	Comments
User Authentication				
1.1	Login with valid username and password	User token is successfully generated and returned.	Pass	Verify user can access the API after successful login.
1.2	Login with invalid username (negative test)	API returns an error message indicating invalid credentials.	Pass	Verify the error message is clear and specific.
1.3	Login with invalid password (negative test)	API returns an error message indicating invalid credentials.	Pass	Verify the error message is clear and specific.
1.4	Login with expired or invalid token (negative test)	API returns an error message indicating invalid token.	Pass	Verify the system requires a valid user session for data access.
User Data Retrieval				
2.1	Retrieve user profile information	User information (username, email, etc.) is retrieved and returned in the response.	Pass	Verify retrieved data matches the user's profile.
2.2	Retrieve user search history	User's past searches are retrieved and returned in a structured format.	Pass	Verify search history includes timestamps and relevant details. (This should align with Frontend Test Case 4.3)
2.3	Retrieve user liked skills	User's liked skills are retrieved and returned in a structured format.	Pass	Verify retrieved skills match the user's selections. (This should align with Frontend Test Case 5.3)
2.4	Handle non-existent user data (negative test)	API returns an informative message indicating the user data is not found.	Pass	Verify the system gracefully handles invalid user requests.
Recommendation System API Interaction				
3.1	Send user description and retrieve recommended skills	List of recommended skills is retrieved from the recommendation system API.	Pass	Verify successful communication and data exchange between APIs.
3.2	Validate successful communication with the recommendation system API	API receives a successful response from the recommendation system API.	Pass	Verify the status code and response structure indicate successful communication.
3.3	Handle unavailable recommendation system API (negative test)	API returns an informative message indicating the recommendation system is unavailable.	Pass	Verify the system gracefully handles external API failures.
3.4	Handle malformed response from recommendation system API (negative test)	API returns an informative message indicating an error occurred while processing the recommendation request.	Pass	Verify the system can handle unexpected responses from external APIs.
User Description Processing				
4.1	Process user description with common keywords (positive test)	User description is parsed and relevant keywords are extracted for recommendation.	Pass	Verify the system can identify relevant keywords from user input. (This should align with Frontend Test Case 2.1 & 2.2)
4.2	Process user description with special characters (positive test)	User description with special characters is parsed correctly and relevant keywords are extracted.	Pass	Verify the system can handle special characters without errors. (This should align with Frontend Test Case 2.5)
4.3	Process user description with empty or minimal content (negative test)	API returns an informative message indicating insufficient information for recommendations.	Pass	Verify the system identifies invalid user descriptions that cannot be processed. (This aligns with Frontend Test Case 2.3 - negative test with empty search)
Data Validation and Error Handling				
5.1	Handle invalid user login credentials (negative test)	API returns a specific error code for invalid login attempts.	Pass	Verify the system validates user credentials and returns appropriate error codes.
5.2	Handle invalid user ID for data retrieval (negative test)	API returns an informative message indicating the user data cannot be found.	Pass	Verify the system validates user IDs to prevent unauthorized data access.
5.3	Validate user description length (negative test)	API returns an informative message indicating the user description is too short or empty.	Pass	Verify the system enforces minimum requirements for user input. (This partially aligns with Frontend Test Case 2.3 - negative test with empty search)
5.4	Return informative error messages for invalid requests (positive test)	API error messages clearly describe the issue encountered in the request.	Pass	Verify error messages are helpful for developer debugging and troubleshooting API integration.
Retrieving Skills Liked by Similar Users:				
6.1	Retrieve skills liked by users within the same cluster as the current user (positive test)	Skills popular among users with similar interests are retrieved.	Pass	Verify the system recommends skills relevant to the user's cluster.
6.2	Verify retrieved skills are relevant to the user's interests based on their cluster (positive test)	Recommended skills should align with the user's keyword profile and the learning interests of their cluster.	Pass	Verify recommendations cater to the user's specific learning goals.
6.3	Handle cases where no similar users.	Text display to say the user must search for a topic to have a similar user section.	Pass	Verify that the no similar users message is displayed in the section where the similar users skills is meant to be displayed.

Table 6.2: Backend sub-system test plan.

Recommendation System

Table 6.3 shows the unit test plan for the recommendation system. Python's unittest library [41] supported the testing of the recommendation system's algorithms and NLP processing.

Test Case ID	Description	Expected Result	Pass/Fail	Comments
Natural Language Processing (NLP) Sub-system:				
1.1	Process user description with common computer science keywords (positive test)	Keywords like "machine learning", "artificial intelligence", etc. are extracted.	Pass	Verify core NLP functionalities are working.
1.2	Process user description with synonyms and related terms (positive test)	Synonyms and related terms like "deep learning" for "artificial intelligence" are extracted.	Pass	Verify NLP can handle variations in user language.
1.3	Process user description with special characters and noise (positive test)	Keywords are extracted despite special characters (e.g., "&") and noise words (e.g., "the").	Pass	Verify NLP is robust to minor data imperfections.
1.4	Handle empty or minimal user description content (negative test)	System identifies the lack of sufficient content for analysis.	Pass	Verify the system prompts the user for more input if necessary.
Keyword Profile Generation:				
2.1	Generate keyword profile for user description based on extracted keywords (positive test)	A profile containing extracted keywords is created for the user description.	Pass	Verify keyword profiles capture the user's interests.
2.2	Generate keyword profile for a skill based on its description (positive test)	A profile containing relevant keywords is created for each skill.	Pass	Verify skill profiles represent the core aspects of each skill.
2.3	Ensure keyword profiles capture relevant information (positive test)	Keyword profiles should reflect the user's intent or the core functionalities of a skill.	Pass	Verify keyword profiles are meaningful for recommendation purposes.
Word Embedding Similarity Score:				
3.1	Calculate similarity score between user description and skill keyword profile (positive test)	A score is generated indicating how well the user description matches the skill.	Pass	Verify the core functionality of word embedding similarity is working.
3.2	Verify higher similarity scores for descriptions with more overlapping keywords (positive test)	Descriptions with more matching keywords should have higher similarity scores.	Pass	Verify the similarity score reflects the degree of keyword overlap.
3.3	Handle cases with low or no keyword overlap (positive test)	The system should gracefully handle low similarity scores and potentially recommend broader introductory skills.	Pass	Verify the system can adapt to varying levels of keyword match.
Skill Scoring:				
4.1	Calculate skill score based on the number of likes and recommendations (positive test)	A score is generated that considers both the number of times a skill is liked and recommended.	Pass	Verify the like-to-recommendation ratio is factored into the score.
4.2	Verify higher scores for skills with more likes relative to recommendations (positive test)	Skills with more likes compared to recommendations should have a higher score.	Pass	Verify the score prioritizes user preferences indicated by likes.
4.3	Ensure score updates reflect changes in user likes (positive test)	The skill score should update accordingly when a user likes or dislikes a skill.	Pass	Verify the system dynamically reflects user interaction.
K-means Clustering:				
5.1	Cluster users based on their keyword profiles (positive test)	Users with similar keyword profiles are grouped into the same cluster.	Pass	Verify the clustering algorithm effectively groups users by interests.
5.2	Verify similar user profiles are grouped within the same cluster (positive test)	Users who like similar skills or have similar descriptions should be clustered together.	Pass	Verify clusters represent users with shared learning interests.
5.3	Test cluster stability with addition of new user data (positive test)	The addition of new user data should not significantly alter existing cluster configurations.	Pass	Verify the clustering algorithm can adapt to growing datasets.

Table 6.3: Recommendation sub-system test plan.

6.1.3 Integration Testing

Table 6.4 shows the integration testing plan. Integration tests verify that different modules of the system work together as intended. For this purpose, Postman is employed to simulate and test API calls [42], ensuring seamless interaction between the frontend, backend, and recommendation sub-systems.

Test Case ID	Description	Expected Result	Pass/Fail	Comments
API Development and Documentation				
1.1	Develop and document backend API endpoints for user authentication, data access, and recommendation requests.	Backend API exposes documented endpoints for user interaction and data exchange.	Pass	Verify all necessary endpoints are implemented and documented clearly.
1.2	Define a clear JSON format for data exchange between backend and recommendation system APIs.	Agreed-upon JSON format is documented for consistent data transfer.	Pass	Verify the format includes user ID, description, and skill data for recommendations.
Data Exchange and Error Handling				
2.1	Test successful user authentication and retrieval of user data (search history, liked skills) from backend API.	Frontend receives user data in the expected JSON format.	Pass	Verify data retrieval aligns with documented API behavior.
2.2	Test sending user description to recommendation system API and receive recommended skills data.	Backend API successfully transmits data and receives recommendations from the recommendation system.	Pass	Verify data exchange adheres to the defined JSON format.
2.3	Simulate API communication failures (backend to recommendation system) and verify frontend error handling.	Frontend gracefully displays user-friendly messages indicating issues with recommendations.	Pass	Verify the system can handle external API unavailability.
2.4	Test backend API error handling for invalid user credentials or data requests from the frontend.	Backend API returns informative error messages for invalid user interactions.	Pass	Verify error messages help developers identify and address frontend issues.
Security Considerations				
3.1	Verify secure user authentication and authorization mechanisms in the backend API.	Unauthorized access attempts are rejected, and user data is protected.	Pass	Security measures should comply with industry standards.
3.2	Confirm HTTPS usage for encrypted communication between all systems (frontend, backend, recommendation system).	Encrypted communication channels are established for secure data transfer.	Pass	Verify no sensitive data is transmitted in plain text.
3.3	Encrypted User Data	Hash user password before being sent to backend.		Verify no sensitive data is transmitted in plain text.
Testing and Deployment				
4.1	Conduct unit tests for each system (frontend, backend, recommendation system) to ensure individual functionality.	Each system operates independently as expected.	Pass	Unit testing focuses on the internal logic and functionalities of each system.
4.2	Perform integration tests to verify seamless communication and data exchange between all systems.	Systems work together to deliver recommendations based on user input.	Pass	This ensures smooth interaction between frontend, backend, and recommendation system APIs.
4.3	Conduct user acceptance testing (UAT) with real users to evaluate system usability and gather feedback.	Users can navigate the system easily and find the recommendations helpful.	Pass	UAT assesses the overall user experience and identifies areas for improvement.
4.4	Coordinate deployment of all systems (frontend, backend, recommendation system) in a controlled manner.	Integrated systems are launched successfully and function as intended.	Pass	This ensures a smooth transition from development to production environment.
Communication and Maintenance				
5.1	Establish communication channels between developers working on each system for ongoing collaboration.	Developers can effectively communicate and resolve integration challenges.	Pass	Regular meetings and information sharing are crucial for maintaining a cohesive system.
5.2	Maintain documentation for APIs, data formats, and integration procedures for future reference.	Up-to-date documentation facilitates system maintenance and future development.	Pass	Clear documentation reduces confusion and streamlines troubleshooting processes.
5.3	Implement a monitoring system to track and identify potential integration issues.	System performance and potential errors are detected for proactive maintenance.	Pass	Monitoring allows for early detection and resolution of integration problems.

Table 6.4: System Integration test plan.

This comprehensive testing ensures the system meets its design specifications and user expectations, enhancing reliability and performance.

6.2 User Acceptance Testing

The User Acceptance Testing (UAT) stage is a critical phase in software development, where the target audience evaluates the system. The primary goal of UAT is to verify whether the system aligns with user expectations and to identify any issues or specific scenarios not found during the previous testing phase.

Figure 6.1 shows a usability event, which served as the setting for a lab designed to gather user feedback on the system's interface design and the relevancy of its recommendations. Participants engaged with the system and subsequently provided their insights by completing a survey. This method is a common user experience data collection method, focusing on collecting particular data from a smaller pool of participants [22].



Figure 6.1: Usability Event Lab setup.

The UAT aims to test if the system meets two of the objectives set out in Section 3:

1. The system is **usable** (Objective 1)
2. The system provides **relevant results**. (Objective 2)

6.2.1 Survey Design

A survey was selected as the data collection method for this lab because it allows participants to complete it independently and at their convenience after testing the system. This flexibility leads to higher participation rates and reduces the overall time required to complete the lab. It is important to ensure that the duration of the survey is not too long, as fewer people will be inclined to complete the study [22]. The questions are designed following the Technology Acceptance Model (TAM), a set of predetermined usability questions, focusing on how easy the participants found the system to use and how the participants felt using the system [43]. To ensure that participants completed the survey properly, the ordering of both the questions and responses were random. The bias of the questions was also changed, to ensure that participants were not just choosing the same option for all questions [22].

Skill Boost Usability Survey

Welcome to our brief survey, designed to take just up to 3 minutes of your time.

This survey is for a recommendation system called 'Skill Boost,' which specialises in recommending Computer Science courses, skills, and events. We aim to understand the recommendations you expect and would like to see from the system.

Thank you for your participation! By completing and submitting your answers, you agree to the anonymous use of your data for evaluation purposes.

* Indicates required question

1. What Description did you enter *

2. How did you feel when you were using the system *

Mark only one oval.

- Relaxed
- Neutral
- Frustrated

3. To what extent do you agree with the following: The Skill Boost System was easy to use *

Mark only one oval.

- Strongly disagree
- Disagree
- Neutral
- Agree
- Strongly Agree

4. To what extent do you agree with the following: The system suggested irrelevant options. *

Mark only one oval.

- Strongly disagree
- Disagree
- Neutral
- Agree
- Strongly Agree

5. How useful is the system to help you find courses, events and software to improve your skills and understanding of Computer Science *

Mark only one oval.

- Extremely Useful
- Fairly Useful
- Neither Useful nor Un-useful
- Fairly Un-useful
- Extremely Un-useful

6. Are there any other improvements or suggestions you would like to make to improve the system?

Figure 6.2: Survey given to lab participants.

Figure 6.2 shows the survey design, which was given to participants after they had completed the lab. The design of each question is described as follows:

1. **“What description did you enter?”**: Participants can include the search criteria that they entered into the system. This is essential as it is important to see if certain topics influence the quality of the response of the system. This also indicates any topics that the system could not produce any results for.
2. **“How did you feel when you were using the system?”**: This question focuses on the system usability, providing insights into how easy the system was to use and navigate. If participants felt “Relaxed” or “Neutral”, this implies they found the system easy to use. Whereas the participant feeling “Frustrated” suggests the system is not meeting their expectations [22].
3. **“To what extent do you agree with the following: The Skill Boost System was easy to use”**: This is a TAM question, focusing on the “Perceived ease of use” of the system, providing insight to the usability of the system [43]. High scores indicate that users find the interface intuitive, interactions straightforward, and the overall experience user-friendly. Conversely, lower scores can highlight areas where the system might be confusing or difficult to navigate, signalling the need for design improvements [43].
4. **“To what extent do you agree with the following: The system suggested irrelevant options.”**: This question focuses on the relevant results aim of the lab, providing insight into what the user believes is relevant.
5. **“How useful is the system to help you find courses, events and software to improve your skills and understanding of Computer Science”**: This question focuses on the relevant results aim of the lab, aiming to understand the relevancy and need of the system as a whole.
6. **“Are there any other improvements or suggestions you would like to make to improve the system?”**: This open-ended question invites other suggestions that can reveal insights that were not anticipated or covered by the survey. This also allows participants to suggest valuable ideas for

new features or improvements.

6.2.2 Analysis of the Results

This section discusses the analysis of the results. 12 participants completed the lab at the usability event, the nature of labs is that they collect detailed information from participants and are very thorough, therefore a pool of 12 is suitable for the context.

Analysis Methodology

The survey was designed for both qualitative and quantitative analysis. Open-ended questions present common themes and opinions of the participants, whereas closed questions can be analysed quantitatively to gauge the opinions of the majority. Google Forms [44] was used to present the survey as well as the results.

Individual Question analysis

Question 1: What description did you enter? Figure 6.3 shows the participant's responses.

Description entries that produced results

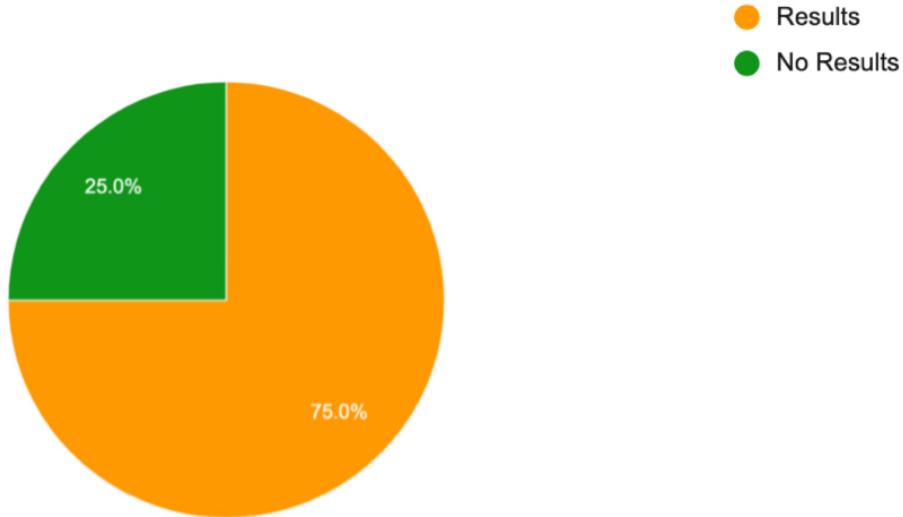


Figure 6.3: Proportion of results that were displayed to the participant from the description entered.

The responses to this question highlight the most common themes that the participants entered. The

individual responses to the question can be seen in Appendix C however, Figure 6.3 shows the proportion of descriptions entered that produced results. This highlights that 25% of the descriptions that the participants entered did not produce any results, this directly links to the NLP dictionary not containing these words. Inputs such as “path-finding graphs” and “k-means clustering” are valid Computer Science terms that should produce results, therefore this highlights a limitation in the dictionary being used by the system. This can also directly impact the overall user satisfaction with using the system, which may reflect in the user’s responses to the other questions in the survey.

Question 2: How did you feel when you were using the system? Figure 6.4 shows the participant’s responses.

How did you feel when you were using the system

12 responses

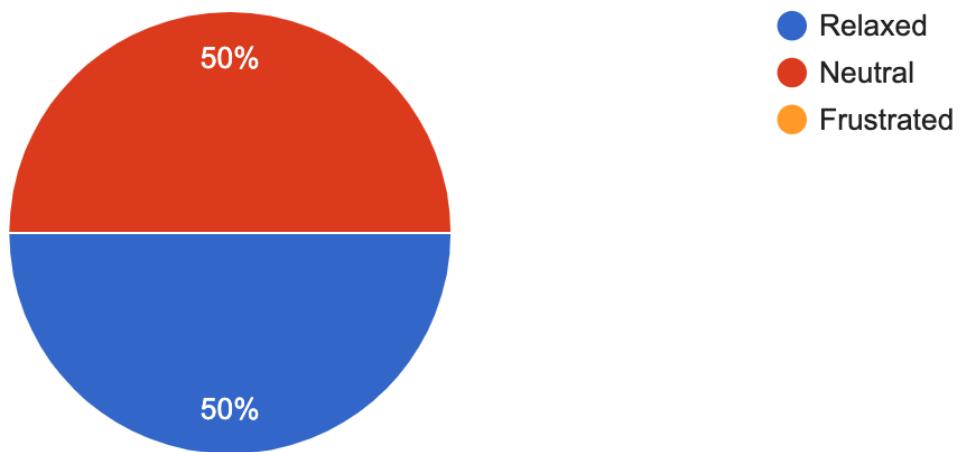


Figure 6.4: Survey responses for Question 2.

The absence of “frustrated” responses suggests that the system is well-conceived in terms of usability, avoiding causing significant stress or confusion amongst users. This aspect is crucial in user experience, as a system that steers clear of inducing frustration is more likely to be utilised consistently and recommended to others [22].

The “relaxed” responses indicate that for half of the users, the system not only meets basic usability standards but also provides a comfortable and stress-free interaction. This is an encouraging indication that the system’s design and workflow potentially contribute to a positive emotional state, which can enhance user satisfaction and engagement [22].

However, the interpretation of the “neutral” responses merits further analysis. While at first glance, it might appear to suggest a lack of strong positive engagement, it’s important to recognise that “neutral” can also be viewed as a positive, valid response. In this context, a neutral emotional state could signify that the system efficiently facilitates task completion without eliciting negative emotions. Hence, “neutral” could be indicative of a system that meets users’ needs and expectations, fulfilling its intended purpose effectively.

In conclusion, the reported user emotions towards the system present a largely positive picture, with no participants experiencing frustration and half of the participants feeling relaxed during use.

Question 3: To what extent do you agree with the following: The Skill Boost System was easy to use. Figure 6.5 shows the participant’s responses.

To what extent do you agree with the following: The Skill Boost System was easy to use

12 responses

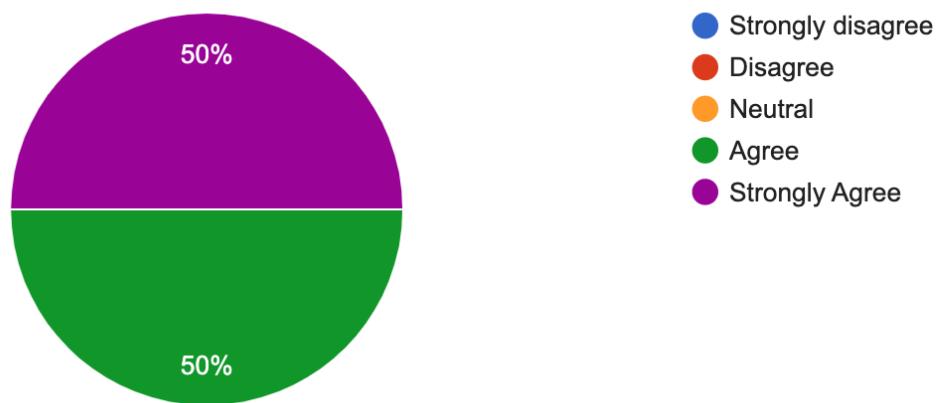


Figure 6.5: Survey responses for Question 3.

The feedback was uniformly positive, with 50% of participants expressing strong agreement and the remaining 50% agreeing that the system was easy to use. This unanimous approval underscores the system's successful design in facilitating an intuitive user experience, effectively meeting participants' needs.

The lack of negative or even neutral responses is particularly telling, indicating that the system's design potentially exceeds participants' expectations for ease of use. However, the variation in the strength of agreement—between those who "Agreed" and those who "Strongly Agreed" may reflect differences in user experiences, particularly regarding the system's ability to deliver relevant results. As described in the analysis of question 1, participants who did not receive results for their queries might naturally find the system less easy to use, even if this was not explicitly stated in their responses to Question 3.

Question 4: To what extent do you agree with the following: The system suggested irrelevant options. Figure 6.6 shows the participant's responses.

To what extent do you agree with the following: The system suggested irrelevant options.

12 responses

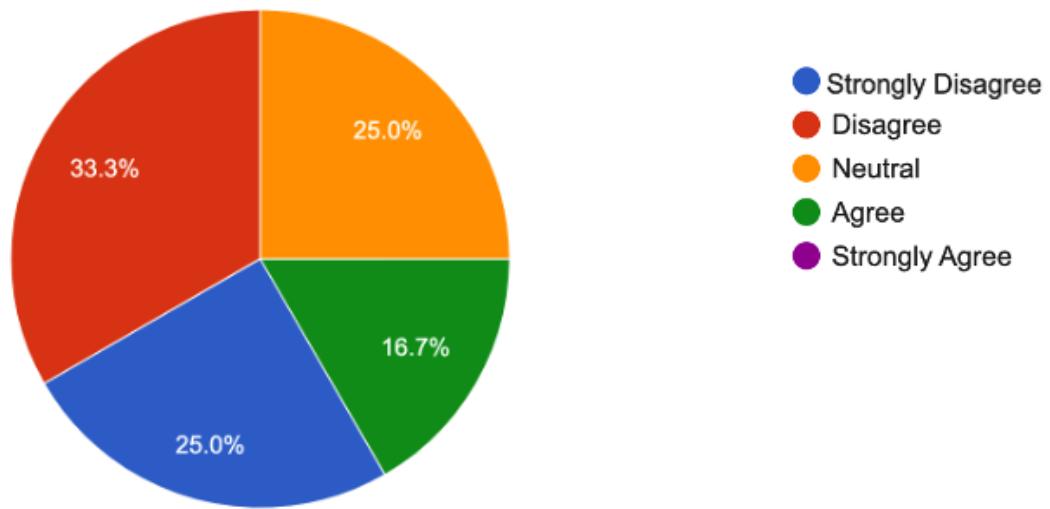


Figure 6.6: Survey responses for Question 4.

The responses reveal nuanced insights into the system's current performance and user expectations. Specifically, 58% of participants viewed the recommendations as relevant, which indicates a majority

satisfaction with the system's ability to provide meaningful suggestions. Nonetheless, the presence of 16.7% of users finding recommendations irrelevant and 25% remaining neutral on this matter suggests there is considerable room for improvement in how the system interprets and responds to user needs.

A significant factor to consider in understanding the perception of irrelevance is the database's limitations. It's plausible that the content or options users deem most relevant simply do not exist within the system's database. This limitation inherently restricts the system's ability to generate ideal recommendations, potentially leading to user experiences that range from finding recommendations mildly off-target (reflected in neutral responses) to entirely irrelevant. This gap between user expectations and the system's deliverables underscores the need for expanding the database or improving how the system identifies and matches user preferences with available options.

Furthermore, users who received no recommendations at all represent a unique subset of the feedback. Their responses are likely to skew towards finding the system's recommendations irrelevant or unhelpful, as the lack of results directly impacts their ability to achieve desired outcomes or discover valuable content. This scenario could significantly impact overall satisfaction and trust in the system's utility.

Question 5: How useful is the system to help you find courses, events and software to improve your skills and understanding of Computer Science Figure 6.7 shows the participant's responses.

How useful is the system to help you find courses, events and software to improve your skills and understanding of Computer Science

12 responses

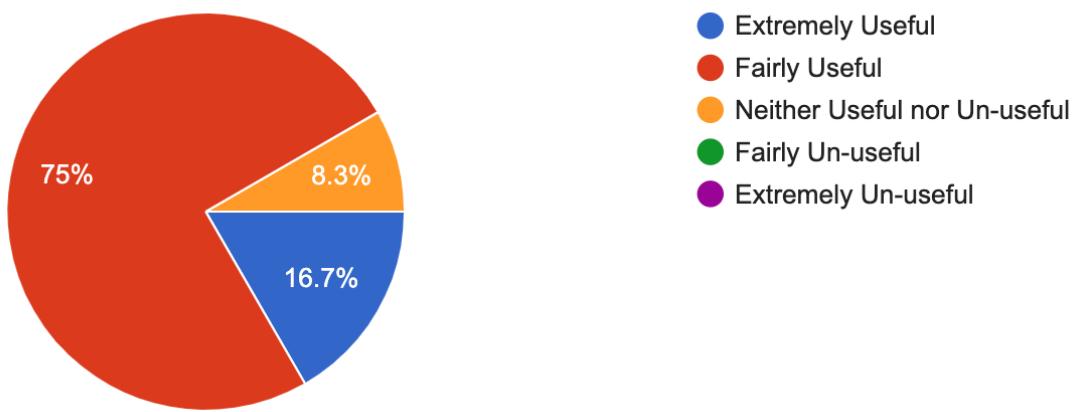


Figure 6.7: Survey responses for Question 5.

The feedback underscores a highly positive reception towards the system's ability to facilitate users in finding courses, events, and software that enhance their skills and understanding of Computer Science. With 75% of respondents acknowledging the system as fairly useful and an additional 16.7% viewing it as extremely useful, it is evident that the system substantially meets, if not exceeds, user expectations in delivering relevant recommendations.

The 8.3% of users who provided a neutral response, considering the system neither useful nor un-useful, could be attributed to instances where the system did not generate any recommendations for their specific queries. This perspective offers invaluable insight rather than a critique, highlighting a unique opportunity to refine the system's responsiveness.

No participants rated the system as un-useful or extremely un-useful, a testament to the system's effectiveness and the quality of its recommendations. This unanimous positive feedback reinforces the system's ability to recommend relevant Computer Science skills. This is a strong affirmation of the system's alignment with user needs and its role as a crucial tool in their learning.

Question 6: Are there any other improvements or suggestions you would like to make to improve the system? This optional, open question allowed the participants to comment on any other opinions or suggestions they had about the system, that the more structured questions did not cover. The common themes amongst the responses were focusing on the UI and data. Participants that commented on the UI, stated that it “looks good” and that the system was “Clear and easy to use”. There were also some suggestions made, participants suggested that “sample prompts” should be used to make it easier to know what to enter into the system. Moreover, the “Read more functionality was not implemented” and they wanted functionality that made it “More user-specific”. Participants that commented on the data available, stated that they wanted “more topics”, “more data” and “more software options” as that was their “preferred way to learn”.

The feedback on the UI demonstrates that the system appeals to the target audience and meets Objective 1 of creating an MLP. The constructive comments regarding the UI mainly focus on aspects of the system that were not in the scope of the lab. The “Read More” functionality had been disabled to ensure that the participants were focusing on the result options and ordering, rather than a particular skill. Additionally, a user-specific home page existed in the system but was not shown as the focus was on the search and the results.

The feedback on the data demonstrates the limitations of the databases used for the system. The suggestion of adding more topics stems from the dictionary limitations discussed in the analysis of Question 1, with searches returning no results due to the limited scope of words in the dictionaries. Furthermore, participants asking for more data and software is a limitation of the skills dataset.

Overall, this question demonstrated users’ satisfaction with the system, as well as highlighting some areas of improvement that can be implemented in the future.

6.2.3 Summary

Overall, the UAT lab has provided significant insights into the system’s design quality and the relevance of its recommendations. Participants offered detailed feedback on the system’s usability, particularly emphasising the design’s quality and providing constructive suggestions for enhancement. The relevance

of the recommendations was evaluated, with users having a positive experience with the system. Despite the relatively small participant pool, the twelve labs conducted were thorough, aligning with the primary objective of a UAT lab to furnish in-depth insights rather than a broad overview. In summary, the findings from this survey affirm that the system achieves its intended objectives by being user-friendly and delivering relevant recommendations.

6.3 Technical Survey

A technical survey was conducted to evaluate the performance of the recommendation system, with a particular focus on its ability to deliver "relevant results to the user." In this context, "relevancy" refers to how closely the results match the user's search terms. Testing for relevancy is inherently challenging due to the diversity and complexity of potential search scenarios. Simulating user behaviour accurately is difficult, making the insights from actual users more valuable [45]. Therefore, this evaluation strategy includes gathering feedback from computer science experts to assess whether the recommendation system's outputs meet the expectations of professionals in the field. This approach aims to provide a robust assessment of the system's effectiveness in delivering pertinent recommendations. This technical survey aims to understand two characteristics of the recommendation system.

1. The results are in the **correct order**, ensuring that the most relevant outcomes appear first.
2. The results are **not trivial**, meaning the system is not recommending based on the number of keyword matches.

Due to the length of the survey, it has been included in Appendix E. However, the questions used are included in the results for clarity.

6.3.1 Design and Methodology

The decision to employ a survey method was driven by the fact surveys are accessible, straightforward to distribute, and offer flexibility for respondents to complete at their convenience [22]. The target demographic for this survey was individuals with a background in computer science. This specific group was

chosen due to their expertise, which is crucial for accurately assessing the relevancy of the results provided by the system.

The survey incorporated two principal types of questions: those requiring participants to order options according to their relevance, and those allowing for selection from a set of multiple-choice options. Given that questions requiring ordering are typically more time-consuming than multiple-choice questions [23], a deliberate decision was made to limit the number of ordering questions to ensure the survey remained engaging and manageable for participants.

Furthermore, it was imperative to design the survey to be concise to avoid deterring participants; extensive surveys are known to reduce completion rates significantly [22]. To mitigate any potential biases in response, both the order of the options presented within each question and the sequence of the questions themselves were randomised. This approach was adopted to eliminate any influence the arrangement of questions or options might have on the participants' answers, ensuring the collection of unbiased and genuine feedback.

Designing Questions to Test the Ordering of Results

To ensure the survey questions on result ordering were both comprehensive and reflective of the Skills dataset's diversity, a *K*-means clustering analysis was conducted across the entire dataset. The objective was to identify common themes or groups within computer science topics, thereby facilitating the formulation of survey questions that were representative of the dataset yet concise enough to prevent overwhelming respondents.

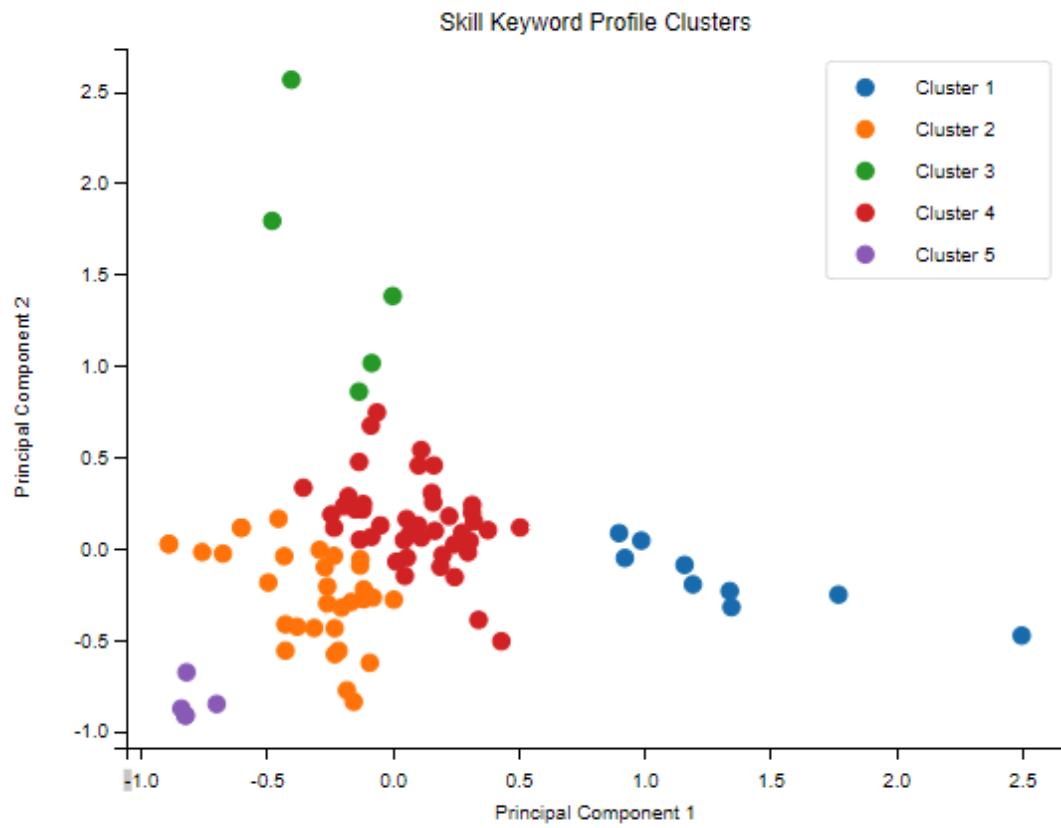


Figure 6.8: K-means clustering on the Skills database with $k=5$

The process involved experimenting with various values for K (namely, 3, 4, 5, 6, and 7) to determine the most appropriate number of clusters that could accurately represent the dataset's diversity without introducing complexity, the results can be seen in Appendix D. Figure 6.8 shows the graph for $K=5$, which emerged as the optimal choice, offering a balanced representation that captures distinct computer science themes within the data. This determination was based on achieving a level of granularity that highlighted clear, meaningful clusters without either oversimplification or unnecessary complexity. For each cluster identified using $K = 5$, the most common two keywords within that cluster were selected as the basis for generating survey questions.

From the five clusters identified, the three largest clusters (1, 2 and 4) were selected for further analysis. The most common keyword within each of these clusters was identified as 'Web Development', 'Cyber Security', and 'Machine Learning' where each keyword was entered into the system and results recorded.

The designed survey questions aimed to present participants with the top three results generated by the system for each identified keyword. Participants were then asked to rank these results based on their relevance. This method intended to directly compare the system's ranking of skills with the participants' perceived ordering. Such a comparison provides valuable insights into the system's effectiveness in prioritising results that align with user expectations and needs.

Designing Questions to Test for Trivial Results

In this section, the K -means clustering graph was again utilized to inform the survey's design, specifically aiming to verify that the system does not yield trivial results. Trivial results, in this context, are defined as recommendations made solely based on the quantity of keyword matches, rather than the quality or relevance of those matches. To assess this, the two most common keywords from each cluster were selected as the search criteria. For each set of keywords, the options chosen were three skills that included at least one (or both) of the keywords and two additional recommendations that did not contain any of the entered keywords.

Participants were then presented with these five results and asked to identify the three they believed were most relevant to the initially entered keywords. For instance, 'artificial intelligence' and 'machine learning' were the keywords representing one of the clusters, and five skill recommendations would be chosen based on these search terms. Users were then tasked with selecting the three out of five results that they felt most accurately aligned with the concepts of artificial intelligence and machine learning.

6.3.2 Analysis of Results

The survey was completed by 11 people, this pool of participants is small however this is due to the time restrictions of the survey window as well as the difficulty of the survey, with many participants commenting that it was "Tricky" and they "had to think quite hard about some of the options".

Ordering of Results

The participant's results can be directly compared to the results of the system. If the results match then the system produces the correct ordering of results. Each question will be analysed to see if this is the

case and then an overall comparison will be made to determine if all results are in the correct order.

Individual Analysis

Question 1 results: Figure 6.9 shows the results from the participants.

Please rank the following options from most to least relevant to: Web Development

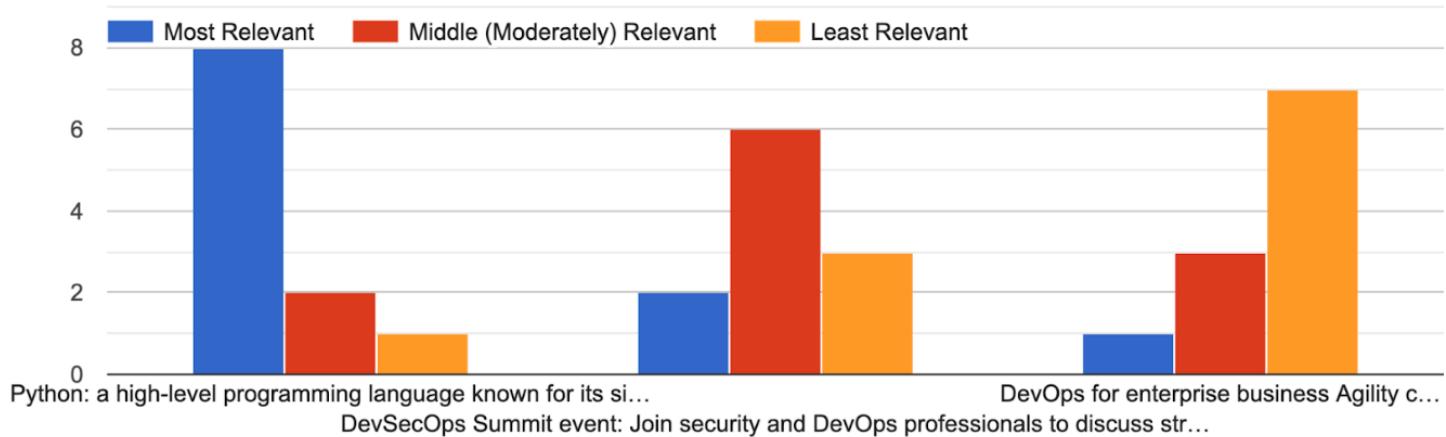


Figure 6.9: Question 1: Ranking options most relevant to “Web Development”

The results produced by the system when using the search term “Web Development” were:

1. Python: Most relevant
2. DevSecOps: Moderately relevant
3. DevOps: Least relevant

Comparing the results from the system and the responses from the survey shows that the results from both were the same, with the majority agreeing with the results that the system produced.

Question 2 results: Figure 6.10 shows the results from the participants.

Please rank the following options from most to least relevant to: Databases

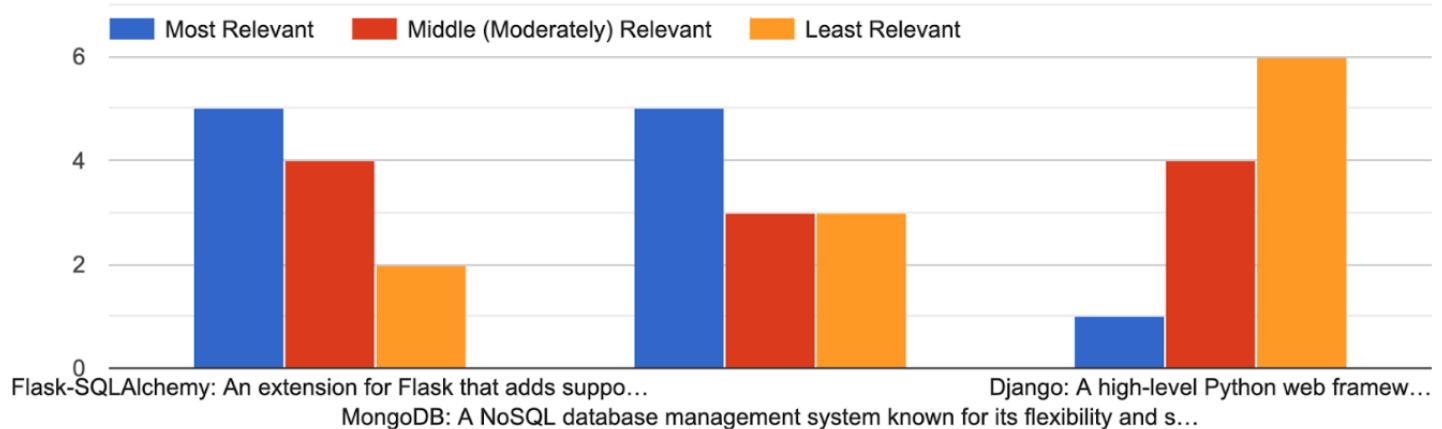


Figure 6.10: Question 2: Ranking options most relevant to “Databases”

The results produced by the system when using the search term “Databases” were:

1. [Flask-SQLAlchemy: Most relevant](#)
2. [MongoDB: Moderately relevant](#)
3. [Django: Least relevant](#)

Comparing the results from the system and the survey, the participants and system identified the same most and least relevant options, suggesting that the ordering is correct even though the conclusions on the moderately relevant skills were different.

Question 3 results: Figure 6.11 shows the results from the participants.

Please rank the following options from most to least relevant to: Artificial Intelligence

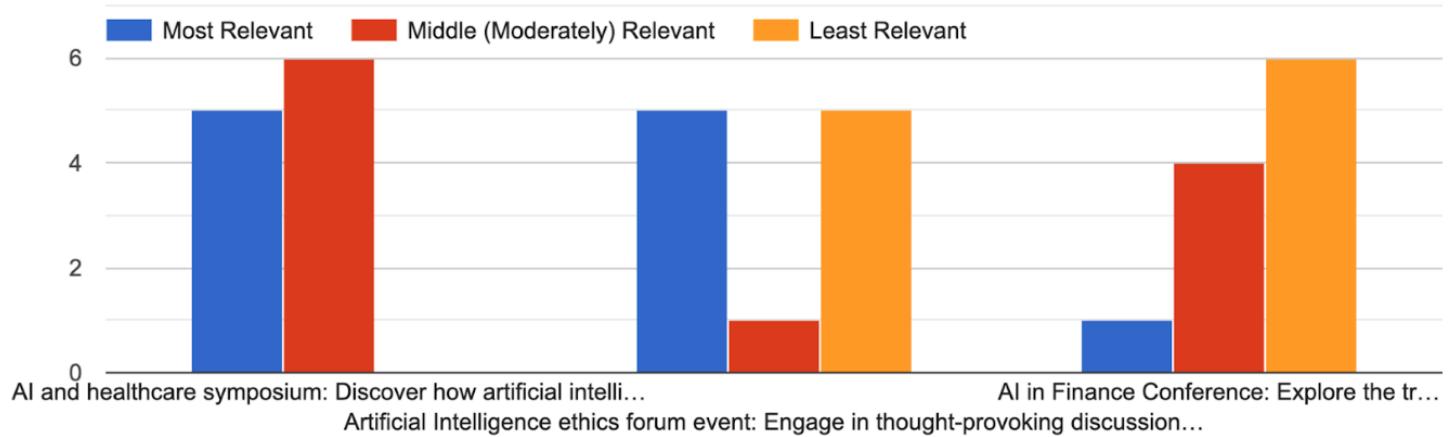


Figure 6.11: Question 3: Ranking options most relevant to “Artificial Intelligence”

The results produced by the system when using the search term “Artificial Intelligence” were:

1. [AI and healthcare symposium: Most relevant](#)
2. [Artificial Intelligence ethics forum event: Moderately relevant](#)
3. [AI in Finance Conference: Least relevant](#)

Comparing the results from the system and survey, the system and participants slightly disagreed with the ordering. Both the system and the participants agreed on the least relevant skill, however they disagreed on the most relevant. Although there is a disagreement, the results are still close, therefore suggesting the ordering is still valid.

Conclusion The analysis shows that overall the rankings of skills produced by the system align with the expectations of expert users. While the participant pool was limited, their feedback underscores the system’s capability to prioritise content effectively, mirroring user expectations in terms of relevance.

Trivial Results

Individual Analysis For each of the following figures, the top 3 options contain the keyword in their profile. Figures 6.12, 6.13, 6.14, 6.15 and 6.16 show the results of the survey.

Please select the 3 options you believe are most relevant to learning about: Artificial Intelligence and Machine Learning

11 responses

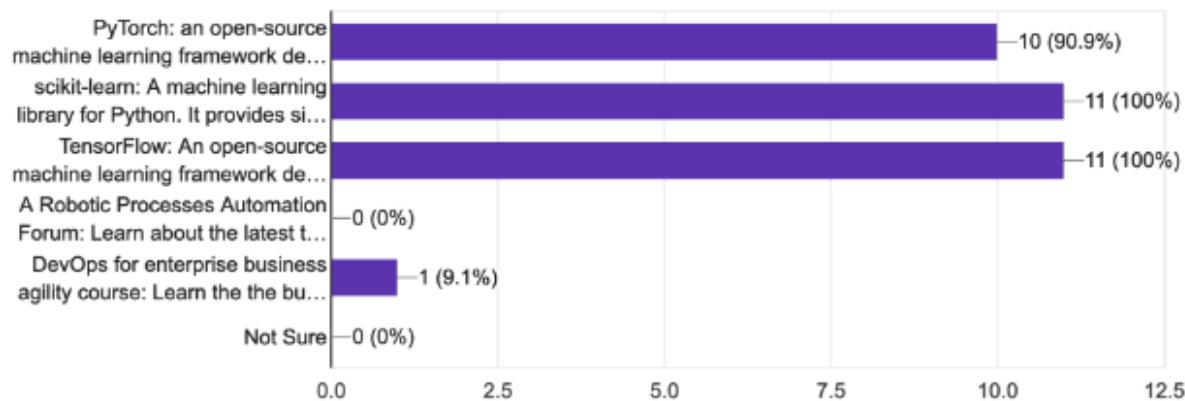


Figure 6.12: Question 4: Choosing the most relevant skills related to “Artificial Intelligence and Machine Learning”

Please select the 3 options you believe are most relevant to learning about: User Interface and Web Development

11 responses

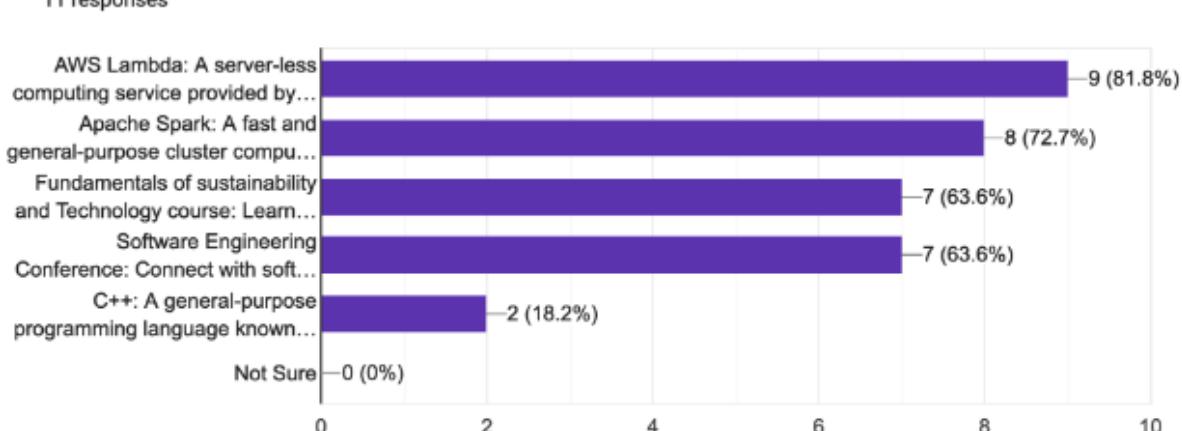


Figure 6.13: Question 5: Choosing the most relevant skills related to “Data Science and Big Data”

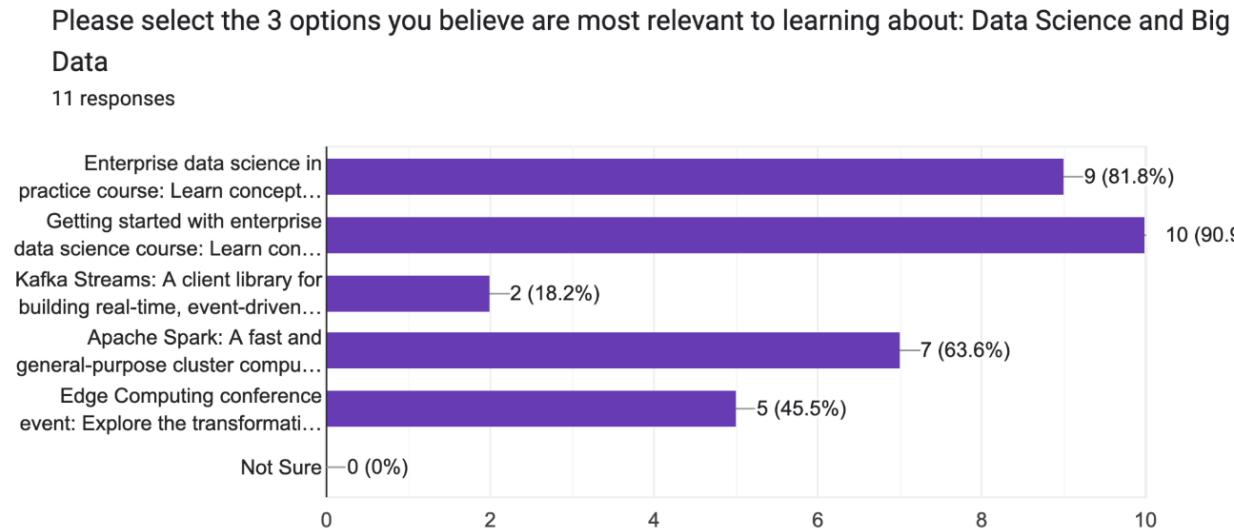


Figure 6.14: Question 6: Choosing the most relevant skills related to “User Interface and Web Development”

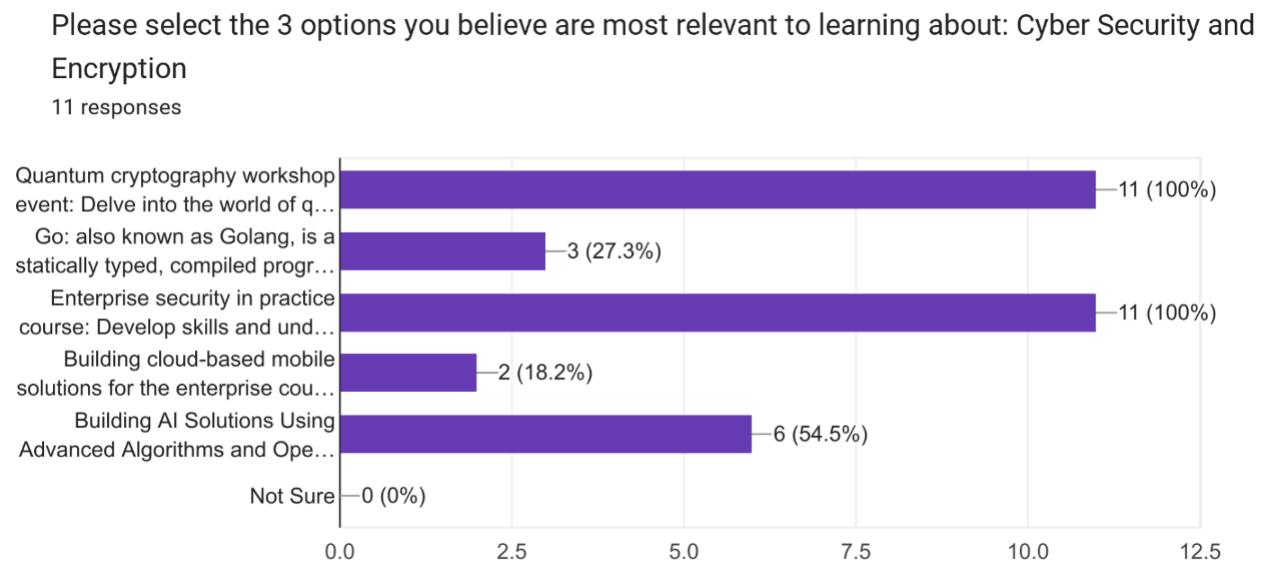


Figure 6.15: Question 7: Choosing the most relevant skills related to “Cyber Security and Encryption”

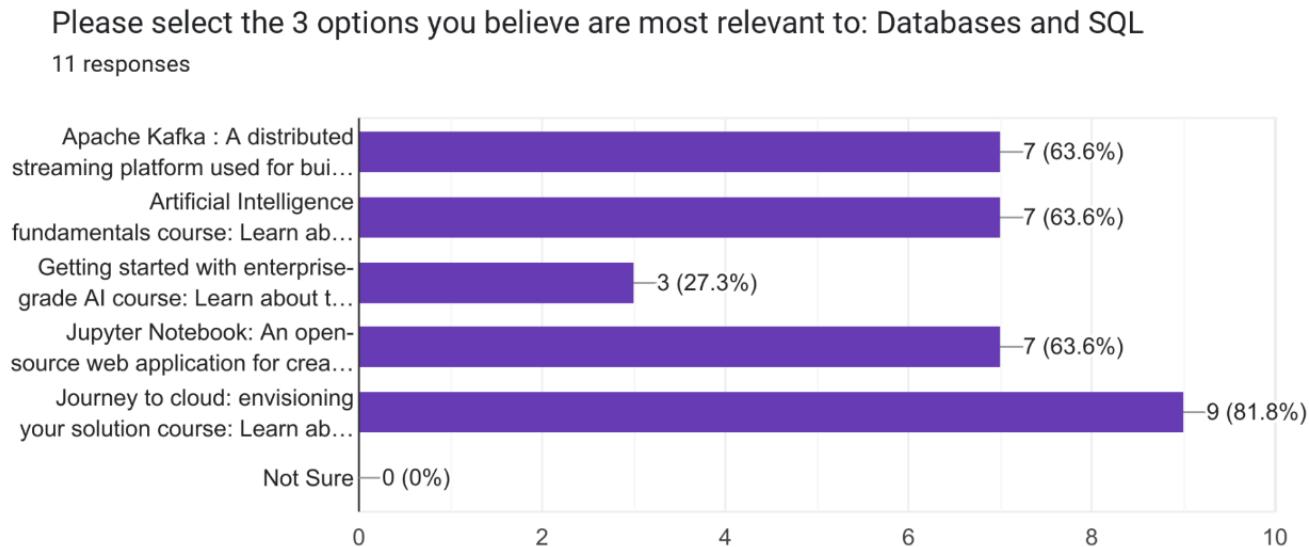


Figure 6.16: Question 8: Choosing the most relevant skills related to “Databases and SQL”

Summary The analysis of user responses to the five survey questions revealed that the recommendation system produces relevant recommendations. Despite the top choices often containing the keyword within their profiles, participants consistently identified options lacking the direct keyword as relevant. This observation highlights the system’s capability to extend beyond keyword matching, highlighting its understanding of related concepts and its ability to match them with user queries effectively.

6.4 Expert Evaluation

6.4.1 Overview

The objective of this evaluation was to conduct an expert walkthrough of the system with a specialist well-versed in the project’s goals and ambitions. The primary aim was to secure an expert’s perspective on the system’s design quality, and the efficiency of its recommendation mechanism, and to determine if the project’s initial objectives have been met.

The expert for this task was a Product Manager at IBM, who has provided mentorship throughout the project’s duration. With their extensive experience in designing and implementing a variety of systems, coupled with an industry-centric viewpoint, this expert’s feedback is especially valuable. Their close

tracking of the project guarantees a comprehensive understanding of its fundamental objectives, aims, and desired outcomes.

6.4.2 Methodology

The evaluation was a walkthrough, showcasing the system's key functionalities, accompanied by specific questions and areas for discussion. This informal approach allowed the expert to be introduced to the system incrementally, facilitating targeted discussions or commentary on different segments of its design and operation.

6.4.3 Summary of the Interview

The interview provided substantial feedback on the project's success, which is invaluable for system evaluation for several reasons:

- The project was observed to have gone "above and beyond" the initial core objectives, indicating not just the achievement but the surpassing of predefined goals. This underscores the system's comprehensive effectiveness and sets a new benchmark for future developments.
- The system's "professional" design, highlighted by the expert, reflects a robust and user-centred development approach. This is crucial as it directly impacts user engagement and satisfaction, key indicators of a system's usability and market viability.
- The expert's commendation of the system's architecture and "good data structure" points to a solid foundational setup enabling optimal performance. This aspect is vital for evaluating the system's efficiency in processing and delivering recommendations, ensuring scalability and reliability.
- The inclusion of "different granularity" in the recommendations, such as "including the ranked data" alongside suggestions for similar users, was identified as fulfilling all set objectives. This demonstrates the system's ability to provide nuanced and tailored recommendations, enhancing user experience by catering to diverse user preferences and needs.
- The system is functioning "as an MLP as it is designed from a product point of view" was particularly

emphasised. This approval from a product management perspective is essential for evaluating the system's overall market readiness and its potential to meet user demands effectively.

The expert's insights, therefore, affirm the system's success in meeting its original objectives and also provide a detailed evaluation of its design, functionality, and market potential. These aspects are crucial for understanding if the system has met its initial objectives.

6.5 Summary

In summary, the evaluation chapter explored the effectiveness and reliability of the Skill Boost recommendation system through a comprehensive suite of testing and feedback mechanisms. Unit and integration testing confirmed the technical robustness of the system, ensuring that individual components function correctly both in isolation and when integrated. User acceptance testing provided crucial insights into the system's usability and alignment with user expectations, revealing high levels of satisfaction and intuitive interaction. The technical survey extended this evaluation by gathering detailed user feedback on the system's recommendations, further affirming the relevance and accuracy of its outputs. Lastly, expert evaluations offered a professional perspective on the system's design and implementation, endorsing its conceptual soundness and suggesting areas for further enhancement.

Chapter 7

Project Management

This chapter assesses the management of the project. Following the evaluation, it is important to discuss and evaluate other elements of the project. This chapter will include a risk management reflection, ethical and legal considerations taken during the project, limitations discussion and a personal reflection on the project's completion, achievements and management.

7.1 Risk Management Reflection

As discussed in Section 4.3, this project incorporated several strategies to mitigate potential risks. Reflecting on the project's completion, it's valuable to evaluate the effectiveness of these mitigations.

- **Iterative Approach:** Adopting Agile methodology facilitated an iterative development process. Starting with a Minimum Viable Product (MVP), the project evolved through successive iterations that each introduced additional functionality. This ensured a consistently operational system at each stage and allowed for the possibility of reverting to a previous iteration in the event of a critical issue, thereby safeguarding the project against major setbacks.
- **Version Control:** The use of GitHub for version control was instrumental in managing the development of three distinct systems across different environments. Daily updates, complete with descriptive commits, enhanced the visibility of development progress and facilitated easier bug tracking and rollback when necessary.
- **Dummy Data:** Faced with the unavailability of IBM Skills Build Data, the project pivoted to the use of dummy data for Software Skills and Event Skills recommendations. ChatGPT was employed to generate this dummy dataset, a decision that proved crucial in maintaining project momentum in light of data access restrictions. The transcript data is detailed in Appendix F and Appendix G.

These mitigations protected the project from potential data loss and established clear protocols for addressing issues as they arose, contributing significantly to the project's resilience and success. Reflecting

on the entire process, it is evident that while these strategies effectively mitigated many risks, there is always room for improvement in anticipating and preparing for unforeseen challenges. This experience has underscored the importance of flexibility and adaptability in project management, particularly in the context of rapidly evolving project requirements and external constraints.

7.2 Legal and Ethical Considerations

This project navigates a range of legal and ethical considerations to ensure responsible and fair use of technology, particularly concerning user data and the operational integrity of the recommendation system.

7.2.1 Bias Mitigation

The ethical obligation to prevent bias in recommendations is critical. The system is developed with algorithms that aim to be neutral and fair, actively avoiding discrimination based on gender, race, or other protected characteristics. To ensure this is avoided, user registration only requires a username, name and email. Likewise, any survey completions were anonymous with no questions asking for sensitive information.

7.2.2 Generative AI Tools

Ensuring that generative AI tools, such as ChatGPT, were used ethically and in accordance with the department's policy on the use of Generative AI Tools (GAITs). Generative AI was used to generate fake data which was used to enable the development of additional software features which required data that was not accessible, details of which are included in Appendix F and G.

7.2.3 User Consent

Transparency in data collection and use is ensured through clear user communications. Users are informed about the data collection process and its purpose. User consent was obtained for surveys, before the survey was completed users were presented with information on the survey, the data being collected and the following statement: "By completing the survey, you consent to your responses being analysed anonymously."

7.2.4 Data Security

Data security practices are implemented to protect against unauthorised access and data breaches. This includes hashing passwords before transmission and storage, maintaining user trust and ensuring the integrity of personal information.

7.2.5 Accessibility

Accessibility is a core ethical principle, guiding the system's design to be inclusive and usable for people with disabilities. The system was designed in accordance with the Web Content Accessibility Guidelines (WCAG) [10] ensuring that the system is accessible to a broad user base.

7.3 Limitations

This project, while achieving significant milestones and delivering a robust recommendation system, encountered some limitations that influenced its scope and depth. The constraints primarily revolved around data accessibility, time constraints, and the reliance on generative AI for content creation.

7.3.1 Data Accessibility

One of the most substantial limitations faced by the project was the accessibility of real, comprehensive datasets, particularly from IBM Skills Build. This limitation meant data needed to be produced generatively to fill the gaps, especially for the Software Skills and Event Skills components of the recommendation system. While this approach ensured the project's progression, it introduced concerns regarding the authenticity and quality of the data used, potentially affecting the recommendation system's accuracy and reliability.

7.3.2 Time Constraints

The project was bounded by time constraints, which imposed a significant limitation on the extent to which the NLP component and the recommendation system could be refined and improved. Given more time, each of these critical components could have been developed further, enhancing the overall effectiveness and user experience of the recommendation system. The decision to focus on one aspect over the other

was dictated by these time constraints, leading to a trade-off between improving the NLP's sophistication and enhancing the recommendation algorithms.

7.3.3 Reliance on Generative AI

The utilisation of ChatGPT for generating dummy data for the Software Skills and Event Skills presented a creative solution to the data accessibility issue. However, this reliance on generative AI introduced a limitation in terms of the predictability and control over the data quality. While generative AI can produce content that mimics real-world data, the nuances that genuine datasets offer were inherently lacking.

7.3.4 Summary

These limitations, while challenging, also provided learning opportunities and insights into managing projects under constraints. The strategies employed to navigate these issues, such as the adoption of generative AI for data creation and the strategic prioritisation of project components due to time constraints, demonstrate the project's adaptability.

7.4 Project Reflection

This section serves as a personal evaluation and reflection on the achievements and management of the project.

Adherence to the project timeline was essential. The integration of slack time between tasks proved invaluable, particularly when certain tasks required more time than anticipated. Weekly meetings with the project's supervisor ensured the project remained on schedule and facilitated ongoing evaluation of progress. Additionally, fortnightly meetings with an IBM supervisor were instrumental in presenting project advancements and gaining external feedback and advice. These meetings promoted accountability through the establishment of clear, achievable goals for subsequent meetings, thereby contributing to the high standard to which the project was completed. Allocating 12 hours a week of work in four-hour blocks was strategic, ensuring timeliness and mitigating stress associated with the workload.

The project encountered significant challenges, notably regarding data access from IBM Skills Build and the NLP System. Initially, it was assumed that being assigned an IBM supervisor would facilitate

access to necessary data. However, this expectation was unmet, which required the generation and use of dummy data instead of the IBM Skills Build Data. Fortunately, this risk had been anticipated, with mitigation strategies already in place, allowing for a swift and effective adaptation to generative AI for data production.

An additional challenge involved sourcing a suitable dictionary for the NLP System. Extensive research into finding an appropriate Computer Science dictionary revealed that the viable options required purchasing. As a result, Wikipedia's online glossary for computer science was utilised, requiring further processing to extract the required data. Despite being an unforeseen challenge, the extensive research conducted ensured a deep understanding of the necessary data, facilitating the identification of alternative data sources with a strict set of requirements.

In summary, the project, despite its challenges, was a resounding success. The thorough planning and organisational efforts laid a solid foundation for smooth development. When obstacles were encountered, the pre-established strategies and a deep understanding of the project's requirements enabled swift and efficient resolutions. This experience has highlighted the importance of flexibility, proactive risk management, and the value of clear communication and regular feedback in project management.

7.5 Summary

This chapter revisited the comprehensive risk management strategies employed throughout the project. The discussion extended to legal and ethical considerations, ensuring the project maintained high ethical standards, particularly in data handling and privacy concerns. The discussion on limitations demonstrated how thorough planning mitigated their impact. The personal reflection provided insights into the project's journey, highlighting the importance of adaptability, the value of regular feedback, and the role of meticulous organisation in overcoming challenges and achieving project objectives.

Chapter 8

Discussion

This final chapter outlines potential future enhancements for the project and concludes the report. The aim is to examine areas for further research and improvement as well as provide an overall summary of the project, reflecting on its outcomes and contributions to the field. This chapter aims to offer a comprehensive overview, highlighting the project's achievements and suggesting paths for future work.

8.1 Future Work

This section outlines the potential future directions and expansions that could further enhance the project's utility and applicability.

8.1.1 NLP Context Enhancement

A significant area for improvement is enriching the NLP component to understand context. Currently tailored to identify computer science terms, the NLP system's next evolution could involve understanding nuanced user intents. For instance, the NLP would be able to correctly process the input “I know lots about ai but i want to learn more about Deep Neural networks”, knowing not to look for AI keywords, but deep neural network keywords.

8.1.2 System Integration Applications

The project offers potential for integration with professional networking platforms, such as LinkedIn. By repurposing the system's business logic without its frontend, it could serve as a dynamic tool for recommending skill development opportunities directly based on a user's LinkedIn profile and activities. This approach could also open avenues for promoting paid educational resources or advertisements, tailored to the user's professional interests and existing skill set.

8.1.3 Expansion to Other Universities or Companies

Expanding the system to serve students from various universities represents a natural progression. Customising the recommendation engine to reflect the unique course offerings and educational resources of different institutions could significantly enhance its appeal and utility. This would involve integrating and processing data specific to each university through the NLP system to generate accurate and relevant keyword profiles for their courses.

8.1.4 Transcript Generation for CV Enhancement

Another promising extension is the development of a feature that allows users to generate a detailed transcript of their academic and skill achievements. This transcript, designed for easy exportation, could be a valuable addition to a student's CV, highlighting their educational journey and skill acquisitions in a formalised manner.

8.1.5 Progress and Achievement Tracking

Incorporating functionality for users to track their learning progress and achievements could further enrich the platform. This feature would enable users to monitor their engagement with recommended resources, assess their academic growth, and remain motivated through visible milestones and feedback.

Overall, these proposed directions for future work aim to enhance the project's core functionalities. Each extension reflects a commitment to continuous improvement and adaptation, ensuring the system remains relevant and valuable to its users.

8.2 Conclusion

This report has narrated the journey of developing the Skill Boost System, illustrating each critical step towards realising a product designed to enhance skill acquisition through tailored recommendations. From the outset, establishing the project's motivations and scope was instrumental in defining a precise direction and objective. An extensive review of existing solutions and recommendation system techniques informed the system's objectives and requirements, laying the foundation for the project's success criteria.

A coherent methodology, alongside a well-articulated plan and rigorous risk assessment, underscored the project's organisation, steering it towards its goals. The adoption of Agile methodology facilitated a flexible yet disciplined approach to the design and implementation of the system's components.

The evaluation segment of the report assessed the system's performance against its objectives, demonstrating the efficacy and relevance of the Skill Boost System. Furthermore, reflections on project management practices, along with an appraisal of the project's limitations and an exploration of potential future enhancements, have highlighted both the successes achieved and the areas for growth.

In conclusion, the Skill Boost project stands as a testament to the value of thorough research, strategic planning, and agile execution. While it has successfully met its current objectives, the insights gained and the challenges encountered establish a foundation for future iterations to improve the system.

Bibliography

- [1] Statistica. *Biggest problems faced by EU businesses 2023*. Oct. 2023. URL: <https://www.statista.com/statistics/1446396/biggest-problem-faced-by-eu-businesses/>.
- [2] URL: <https://skillsbuild.org>.
- [3] John McNamara. *Trello*. Project requirements document , Accessed on 10/11/2023. Sept. 2022. URL: <https://trello.com/c/usCMutS0/65-create-an-ai-tool-to-help-students-identify-industry-skills-to-compliment-their-academic-skills>.
- [4] Brian De Haaff. *Lovability : how to build a business that people love and be happy doing it*. Austin, Texas: Greenleaf Book Group Press, 2017, pp. 107–109, 111. ISBN: 9781626344037.
- [5] UCAS. *UCAS*. Accessed on 20/11/2023. 2018. URL: <https://www.ucas.com/>.
- [6] Accessed on 20/11/2023. 2019. URL: <https://www.unifrog.org/>.
- [7] 2019. URL: <https://www.thestudentroom.co.uk>.
- [8] Coursera. *Coursera — Online Courses Credentials by Top Educators. Join for Free*. 2018. URL: <https://www.coursera.org>.
- [9] On Your Schedule — Udemy Online Courses - Learn Anything. *Online Courses - Learn Anything, On Your Schedule — Udemy*. 2015. URL: <https://www.udemy.com>.
- [10] W3C. *Web Content Accessibility Guidelines (WCAG) 2.1*. June 2018. URL: <https://www.w3.org/TR/WCAG21>.
- [11] Deepjyoti Roy and Mala Dutta. “A systematic review and research perspective on recommender systems”. In: *Journal of Big Data* 9.1 (May 2022). DOI: <https://doi.org/10.1186/s40537-022-00592-5>.
- [12] Charu C. Aggarwal. *Content-Based Recommender Systems*. Cham: Springer International Publishing, 2016. Chap. 5, pp. 139–166. ISBN: 978-3-319-29659-3. DOI: [10.1007/978-3-319-29659-3_4](https://doi.org/10.1007/978-3-319-29659-3_4). URL: https://doi.org/10.1007/978-3-319-29659-3_4.
- [13] IBM. *What is Natural Language Processing? — IBM*. Accessed on 18/11/2023. URL: <https://www.ibm.com/topics/natural-language-processing#:~:text=the%20next%20step->.
- [14] Diksha Khurana et al. “Natural Language processing: State of the art, Current Trends and Challenges”. In: *Multimedia Tools and Applications* 82.3 (July 2022), pp. 3713–3744. DOI: <https://doi.org/10.1007/s11042-022-13428-4>. URL: <https://link.springer.com/article/10.1007/s11042-022-13428-4>.
- [15] Chuhan Wu et al. “Named Entity Recognition with Context-Aware Dictionary Knowledge”. In: *Chinese Computational Linguistics*. Ed. by Maosong Sun et al. Cham: Springer International Publishing, 2020, pp. 129–143. ISBN: 978-3-030-63031-7.
- [16] Melania Berbatova. “Overview on NLP Techniques for Content-Based Recommender Systems for Books”. In: Accessed on 01/03/2024. Sept. 2019, pp. 55–61. DOI: [10.26615/issn.2603-2821.2019_009](https://doi.org/10.26615/issn.2603-2821.2019_009).
- [17] Qian Zhang, Jie Lu, and Yaochu Jin. “Artificial intelligence in recommender systems”. In: *Complex Intelligent Systems* 7 (Nov. 2020). Accessed on 10/11/2023. DOI: <https://doi.org/10.1007/s40747-020-00212-w>.

- [18] F.O. Isinkaye, Y.O. Folajimi, and B.A. Ojokoh. “Recommendation systems: Principles, methods and evaluation”. In: *Egyptian Informatics Journal* 16.3 (Nov. 2015), pp. 261–273. DOI: <https://doi.org/10.1016/j.eij.2015.06.005>.
- [19] Ian Sommerville. *Software engineering*. 10th ed. Boston, Mass. Amsterdam Cape Town Pearson Education Limited, 2016, pp. 43–72. ISBN: 9781292096131.
- [20] Chandler Harris. *Microservices vs. monolithic architecture*. URL: <https://www.atlassian.com/microservices/microservices-architecture/microservices-vs-monolith>.
- [21] Jakob Nielsen. “Enhancing the explanatory power of usability heuristics”. In: *Conference Companion on Human Factors in Computing Systems*. CHI ’94. Boston, Massachusetts, USA: Association for Computing Machinery, 1994, p. 210. ISBN: 0897916514. DOI: 10.1145/259963.260333. URL: <https://doi.org/10.1145/259963.260333>.
- [22] Yvonne Rogers, Helen Sharp, and Jennifer Preece. *Interaction Design*. John Wiley Sons, Mar. 2023. ISBN: 9781119901112.
- [23] David Benyon. *Designing user experience a guide to HCI, UX and Interaction Design*. Harlow, England: Pearson, 2019, pp. 238–269. ISBN: 9781292155524.
- [24] Accessed on 13/03/2024. URL: <https://coolors.co>.
- [25] URL: <https://procreate.com>.
- [26] Accessed on 13/03/2024. URL: <https://wireframe.cc>.
- [27] Stuart K Card, Thomas P Moran, and Allen Newell. “The model human processor: An engineering model of human performance.” In: (Jan. 1986).
- [28] Accessed on 13/03/2024. URL: <https://react.dev>.
- [29] Accessed on 13/03/2024. URL: <https://mui.com>.
- [30] Accessed on 13/03/2024. URL: <https://www.axios.com>.
- [31] Accessed on 13/03/2024. URL: <https://www.mongodb.com>.
- [32] David Laribee. *Best Practice - An Introduction To Domain-Driven Design*. Feb. 2009. URL: <https://learn.microsoft.com/en-us/archive/msdn-magazine/2009/february/best-practice-an-introduction-to-domain-driven-design>.
- [33] Sourabh Sharma. *Mastering Microservices with Java*. 3rd ed. Packt Publishing Ltd, Feb. 2019. ISBN: 9781789531251.
- [34] Microsoft. *What Is Java Spring Boot?—Intro to Spring Boot — Microsoft Azure*. Accessed on 27/11/2023. URL: <https://azure.microsoft.com/en-gb/resources/cloud-computing-dictionary/what-is-java-spring-boot>.
- [35] Apache. *Maven – Welcome to Apache Maven*. Accessed on 27/11/2023. URL: <https://maven.apache.org>.
- [36] Accessed on 13/03/2024. URL: <https://www.nltk.org>.
- [37] Accessed on 13/03/2024. URL: <https://radimrehurek.com/gensim/models/word2vec.html>.
- [38] Accessed on 13/03/2024. URL: <https://faker.readthedocs.io/en/master/>.

- [39] Accessed on 13/03/2024. URL: <https://matplotlib.org>.
- [40] Accessed on 13/03/2024. URL: <https://junit.org/junit5/>.
- [41] Accessed on 13/03/2024. URL: <https://docs.python.org/3/library/unittest.html>.
- [42] Accessed on 13/03/2024. URL: <https://www.postman.com>.
- [43] Fred D. Davis. “Perceived Usefulness, Perceived Ease of Use, and User Acceptance of Information Technology”. In: *MIS Quarterly* 13.3 (1989), pp. 319–340. ISSN: 02767783. URL: <http://www.jstor.org/stable/249008> (visited on 02/21/2024).
- [44] Google forms. *Google Forms: Free Online Surveys for Personal Use*. 2019. URL: <https://www.google.com/forms/about/>.
- [45] Guy Shani and Asela Gunawardana. “Evaluating Recommendation Systems”. In: *Recommender Systems Handbook*. Ed. by Francesco Ricci et al. Boston, MA: Springer US, 2011, pp. 257–297. ISBN: 978-0-387-85820-3. DOI: 10.1007/978-0-387-85820-3_8. URL: https://doi.org/10.1007/978-0-387-85820-3_8.

Appendix A

Colour Pallete analysis with different colour blindness.

The following figures show the impact different types of colour blindness have on the chosen colour palette. These images were produced with help using Colours.com



Figure A.1: Colour palette compared with Protanopia



Figure A.2: Colour palette compared with Deuteranopia



Figure A.3: Colour palette compared with Tritanopia



Figure A.4: Colour palette compared with Protanomaly

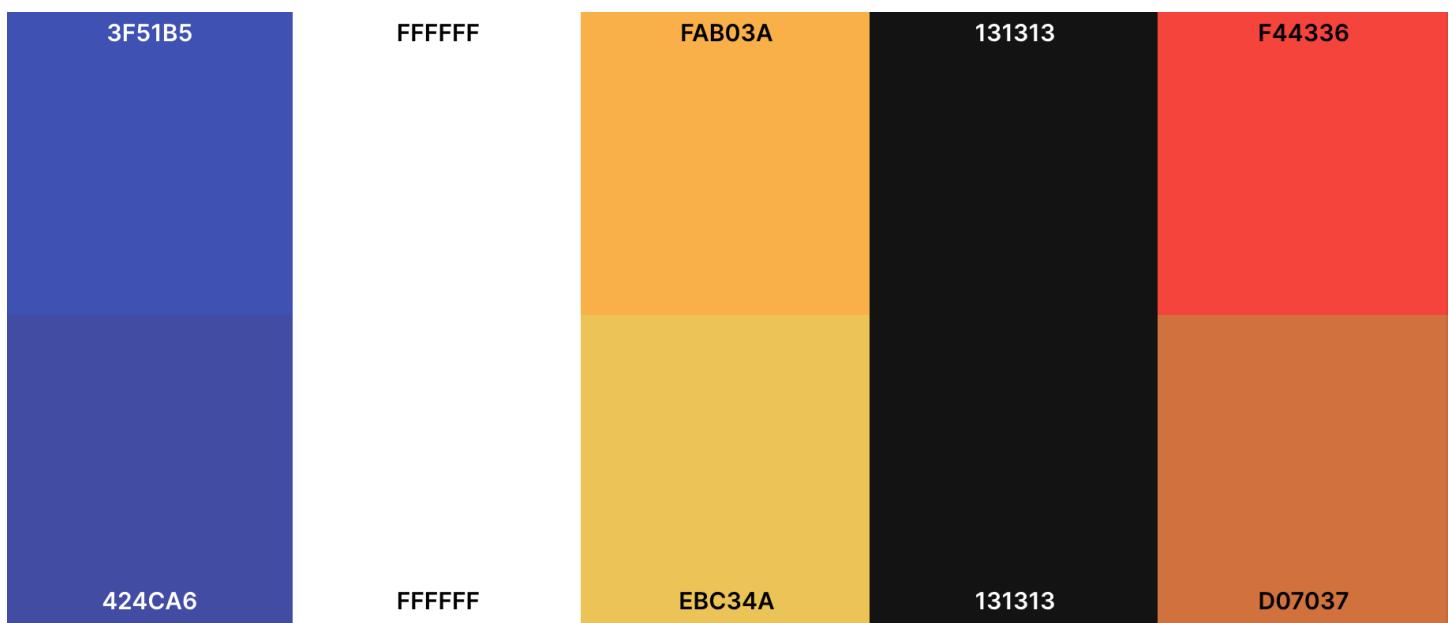


Figure A.5: Colour palette compared with Deutranomaly

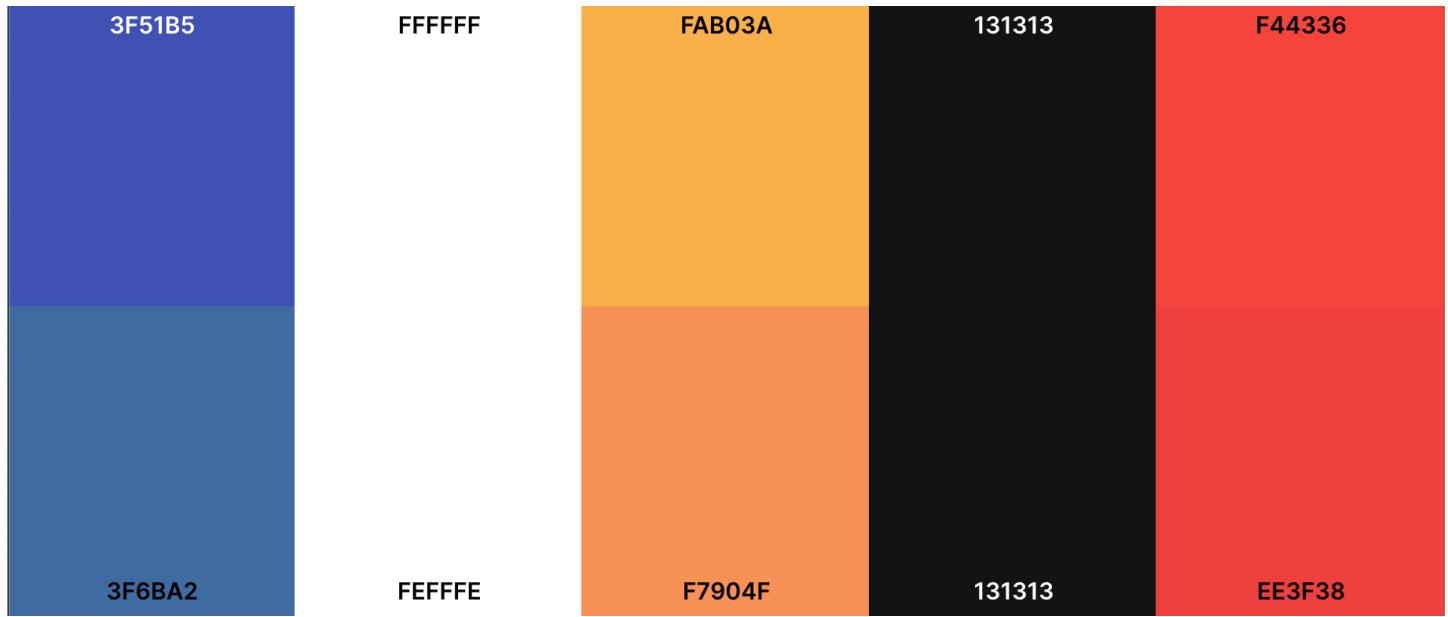


Figure A.6: Colour palette compared with Triatanomaly

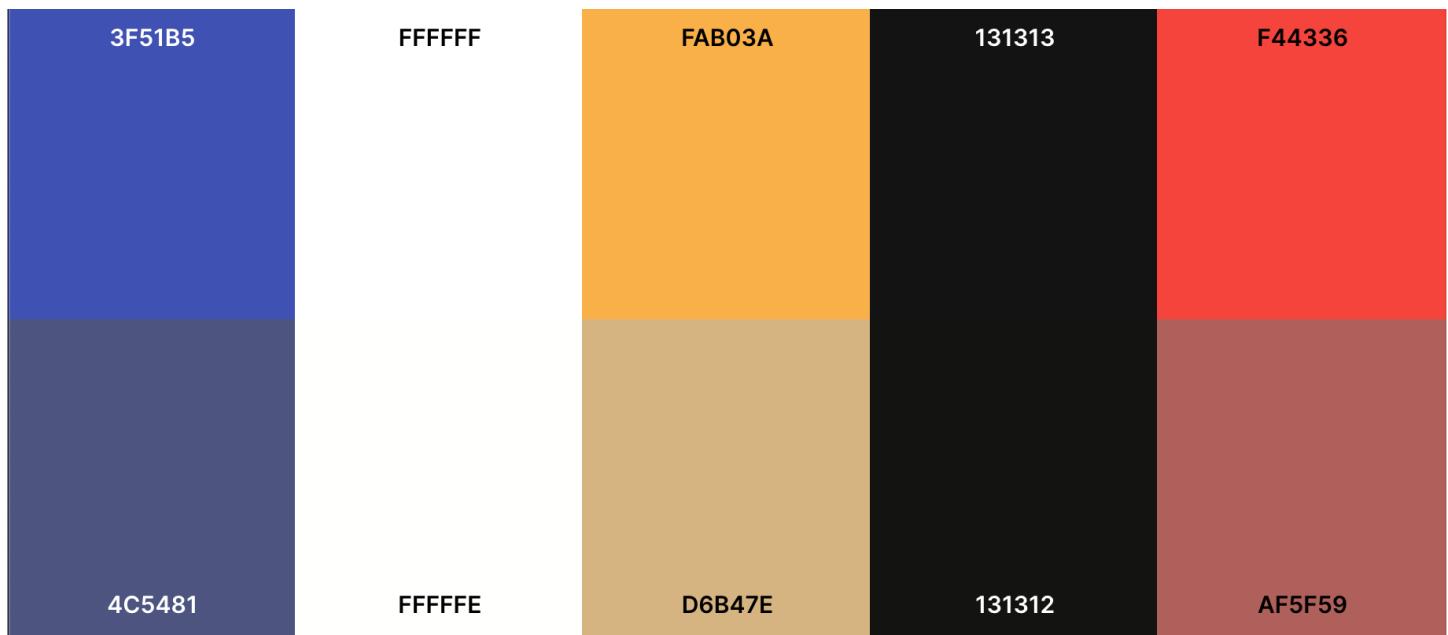


Figure A.7: Achromatomaly

Appendix B

K-means clustering analysis on User Profiles for Similar User recommendations for vary- ing K values

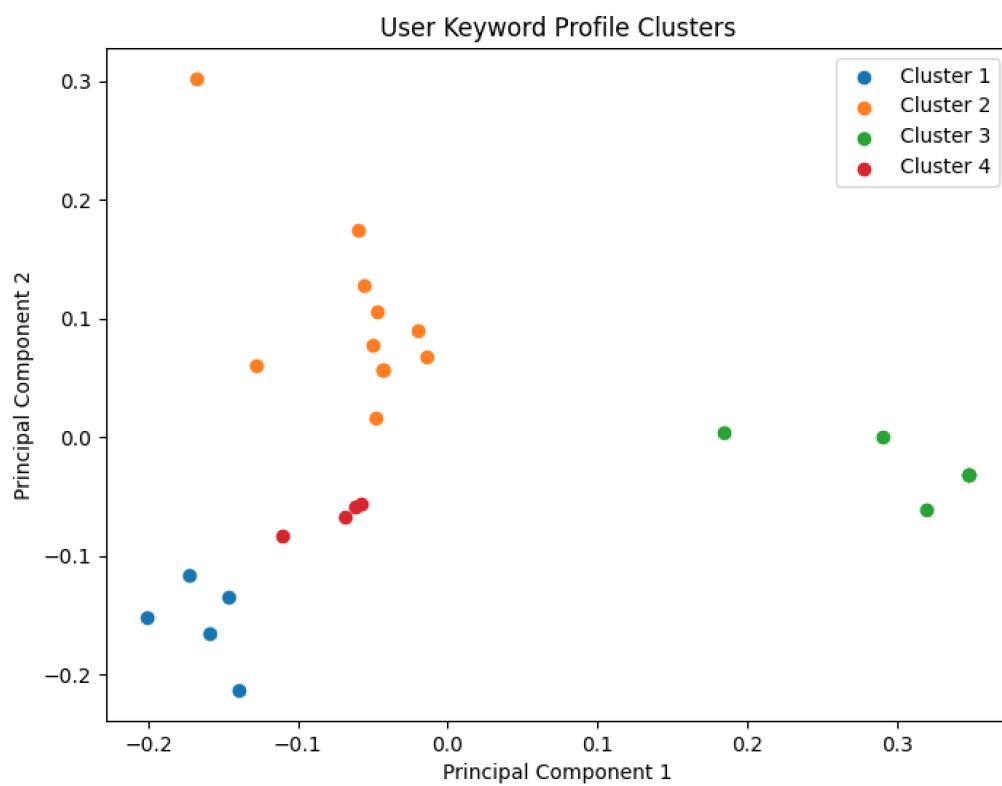


Figure B.1: K-means clustering on User profiles with k=4

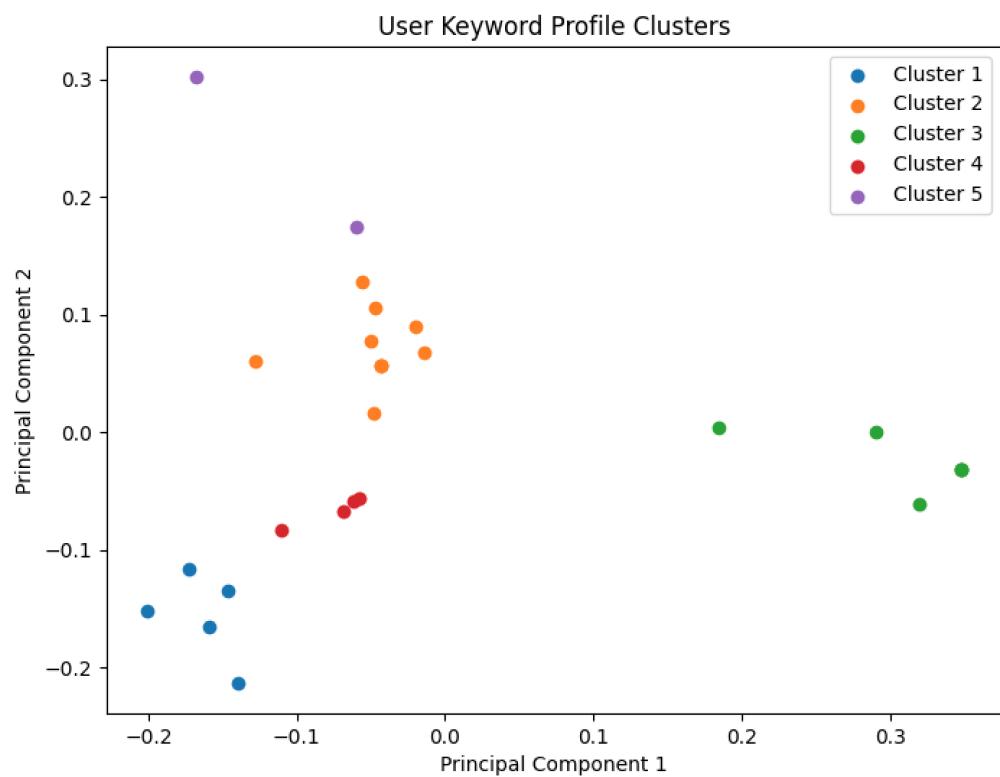


Figure B.2: K-means clustering on User profiles with k=5

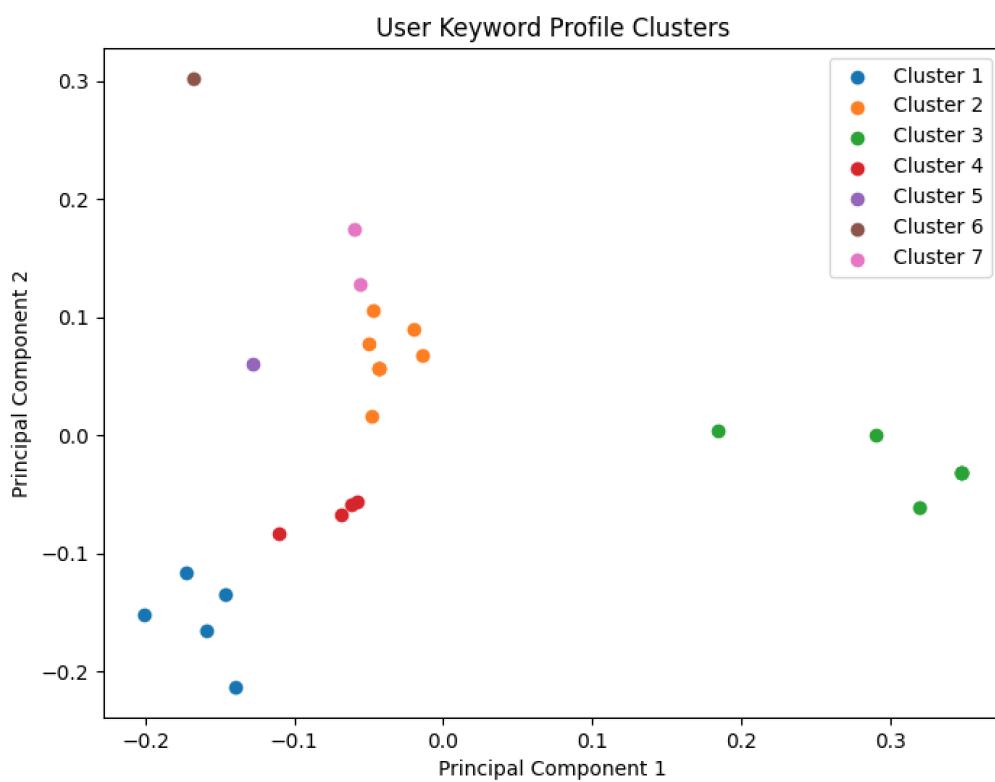


Figure B.3: K-means clustering on User profiles with k=7

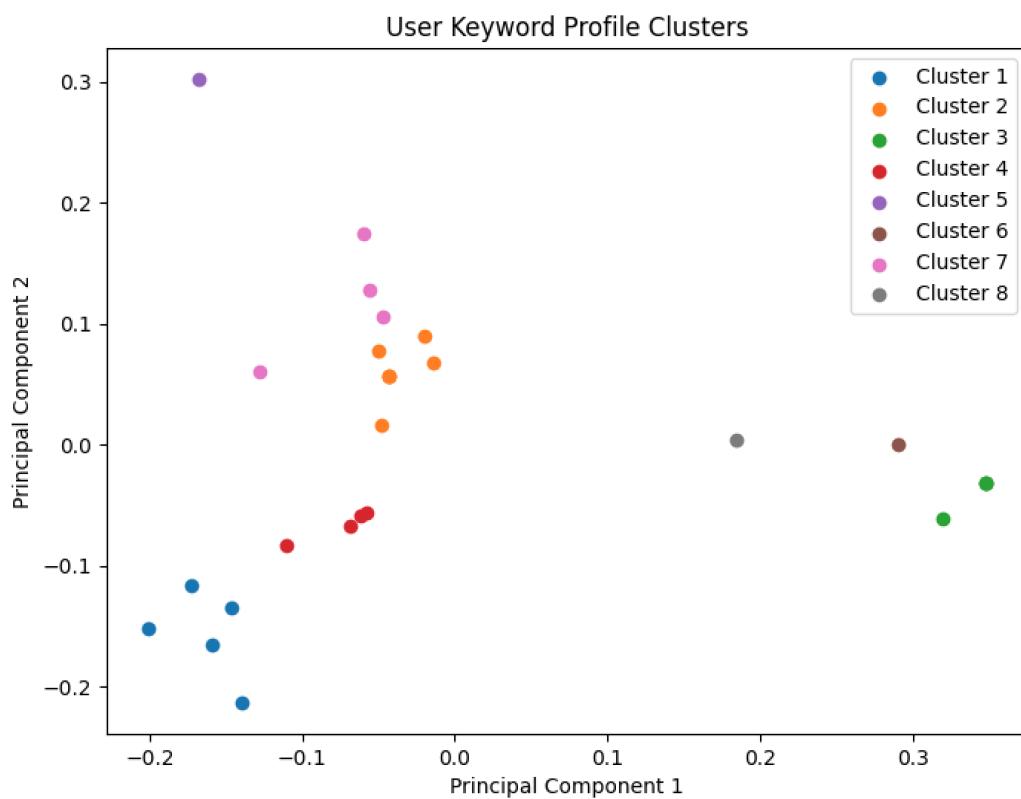


Figure B.4: K-means clustering on User profiles with k=8

Appendix C

Responses for Usability Survey Question 1

What Description did you enter

12 responses



Figure C.1: Participant responses from Usability Event Question 1: “What description did you enter?”

Appendix D

K-means clustering values on Skills for the Technical Survey

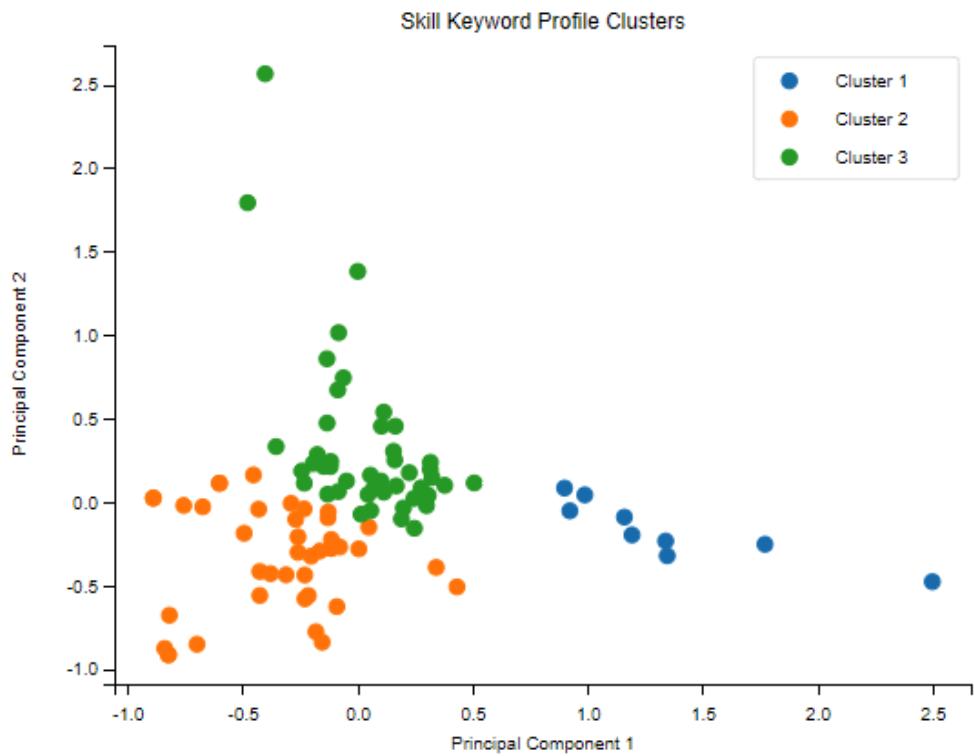


Figure D.1: K-means clustering on the Skills database with k=3

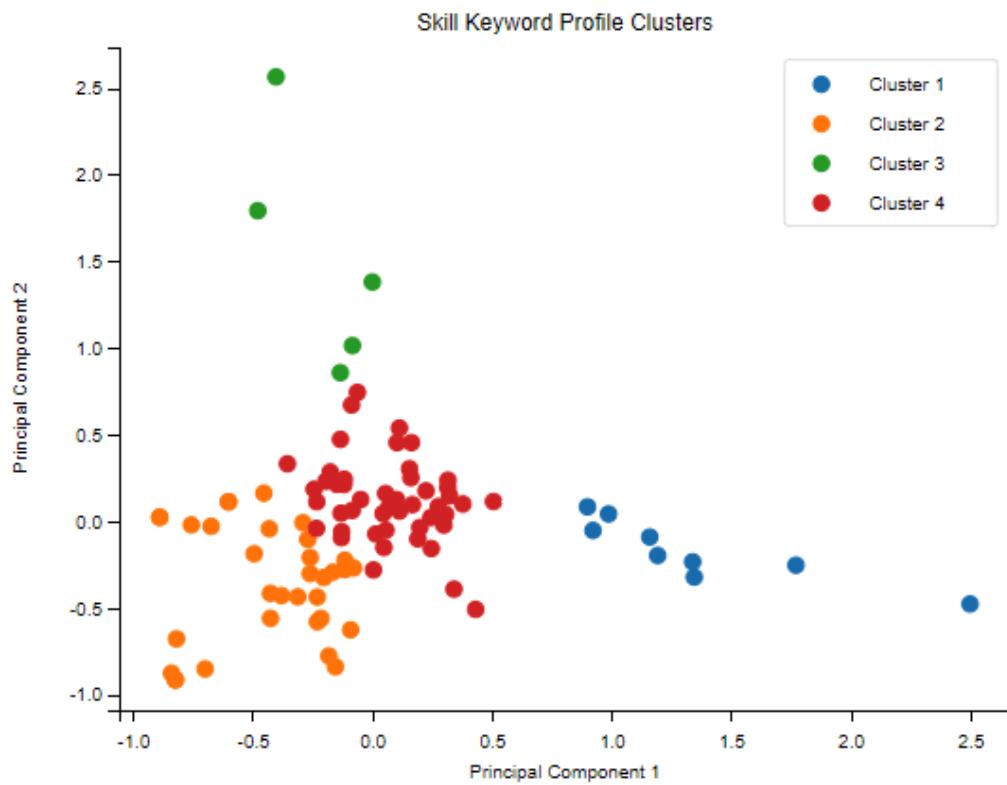


Figure D.2: K-means clustering on the Skills database with k=4

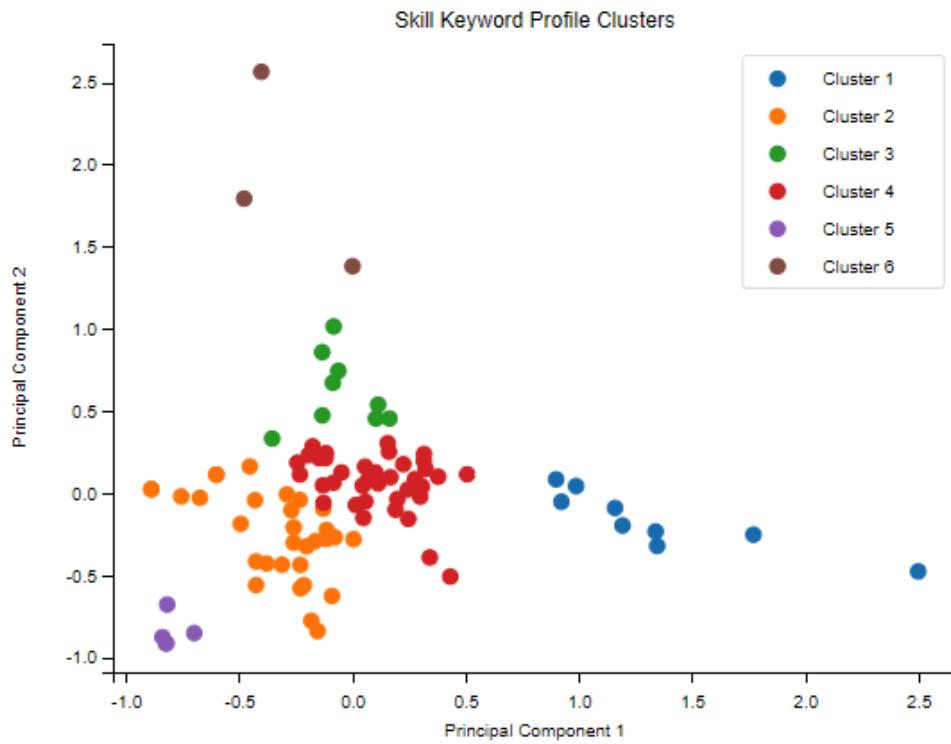


Figure D.3: K-means clustering on the Skills database with k=6

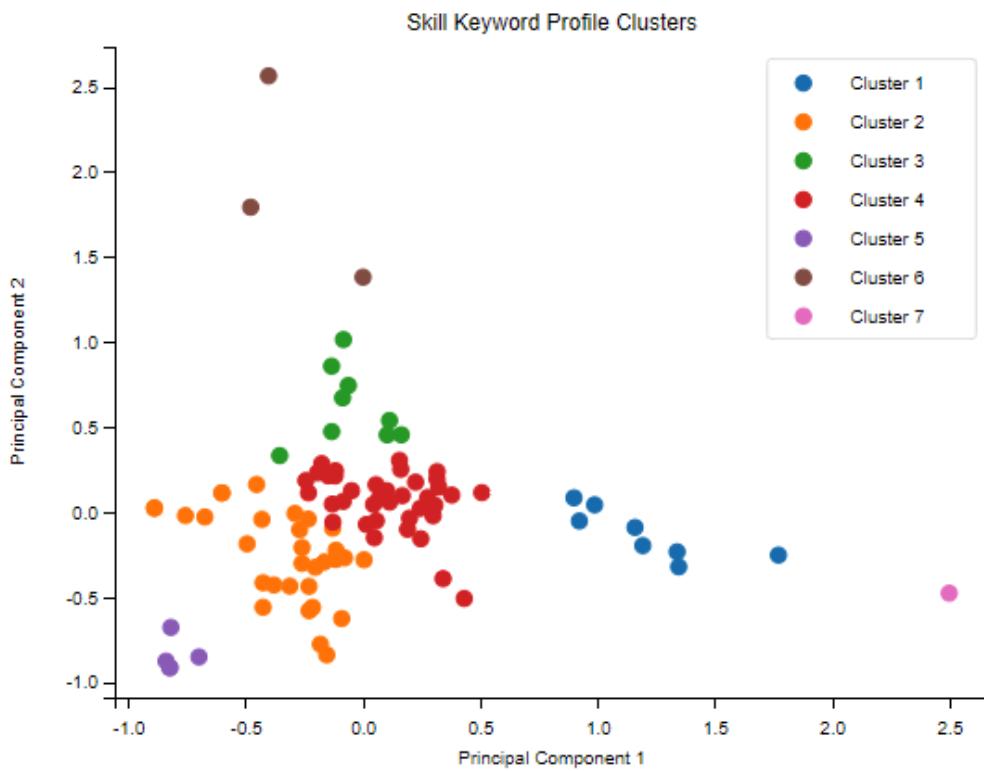


Figure D.4: K-means clustering on the Skills database with k=7

Appendix E

Technical Survey Question Format

Skill Boost Technical Survey

lucyinett02@gmail.com [Switch accounts](#)



Not shared

Ranking Skills based on a description

The following questions will ask you to choose the most relevant options that relate to a given topic. If you are unsure on what the topic or any of the options mean, please choose 'Not Sure'

Please select the 3 options you believe are most relevant to learning about:

Artificial Intelligence and Machine Learning

- A Robotic Processes Automation Forum: Learn about the latest trends and applications of robotic processes.
- Not Sure
- TensorFlow: An open-source machine learning framework developed by Google. It allows developers to build and train neural networks for various tasks such as image classification, natural language processing, and reinforcement learning.
- PyTorch: an open-source machine learning framework developed by Facebook. It provides a flexible and dynamic computational graph that makes it easy to build and train deep learning models.
- DevOps for enterprise business agility course: Learn the the business application of cloud computing technologies for the enterprise.
- scikit-learn: A machine learning library for Python. It provides simple and efficient tools for data mining and data analysis, including classification, regression, clustering, and dimensionality reduction.

Please select the 3 options you believe are most relevant to learning about: User Interface and Web Development

- Software Engineering Conference: Connect with software engineers, architects, and developers to explore best practices, methodologies, and tools for building high-quality software at scale.
- AWS Lambda: A server-less computing service provided by Amazon Web Services. It allows developers to run code in response to events without provisioning or managing servers, making it easy to build scalable and cost-effective applications.
- Fundamentals of sustainability and Technology course: Learn how data analytics, artificial intelligence (AI), and hybrid cloud computing are revolutionizing ways that humans support each other, while protecting Earth's resources.
- Not Sure
- C++: A general-purpose programming language known for its performance and efficiency. It is commonly used for system programming, game development, and high-performance computing.
- Apache Spark: A fast and general-purpose cluster computing system for big data processing. It provides APIs for building parallel applications and supports various programming languages, including Java, Scala, and Python.



Please select the 3 options you believe are most relevant to learning about: Data Science and Big Data

- Edge Computing conference event: Explore the transformative potential of edge computing in IoT, AI, and other domains, and learn how edge computing architectures are reshaping the future of computing.
- Apache Spark: A fast and general-purpose cluster computing system for big data processing. It provides APIs for building parallel applications and supports various programming languages, including Java, Scala, and Python.
- Getting started with enterprise data science course: Learn concepts, methods, and tools related to Data Science roles and their use of technology applied to enterprise projects.
- Not Sure
- Kafka Streams: A client library for building real-time, event-driven applications and microservices using Apache Kafka. It provides APIs for processing and analyzing streams of data in a scalable and fault-tolerant manner.
- Enterprise data science in practice course: Learn concepts, methods, and tools related to the data science methodology.

Please select the 3 options you believe are most relevant to learning about: Cyber Security and Encryption

- Quantum cryptography workshop event: Delve into the world of quantum cryptography and learn about the latest techniques for secure communication in the quantum era at our interactive workshop.
- Enterprise security in practice course: Develop skills and understanding in the approaches to elevate an organization's overall security posture, by adopting practices, methods and tools that increase enterprise cyber resilience.
- Building cloud-based mobile solutions for the enterprise course: Learn concepts, methods, and tools related to exploring the range of underlying cloud technologies used for application development.
- Go: also known as Golang, is a statically typed, compiled programming language developed by Google. It is designed for simplicity, performance, and concurrency, making it ideal for building scalable web services and systems software.
- Not Sure
- Building AI Solutions Using Advanced Algorithms and Open Source Frameworks Course: Learn concepts related to building AI solutions with open-source frameworks



Please select the 3 options you believe are most relevant to: Databases and SQL

- Artificial Intelligence fundamentals course: Learn about artificial intelligence (AI) concepts, such as natural language processing, computer vision, machine learning, deep learning, chatbots, and neural networks; AI ethics; and the applications of AI.
- Jupyter Notebook: An open-source web application for creating and sharing documents that contain live code, equations, visualizations, and narrative text. It supports various programming languages, including Python, R, and Julia.
- Apache Kafka : A distributed streaming platform used for building real-time data pipelines and event-driven applications. It provides high throughput, fault tolerance, and horizontal scalability.
- Not Sure
- Getting started with enterprise-grade AI course: Learn about the foundations of Artificial Intelligence for business,
- Journey to cloud: envisioning your solution course: Learn about how cloud works, its capabilities, types, and delivery models.

Please rank the following options from most to least relevant to: Web Development

	Most Relevant	Middle (Moderately) Relevant	Least Relevant
DevSecOps Summit event: Join security and DevOps professionals to discuss strategies for integrating security into the software development lifecycle, ensuring secure and resilient applications.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Python: a high-level programming language known for its simplicity and readability. It is widely used for web development, data analysis, artificial intelligence, and scientific computing.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
DevOps for enterprise business Agility course: Learn concepts, methods, and tools related to the business application of cloud computing technologies for the enterprise.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>



Please rank the following options from most to least relevant to: Databases

Most Relevant Middle (Moderately) Relevant Least Relevant

Django: A high-level Python web framework that encourages rapid development and clean, pragmatic design. It follows the Model-View-Template (MVT) pattern and includes built-in features for authentication, routing, and database management.

Flask-SQLAlchemy: An extension for Flask that adds support for SQLAlchemy, a powerful SQL toolkit and ORM. It simplifies database integration in Flask applications and provides tools for database migrations, querying, and schema management.

MongoDB: A NoSQL database management system known for its flexibility and scalability. It stores data in flexible, JSON-like documents and is commonly used for big data and real-time applications.

Please rank the following options from most to least relevant to: Artificial Intelligence

Most Relevant Middle (Moderately) Relevant Least Relevant

Artificial Intelligence ethics forum event:
Engage in thought-provoking discussions about the ethical implications of artificial intelligence and machine learning, exploring topics such as bias, privacy, and accountability.

AI in Finance Conference: Explore the transformative impact of artificial intelligence on the finance industry, from algorithmic trading and risk management to customer service and fraud detection.

AI and healthcare symposium:
Discover how artificial intelligence is revolutionizing healthcare delivery, diagnostics, and patient care, and explore the ethical and regulatory considerations of AI in healthcare.

 [Back](#)[Next](#)[Clear form](#)

Never submit passwords through Google Forms.

This content is neither created nor endorsed by Google. [Report Abuse](#) - [Terms of Service](#) - [Privacy Policy](#)

Google Forms





Appendix F

ChatGPT Data Generation for Software Skills

Prompt: I need to create a dummy dataset for my recommendation system. please could you generate a database with 50 entries, each with a programming language, library or tool as the title, a description of the software and then a link to its documentation **Response:** Sure, here's a dummy dataset with 50 entries for your recommendation system:

1. Title: Python Description: Python is a high-level programming language known for its simplicity and readability. It is widely used for web development, data analysis, artificial intelligence, and scientific computing. Documentation: [Python Documentation](<https://docs.python.org/3/>)
2. Title: TensorFlow Description: TensorFlow is an open-source machine learning framework developed by Google. It allows developers to build and train neural networks for various tasks such as image classification, natural language processing, and reinforcement learning. Documentation: [TensorFlow Documentation](https://www.tensorflow.org/api_docs)
3. Title: React Description: React is a JavaScript library for building user interfaces. It allows developers to create reusable UI components and efficiently manage the state of their applications. Documentation: [React Documentation](<https://reactjs.org/docs/getting-started.html>)
4. Title: Java Description: Java is a popular programming language known for its platform independence. It is used for building enterprise-level applications, mobile apps, and web services. Documentation: [Java Documentation](<https://docs.oracle.com/en/java/>)
5. Title: Pandas Description: Pandas is a Python library for data manipulation and analysis. It provides data structures and functions for efficiently working with structured data such as tables and time series. Documentation: [Pandas Documentation](<https://pandas.pydata.org/docs/>)
6. Title: Docker Description: Docker is a platform for developing, shipping, and running applications in containers. It allows developers to package their applications and dependencies into portable containers that can run on any environment. Documentation: [Docker Documentation](<https://docs.docker.com/>)
7. Title: Vue.js Description: Vue.js is a progressive JavaScript framework for building user interfaces.

It focuses on the view layer and provides tools for building interactive web applications. Documentation: [Vue.js Documentation](<https://vuejs.org/v2/guide/>)

8. Title: C++ Description: C++ is a general-purpose programming language known for its performance and efficiency. It is commonly used for system programming, game development, and high-performance computing. Documentation: [C++ Documentation](<https://en.cppreference.com/w/cpp>)

9. Title: NumPy Description: NumPy is a Python library for numerical computing. It provides support for large, multi-dimensional arrays and matrices, along with a collection of mathematical functions to operate on these arrays. Documentation: [NumPy Documentation](<https://numpy.org/doc/stable/>)

10. Title: Node.js Description: Node.js is a JavaScript runtime built on Chrome's V8 JavaScript engine. It allows developers to run JavaScript code outside of a web browser, enabling server-side scripting and building scalable network applications. Documentation: [Node.js Documentation](<https://nodejs.org/en/docs/>)

11. Title: Ruby on Rails Description: Ruby on Rails is a web application framework written in Ruby. It follows the Model-View-Controller (MVC) architecture and emphasizes convention over configuration, making it easier to build and maintain web applications. Documentation: [Ruby on Rails Documentation](<https://guides.rubyonrails.org/>)

12. Title: Flask Description: Flask is a lightweight web framework for Python. It is designed to be simple and easy to use, making it ideal for building small to medium-sized web applications and APIs. Documentation: [Flask Documentation](<https://flask.palletsprojects.com/en/2.0.x/>)

13. Title: Rust Description: Rust is a systems programming language known for its safety and concurrency features. It aims to provide low-level control without sacrificing high-level abstractions, making it suitable for performance-critical applications. Documentation: [Rust Documentation](<https://doc.rust-lang.org/>)

14. Title: scikit-learn Description: scikit-learn is a machine learning library for Python. It provides simple and efficient tools for data mining and data analysis, including classification, regression, clustering, and dimensionality reduction. Documentation: [scikit-learn Documentation](<https://scikit-learn.org/stable/documentation.html>)

15. Title: Angular Description: Angular is a TypeScript-based web application framework developed by

Google. It is used for building single-page applications and dynamic web pages, with a focus on modularity and testability. Documentation: [Angular Documentation](<https://angular.io/docs>)

16. Title: PostgreSQL Description: PostgreSQL is a powerful, open-source relational database management system. It supports a wide range of features, including ACID transactions, foreign keys, and multi-version concurrency control. Documentation: [PostgreSQL Documentation](<https://www.postgresql.org/docs/>)
17. Title: Git Description: Git is a distributed version control system used for tracking changes in source code during software development. It allows multiple developers to collaborate on projects and manage their codebase efficiently. Documentation: [Git Documentation](<https://git-scm.com/doc>)
18. Title: Matplotlib Description: Matplotlib is a Python library for creating static, animated, and interactive visualizations. It provides a wide range of plotting functions for generating graphs, histograms, scatter plots, and more. Documentation: [Matplotlib Documentation](<https://matplotlib.org/stable/contents.html>)
19. Title: Spring Boot Description: Spring Boot is a Java-based framework for building stand-alone, production-grade Spring applications. It provides a set of tools and conventions to streamline the development process and reduce boilerplate code. Documentation: [Spring Boot Documentation](<https://docs.spring.io/spring-boot/docs/current/reference/html/>)
20. Title: Keras Description: Keras is a high-level neural networks API written in Python. It provides a user-friendly interface for building and training deep learning models, with support for convolutional networks, recurrent networks, and more. Documentation: [Keras Documentation](<https://keras.io/api/>)
21. Title: Ruby Description: Ruby is a dynamic, object-oriented programming language known for its simplicity and productivity. It is used for web development, automation, and prototyping. Documentation: [Ruby Documentation](<https://www.ruby-lang.org/en/documentation/>)
22. Title: SQLAlchemy Description: SQLAlchemy is a SQL toolkit and Object-Relational Mapping (ORM) library for Python. It provides a powerful and flexible way to interact with databases using Python code. Documentation: [SQLAlchemy Documentation](<https://docs.sqlalchemy.org/en/21/>)
23. Title: Bootstrap Description: Bootstrap is a popular front-end framework for building responsive and mobile-first websites. It provides a collection of CSS and JavaScript components for creating sleek and modern user interfaces. Documentation: [Bootstrap Documentation](<https://getbootstrap.com/docs/5.1/getting-started/>)

started/introduction/)

24. Title: Django Description: Django is a high-level Python web framework that encourages rapid development and clean, pragmatic design. It follows the Model-View-Template (MVT) pattern and includes built-in features for authentication, routing, and database management. Documentation: [Django Documentation](<https://docs.djangoproject.com/en/4.0/>)

25. Title: Swift Description: Swift is a powerful and intuitive programming language developed by Apple for building iOS, macOS, watchOS, and

tvOS applications. It is designed to be safe, fast, and expressive. Documentation: [Swift Documentation](<https://swift.org/documentation/>)

26. Title: Apache Spark Description: Apache Spark is a fast and general-purpose cluster computing system for big data processing. It provides APIs for building parallel applications and supports various programming languages, including Java, Scala, and Python. Documentation: [Apache Spark Documentation](<https://spark.apache.org/docs/latest/>)

27. Title: Express.js Description: Express.js is a minimalist web framework for Node.js. It provides a robust set of features for building web applications and APIs, including routing, middleware support, and template engines. Documentation: [Express.js Documentation](<https://expressjs.com/en/5x/api.html>)

28. Title: Unity Description: Unity is a cross-platform game engine used for developing 2D and 3D games, simulations, and interactive experiences. It provides a comprehensive set of tools for game development, including physics, rendering, and scripting. Documentation: [Unity Documentation](<https://docs.unity3d.com>)

29. Title: Flask RESTful Description: Flask-RESTful is an extension for Flask that adds support for building REST APIs. It simplifies the process of creating web services by providing tools for request parsing, response formatting, and error handling. Documentation: [Flask-RESTful Documentation](<https://flask-restful.readthedocs.io/en/latest/>)

30. Title: Apache Kafka Description: Apache Kafka is a distributed streaming platform used for building real-time data pipelines and event-driven applications. It provides high throughput, fault tolerance, and horizontal scalability. Documentation: [Apache Kafka Documentation](<https://kafka.apache.org/documentation/>)

31. Title: Electron Description: Electron is a framework for building cross-platform desktop applica-

tions using web technologies such as HTML, CSS, and JavaScript. It allows developers to create native-like apps with web technologies. Documentation: [Electron Documentation](<https://www.electronjs.org/docs>)

32. Title: MongoDB Description: MongoDB is a NoSQL database management system known for its flexibility and scalability. It stores data in flexible, JSON-like documents and is commonly used for big data and real-time applications. Documentation: [MongoDB Documentation](<https://docs.mongodb.com/>)

33. Title: Flask-SQLAlchemy Description: Flask-SQLAlchemy is an extension for Flask that adds support for SQLAlchemy, a powerful SQL toolkit and ORM. It simplifies database integration in Flask applications and provides tools for database migrations, querying, and schema management. Documentation: [Flask-SQLAlchemy Documentation](<https://flask-sqlalchemy.palletsprojects.com/en/3.x/>)

34. Title: PyTorch Description: PyTorch is an open-source machine learning framework developed by Facebook. It provides a flexible and dynamic computational graph that makes it easy to build and train deep learning models. Documentation: [PyTorch Documentation](<https://pytorch.org/docs/stable/index.html>)

35. Title: GraphQL Description: GraphQL is a query language for APIs and a runtime for executing those queries. It provides a more efficient, powerful, and flexible alternative to traditional REST APIs by allowing clients to request only the data they need. Documentation: [GraphQL Documentation](<https://graphql.org/learn/>)

36. Title: Go Description: Go, also known as Golang, is a statically typed, compiled programming language developed by Google. It is designed for simplicity, performance, and concurrency, making it ideal for building scalable web services and systems software. Documentation: [Go Documentation](<https://golang.org/doc/>)

37. Title: Selenium Description: Selenium is a portable framework for testing web applications. It provides a set of tools for automating web browsers and simulating user interactions, making it useful for testing web applications across different browsers and platforms. Documentation: [Selenium Documentation](<https://www.selenium.dev/documentation/en/>)

38. Title: Redux Description: Redux is a predictable state container for JavaScript apps. It helps manage the state of your application in a single immutable store, making it easier to develop complex UIs and maintain consistent behavior across components. Documentation: [Redux Documentation](<https://redux.js.org/>)

started)

39. Title: ASP.NET Core Description: ASP.NET Core is a cross-platform, high-performance framework for building modern, cloud-based web applications. It provides support for building web APIs, web applications, and real-time services using .NET. Documentation: [ASP.NET Core Documentation](<https://docs.microsoft.com/en-us/aspnet/core/>)
40. Title: TensorFlow.js Description: TensorFlow.js is a JavaScript library for training and deploying machine learning models in the browser and on Node.js. It provides tools for building and training models using TensorFlow's high-level API. Documentation: [TensorFlow.js Documentation](<https://www.tensorflow.org/js>)
41. Title: Django REST Framework Description: Django REST Framework is a powerful and flexible toolkit for building Web APIs in Django. It provides a set of tools for serializing and deserializing complex data types, handling authentication and permissions, and building browsable APIs. Documentation: [Django REST Framework Documentation](<https://www.django-rest-framework.org/documentation>)
42. Title: AWS Lambda Description: AWS Lambda is a serverless computing service provided by Amazon Web Services. It allows developers to run code in response to events without provisioning or managing servers, making it easy to build scalable and cost-effective applications. Documentation: [AWS Lambda Documentation](<https://docs.aws.amazon.com/lambda/>)
43. Title: Flask-WTF Description: Flask-WTF is an extension for Flask that integrates with the WTForms library, allowing developers to create and validate forms in Flask applications. It provides tools for generating forms, handling submissions, and protecting against CSRF attacks. Documentation: [Flask-WTF Documentation](<https://flask-wtf.readthedocs.io/en/stable/>)
44. Title: Jupyter Notebook Description: Jupyter Notebook is an open-source web application for creating and sharing documents that contain live code, equations, visualizations, and narrative text. It supports various programming languages, including Python, R, and Julia. Documentation: [Jupyter Notebook Documentation](<https://jupyter-notebook.readthedocs.io/en/stable/>)
45. Title: TensorFlow Lite Description: TensorFlow Lite is a lightweight version of TensorFlow optimized for mobile and embedded devices. It allows developers to deploy machine learning models on smartphones, IoT devices, and microcontrollers with low latency and power consumption. Documenta-

tion: [TensorFlow Lite Documentation](<https://www.tensorflow.org/lite>)

46. Title: Kafka Streams Description: Kafka Streams is a client library for building real-time, event-driven applications and microservices using Apache Kafka. It provides APIs for processing and analyzing streams of data in a scalable and fault-tolerant manner. Documentation: [Kafka Streams Documentation](<https://kafka.apache.org/documentationstreams/>)

47. Title: Flutter Description: Flutter is a cross-platform UI toolkit for building natively compiled applications for mobile, web, and desktop from a single codebase. It provides a rich set of pre-built widgets and tools for building beautiful and highly performant apps. Documentation: [Flutter Documentation](<https://flutter.dev/docs>)

48. Title: AWS SDK for JavaScript Description: The AWS SDK for JavaScript is a collection of libraries and tools for building applications with Amazon Web Services using JavaScript and Node.js. It provides APIs for interacting with various AWS services, such as S3, DynamoDB, and Lambda. Documentation: [AWS SDK for JavaScript Documentation](<https://docs.aws.amazon.com/AWSJavaScriptSDK/latest/>)

49. Title: Vue Router Description: Vue Router is the official router for Vue.js. It provides a way to navigate between different views or pages in a Vue.js application, with support for nested routes, named routes, and route parameters. Documentation: [Vue Router Documentation](<https://router.vuejs.org/>)

50. Title: SQLAlchemy-Utils Description: SQLAlchemy-Utils is a collection of utility functions and custom data types for SQLAlchemy. It provides additional features and tools for working with databases in SQLAlchemy applications, such as UUIDs, JSON fields, and encrypted columns. Documentation: [SQLAlchemy-Utils Documentation](<https://sqlalchemy-utils.readthedocs.io/en/latest/>)

Appendix G

ChatGPT Data Generation for Event Skills

Prompt: can you please create another dummy dataset for 50 computer science events, where the event has a title (i want the events to be on a wide range of computer science things), description of the event, details (time and date where the date is after 1st April 2024)

Response: Sure, here's a dummy dataset for 50 computer science events:

1. Title: Machine Learning Summit 2024 Description: Join us for the annual Machine Learning Summit where experts from academia and industry will discuss the latest advancements, trends, and applications in machine learning and artificial intelligence. Details: April 15-17, 2024
2. Title: Cybersecurity Conference Expo Description: This conference brings together cybersecurity professionals, researchers, and practitioners to discuss emerging threats, strategies for defense, and innovative solutions in cybersecurity. Details: May 5-7, 2024
3. Title: Data Science Hackathon Description: Calling all data enthusiasts! Participate in our hackathon to tackle real-world data challenges and showcase your data science skills. Prizes and networking opportunities await! Details: June 8-10, 2024
4. Title: DevOps World Description: DevOps World is a premier conference for developers, operations professionals, and industry leaders to explore best practices, tools, and methodologies for continuous integration and delivery. Details: July 12-14, 2024
5. Title: Quantum Computing Symposium Description: Dive into the fascinating world of quantum computing at our symposium, featuring talks, workshops, and hands-on demonstrations by leading experts in the field. Details: August 18-20, 2024
6. Title: Blockchain Summit Description: Discover the latest developments and applications of blockchain technology at our summit, featuring keynote speeches, panel discussions, and interactive workshops. Details: September 7-9, 2024
7. Title: Cloud Computing Expo Description: Explore the future of cloud computing at our expo, where industry leaders will showcase innovative solutions, strategies, and trends shaping the cloud landscape.

Details: October 11-13, 2024

8. Title: Robotics Symposium Description: Join us for a deep dive into the field of robotics, featuring talks on autonomous systems, human-robot interaction, and the latest advancements in robotics research.

Details: November 15-17, 2024

9. Title: Internet of Things (IoT) Conference Description: Learn about the latest developments and applications of IoT technology, including smart devices, sensors, and IoT platforms, at our interactive conference. Details: December 6-8, 2024

10. Title: Artificial Intelligence Ethics Forum Description: Engage in thought-provoking discussions about the ethical implications of artificial intelligence and machine learning, exploring topics such as bias, privacy, and accountability. Details: January 18-20, 2025

11. Title: Big Data Expo Description: Discover how organizations are leveraging big data to drive innovation, improve decision-making, and gain a competitive edge in today's data-driven world. Details: February 22-24, 2025

12. Title: Cybersecurity Bootcamp Description: Enhance your cybersecurity skills and knowledge at our intensive bootcamp, featuring hands-on training, interactive workshops, and expert-led sessions. Details: March 8-10, 2025

13. Title: Software Engineering Conference Description: Connect with software engineers, architects, and developers to explore best practices, methodologies, and tools for building high-quality software at scale. Details: April 19-21, 2025

14. Title: Quantum Cryptography Workshop Description: Delve into the world of quantum cryptography and learn about the latest techniques for secure communication in the quantum era at our interactive workshop. Details: May 14-16, 2025

15. Title: AI for Social Good Symposium Description: Explore how artificial intelligence can be used to address social, environmental, and humanitarian challenges, and learn about impactful AI initiatives from around the world. Details: June 20-22, 2025

16. Title: DevSecOps Summit Description: Join security and DevOps professionals to discuss strategies for integrating security into the software development lifecycle, ensuring secure and resilient applications.

Details: July 9-11, 2025

17. Title: Robotics Hackathon Description: Unleash your creativity and problem-solving skills at our robotics hackathon, where teams compete to develop innovative solutions to real-world challenges using robotics technology. Details: August 15-17, 2025

18. Title: Edge Computing Conference Description: Explore the transformative potential of edge computing in IoT, AI, and other domains, and learn how edge computing architectures are reshaping the future of computing. Details: September 12-14, 2025

19. Title: AI and Healthcare Symposium Description: Discover how artificial intelligence is revolutionizing healthcare delivery, diagnostics, and patient care, and explore the ethical and regulatory considerations of AI in healthcare. Details: October 17-19, 2025

20. Title: Data Privacy Summit Description: Dive into the complex world of data privacy and security, exploring regulatory compliance, privacy-enhancing technologies, and best practices for protecting sensitive data. Details: November 21-23, 2025

21. Title: Cloud-Native Applications Workshop Description: Learn how to design, develop, and deploy cloud-native applications using modern technologies and methodologies such as containers, microservices, and serverless computing. Details: December 19-21, 2025

22. Title: Quantum Machine Learning Seminar Description: Explore the intersection of quantum computing and machine learning, and learn about cutting-edge research and applications in quantum machine learning from leading experts. Details: January 23-25, 2026

23. Title: Cybersecurity Awareness Month Webinar Series Description: Join our series of webinars during Cybersecurity Awareness Month to learn about common cybersecurity threats, best practices for securing digital assets, and tips for staying safe online. Details: February 10-12, 2026

24. Title: AI in Finance Conference Description: Explore the transformative impact of artificial intelligence on the finance industry, from algorithmic trading and risk management to customer service and fraud detection. Details: March 18-20, 2026

25. Title: Hack the Climate Hackathon Description: Join forces with fellow hackers, developers, and climate enthusiasts to develop innovative solutions addressing climate change challenges, such as renewable

energy, conservation, and adaptation. Details: April 14-16, 2026

26. Title: Internet of Medical Things (IoMT) Summit Description: Explore the convergence of IoT and healthcare at our summit, featuring discussions on connected medical devices, remote patient monitoring, and the future of digital health. Details: May 19-21, 2026

27. Title: DevOps Leadership Retreat Description: Join DevOps leaders and executives for a retreat focused on organizational transformation, cultural change, and leadership strategies for driving DevOps adoption and success. Details: June 25-27, 2026

28. Title: AI Art and Creativity Showcase Description: Experience the intersection of artificial intelligence and art at our showcase, featuring AI-generated artwork, interactive installations, and performances exploring creativity and technology. Details: July 8-10, 2026

29. Title: Robotic Process Automation (RPA) Forum Description: Learn about the latest trends and applications of robotic process automation in business processes, finance, and operations, and discover how RPA can drive efficiency and innovation. Details: August 12-14, 2026