Technology Investigation Summary

For

CurtinTalentTrack

Version 1.0

Prepared By

SD06

Anjana Wishwajith Premadasa - 20463382
Malith Pramuditha - 20926076
Hesara Yasaswin Pathirana – 20928386
Uvindu Gamage - 20528142
Rizkan Rhazes – 21309380

August 29th, 2024

ISAD 3000 - Capstone Computing Project 1
Curtin University

Table of Contents

Part One - Main Document (TIS)

1. Introduction

1.1 Overview

The CurtinTalentTrack project is an inventive web application developed to develop communication skills in the members of the broad Curtin Colombo Community – students, academic staff and non-academic staff as well alumni. It is designed to help users improve public speaking, interview skills, writing and critical thinking. We want to incorporate identity exploration but also build a community together that is based on growth.

We are building CurtinTalentTrack with MERN stack, which is a neat combination of technologies: React as the frontend, Express. js for the middleware, Node. js, and then on the back-end side a new stack for us which is MongoDB. This stack is selected for its popularity, versatility and most importantly because it enables a unified development experience leveraging JavaScript in all application tiers.

1.2 Objectives of the Investigation

The main goal of this technology investigation report is to find the most appropriate technologies in order to develop application CurtinTalentTrack. The investigation makes sure that the selected technologies will sustain what our application intends to do when it comes to performance, scalability & security as well.

2. Front End Technology

2.1 Introduction

In the development of **CurtinTalentTrack** which is a web platform really focused on enhancing the basic communication skills within the Curtin community, selecting the best as well the most suitable front-end technology is and also was our main goal. As we know our platform aims to provide users ranging from students to staff and alumni with tools that are important for improving public speaking skills, written communication, and critical thinking. To provide a seamless, responsive, and also engaging user experience, we really need a front-end framework that could meet all these requirements.

After evaluating several options, we decided to use **React (Version 8.2.0)**. In this section, we'll walk you through the reason behind our choice, and how **React** aligns with the needs of **CurtinTalentTrack**. We'll explore factors such as functionality, support, and flexibility, and demonstrate why we chose and selected **React** as our final framework.

2.2 Front End Architecture

2.2.1 Modular Architecture

After a deep analysis and a careful consideration, we decided to use a **modular architectural** approach for the front-end of our **CurtinTalentTrack** platform. This decision really going to improve the scalability, flexibility, and efficiency while offering significant advantages over both **micro front-end architectures** and **monolithic architectures**.

2.2.1.1 Advantages of Modular Architecture

A modular architecture allows us to break down the complex **CurtinTalentTrack** into loosely connected and also highly reusable components within a single, unified system. This structure really going to provide the flexibility to add or update features without causing any disruption to the already built system. For example, integrating a new module for a ***machine learning*** feature or any other additional communication skill can be done smoothly if we follow and use the modular approach. This seamless integration is crucial for maintaining a consistent user experience across all modules, whether they cater to students, staff, or evaluators[R1].

As we know **micro front-ends** break the application into independently developed and deployable units, but unfortunately, they introduce significant integration complexities because of the extreme independence of the modules. So, managing these separate independent modules going to need additional infrastructure and coordination, which really complicates the development process. For **CurtinTalentTrack**, it's really better to have one unified application. A simple and organized modular structure will make things clearer and easier to manage [R1].

If we consider **monolithic architecture**, which combines all components into a single and also tightly connected system, which may seem kind of simple and nice at the beginning, but it is going to be really challenging as the platform grows. Even a simple minor update in a monolithic system can force us to redeploy the entire system, which going to be cumbersome and time-consuming. Moreover, the rigidity of a monolithic system would hinder our ability to quickly adapt to new educational needs or integrate advanced technologies like machine learning, which are critical for the future growth of **CurtinTalentTrack**[R1].

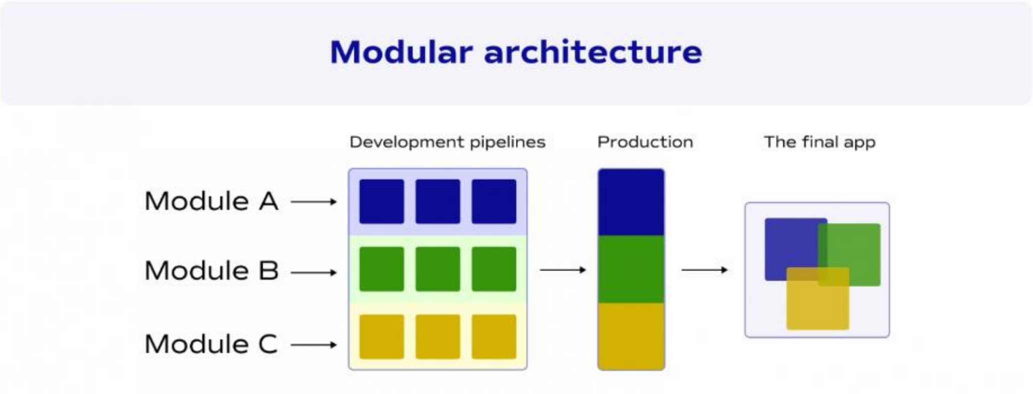


Figure 1

2.2.2 Frameworks Considered for Modular Architecture

In our research, we evaluated several of the most popular front-end frameworks available in 2024 to determine the best fit for **CurtinTalentTrack** modular architecture, including:

- **React (Version 8.2.0)**
- **Angular (Version 16.0.0)**
- **Vue.js (Version 3.3.4)**
- **Ember.js (Version 4.0.3)**
- **Svelte (Version 4.11.1)**

Each of these frameworks and libraries emphasizes modularity, reusability, and maintainability, which are essential qualities for the evolving needs of **CurtinTalentTrack**[R1].

2.3 Evaluating Front-End Frameworks

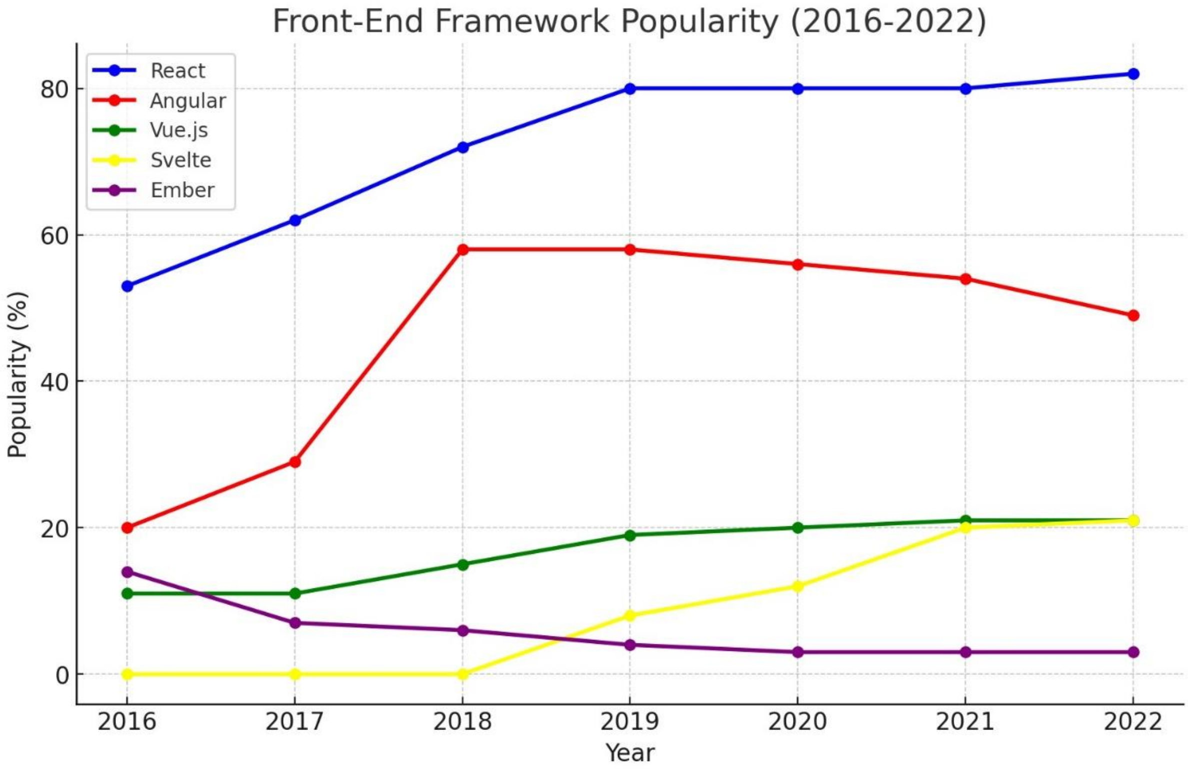


Figure 2

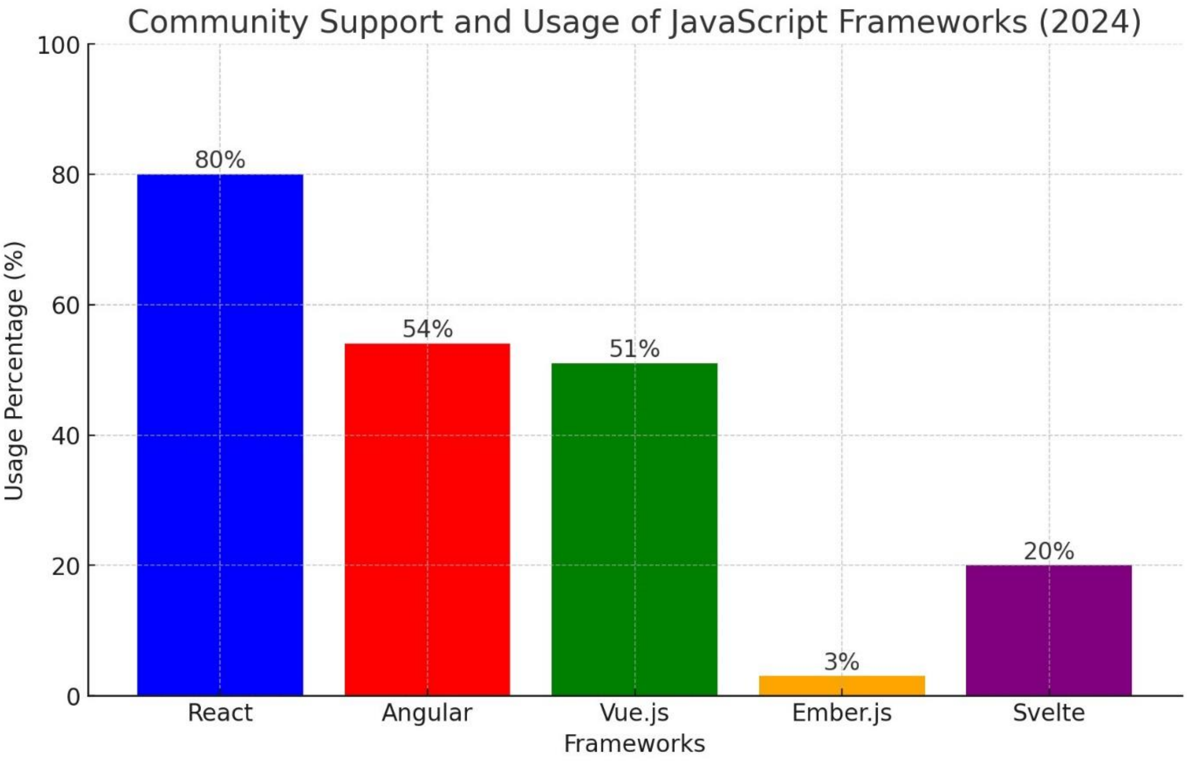


Figure 3

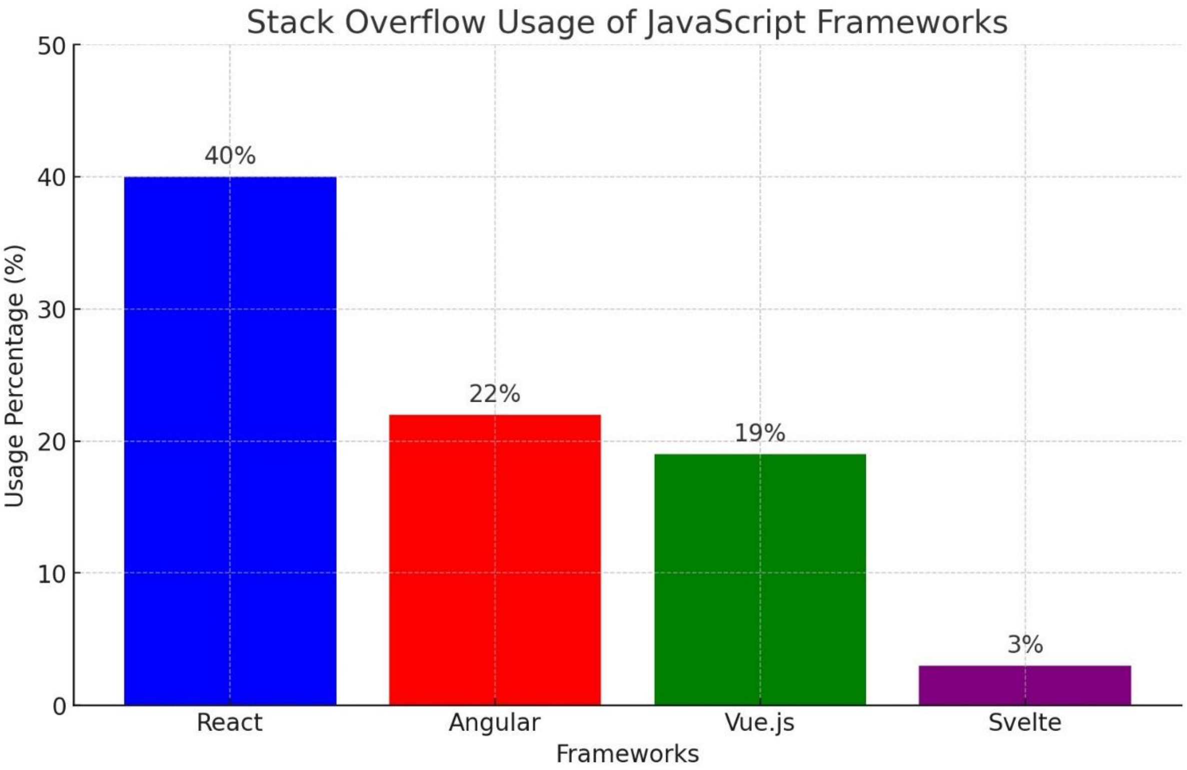


Figure 4

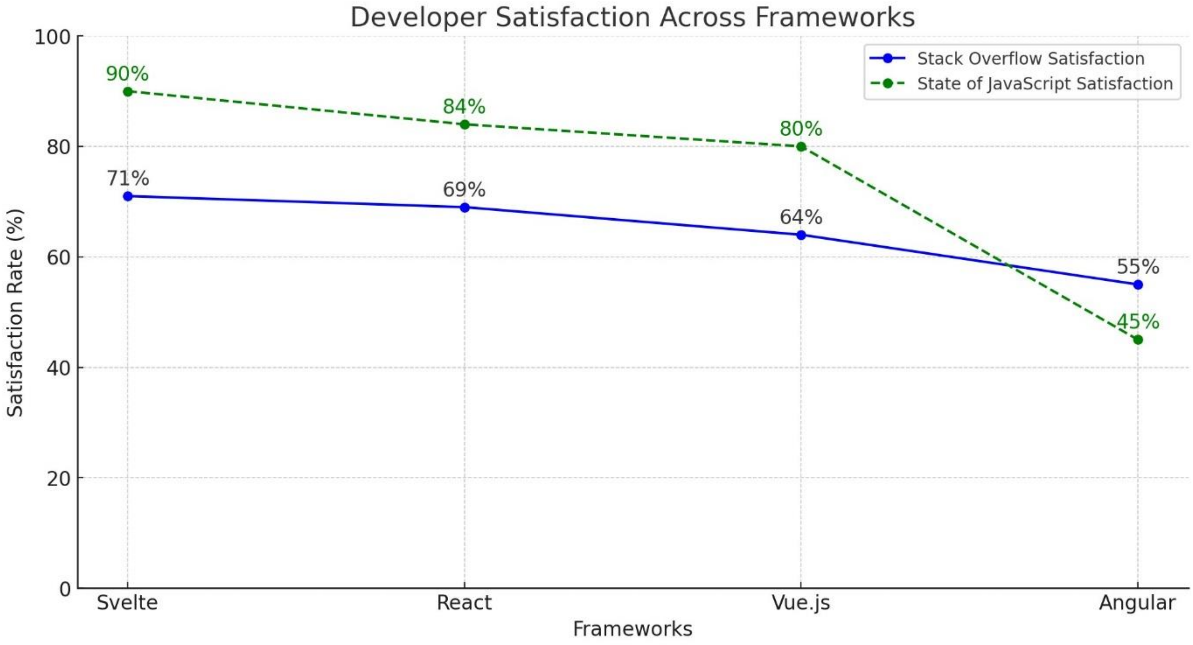


Figure 5

2.3.1 Strengths and Weaknesses of Leading Technologies

2.3.1.1 React

Strengths:

- **Component-Based Architecture:** this improves the reusability and modularity, making it easier to maintain and expand our application with the evolving needs.
- **Virtual DOM:** this helps us to improve the performance by updating only the parts of the UI that have changed.
- **Extensive Ecosystem: React** got a real large community offering a wide range of libraries and support and tools to extend functionality.
- **Flexibility:** This helps us to very easily integrate and connect with other libraries and frameworks, providing versatility in development.
- **Reusable Components:** Encourages efficient code reuse, streamlining development across multiple projects.
- **User-Friendly Learning Curve:** It is Easy for developers to learn and adopt with little amount of effort.
- **High Performance:** Ensures fast and responsive applications.
- **Redux:** Provides a predictable state management solution, which going to simplify the task of handling complex states

Weaknesses:

- **Rapidly Changing:** React goes through frequent updates and changes so it is kind of challenging to keep up with the latest technology and best practices
- **Not a Complete Framework:** So sometimes we going to require additional tools and libraries to develop a well-structured platform

2.3.1.2. Angular

Strengths:

- **Comprehensive Framework:** Offers us a full solution with many built-in tools and reduce the need to depend on external libraries.
- **Two-Way Data Binding:** this Simplifies the alignment and extreme dependencies between the model and view, making data management a lot easy.
- **Strong Community Support:** Supported by Google which is a leading tech giant so no wonder that we got a lot of documentation and a wealth of resources.
- **TypeScript:** this helps us to Enhances the code quality and safety, making the development much stronger.

Weaknesses:

- **Steep Learning Curve:** The framework is kind of complicated, especially for beginner like us like us.
- **Performance Overhead:** got a lot of features that can really make the final code a lot more complicated than what it is supposed to be, so it makes the process a lot more complicated than needed so it is not a good feature for a beginner friendly development process.

2.3.1.3 Vue.js

Strengths:

- **Lightweight and Fast:** Better for the platforms where the speed is the focus, because its light weight and simplicity
- **Easy to Learn:** The syntax is not complicated, it is really easy to understand even for a beginner front end developer like you and me.
- **Flexible Integration:** Can be gradually added to existing projects.so it is really easy to integrate and expand a project
- **Component-Based Architecture:** Like **React**, it encourages reusing code, so it really makes the maintenance easy and help to maintain the consistence.

Weaknesses:

- **Smaller Community:** Even though it is growing the community and eco system is not large and established like **React** and angular so, it is kind of a pull back
- **Dependency on Creator:** The framework kind of depends on the creator of it so it can cause a lot of inconsistencies and problems in the long run.

2.3.1.4 Svelte

Strengths:

- **Exceptional Performance:** helps to compile the code into JavaScript kinds of fast when compared so it is really an advantage.
- **Simplified Syntax:** simplicity in the code and the syntax makes it easy and enjoyable to work on.
- **No Virtual DOM:** increase the speed because we can get rid of the extra DOM layer

Weaknesses:

- **Smaller Ecosystem:** Even though it is growing the community and eco system is not large and established like **React** and angular so, it is kind of a pull back
- **Less Mature:** The framework is kind of new to the field, so the community is still growing so compared to the other frame works like **React** it is in the beginning stage, so it is kind of not easy when we needed support in the development and maintenance in the future

2.3.1.5 Ember.js

Strengths:

- **Convention Over Configuration:** Really simplifies the development because of its best practices.
- **Robust Tooling:** It really got some great tools like Ember CLI and a powerful routing system that help us to work a lot faster
- **Stability:** This framework prioritize stability, so it is really a good choice for complex projects.

Weaknesses:

- **Steep Learning Curve:** The framework is kind of hard and complex to learn specially for beginners like me.
- **Smaller Community:** Even though it is growing the community and eco system is not large and established like **React** and angular so, it is kind of a pull back.

So, it is clear after analysing the above options, **React** emerges as the strongest framework with many benefits including balance of performance, flexibility, and easy to use which we really need in developing the **CurtinTalentTrack**. The extensive ecosystem, supported by a large community than any other framework, ensures that our project will have the support and resources needed for our long-term success. While other frameworks like Angular, Vue.js, Svelte, and Ember.js each have their strengths, **React's** combination of features and developer-friendly environment makes it the ideal and the very best choice for our project [R2].

2.4 Why React is the Ideal Choice for CurtinTalentTrack

So, when we were looking to build a solid and dynamic platform like CurtinTalentTrack, selecting the right front-end technology was very important. React (Version 8.2.0**)** will be the way to go since it is popular and at the same time, highly universal with unmatched ultimate performance along Developer-friendly ecosystem. Reason behind React be suitable for CurtinTalentTrack

2.4.1 Component-Based Architecture

- **Reusable Components:** Component-based architecture the component-based nature of React, allows you to create reusable UI components that can be used in different parts of the application. This is quite beneficial, especially for CurtinTalentTrack where various features such as dashboards, forms and content modules (such public speaking module or writing exercises) can be modularized & re-utilised to save time in development process while ensuring uniformity [R3].
- **Maintainability:** React helps in creating complex user interfaces by dividing an application UI into multiple smaller, independent pieces which don't depend on each other ensuring the same [R3].

2.4.2 Virtual DOM for High Performance

- **Efficient Updates:** React uses what it calls the Virtual DOM to keep track of and efficiently re-render updates. In case of CurtinTalentTrack, the systems need to provide quick and smooth UI updates as users would interact a lot with the platform (Submit content/Feedback) [R3].
- **Smooth User Experience:** Very fast because it only updates parts of the DOM where necessary to guaranty us that there's plenty interactivity with the platform and synchronous work [R3].

2.4.3 Strong Ecosystem and Community Support

- **Rich Ecosystem:** React has a big ecosystem of libraries and tools that can be used to improve your project, CurtinTalentTrack is no different. Redux for state management, React Router for navigation and libraries such as Material-UI that offer prebuilt. [R4, R5]
- **Community Support:** React is very popular so no wonder it got a large community, which means a lot of resources, tutorials are available. This is really helpful solve problems quickly and efficiently [R4, R5].

2.4.4 Flexibility and Integration

- **Seamless Integration:** React, when combined with Next.js, creates a powerful interoperability framework. This synergy is further enhanced by Next.js, which extends React's capabilities by offering features like Server-Side Rendering (SSR) and Static Site Generation (SSG). These tools are particularly valuable for CurtinTalentTrack, as they allow us to leverage React's dynamic UI capabilities while benefiting from Next.js's advanced features. Additionally, using a full stack that includes Redis and Node.js optimizes server-side operations, providing flexibility in deployment options. [R6].

- **Scalability:** React is very flexible and allows to improve the platform and to scale efficiently as the project grows with the time whether by adding new features or handling more users [R6].

2.4.5 Developer Experience

- **JSX Syntax:** JSX syntax allows developers to write UI components declaratively mixing HTML with the familiar script (JavaScript). This leads to a more intuitive and less error-prone development process [R7].

- **Hot Reloading:** React supports hot reloading which enables the developers to see the live changes in real-time without having to refresh whole application. It also speeds up the development and increases productivity[R8].

2.4.6 Rich User Interface

- **Dynamic UI:** With the dynamic and interactive UIs that React is fit to build, this was a must-have as CurtinTalentTrack involves quite of number of features where users will be interacting with (Video recording + Quiz taking + Content submission) etc. Since the interactive features of this application require careful UI state and render updates management, React is a perfect fit for my solution[R9].

2.4.7 Future-Ready

- **Continuous Updates:** As I mentioned earlier, React is supported by Facebook and has an active roadmap for future features. The use API and React Server Components in the recent enhancements like React 19 improve performance, are friendlier to resources — but also enable more advanced server-side functionality. These will be particularly useful to the CurtinTalentTrack project, as it endeavours to incorporate new AI advances in future. Implementing machine learning features within React will enable the platform to safely deal with server driven, dynamic tasks and being ahead of time. CurtinTalentTrack will continue to evolve with new tech, without actuating a stop on the street through constant effort spent swipe carding CNT but this is all because of what underpins it and that feels good. [R10].

As we saw above, the recipe of performance, flexibility and ease of use in React, made it ideal for building a CurtinTalentTrack. We believe that its ability to make super-fast, dynamic and reliable form of UI which falls exactly with respect out our cause in empowering the communication between local Curtin friends.

2.5 React vs. Others for CurtinTalentTrack

1.5.1React vs. Angular

React has easier learning curve than the accelerate framework of angular. Because **React** is purely the view layer, it's easy to pick up whereas Agular's everything can be overly complex especially for beginners. **CurtinTalentTrack** is an ongoing project, and **React** being component-based makes it modular & scalable to adjust according the evolving needs. **React** has its own Virtual DOM that makes sure you get performance efficient UI updates[R11].

2.5.2 React vs. Vue.js

Although Vue.js is more suitable for beginners, **React** takes the lead in terms of a big community and ecosystem. **React** is a product of Meta (formerly Facebook) which means you are in good hands i.e. stabilized and help will be available for long term and this is absolutely essential for a project like **CurtinTalentTrack**, it has to be built on such solid base. **React** has high flexibility that allows it to easily integrate with different tools and have a project adapted when necessary [R11].

2.5.3 React vs. Svelte

Svelte's performance is impressive, but **React**'s mature ecosystem and widespread use make it more suitable for **CurtinTalentTrack**. **React**'s familiarity in the developer community ensures that beginners can find ample resources and support. This is essential for a project that requires sustainable growth and maintenance over time [R12].

2.5.4 React vs. Ember.Js

React's simpler, more flexible approach makes it easier for beginners compared to Ember.js's convention-over-configuration style. React's vast community and resources offer better support, which is critical for **CurtinTalentTrack** success. React's modularity and ease of integration also make it more adaptable to the project's dynamic requirements [R12].

2.6 Evaluating React Based on Key Metrics

2.6.1 Functionality:

- **Does it do what your project needs?**
    - **Component-Based Architecture**: **React**, framework makes it particularly well-suited to this kind of modular requirement as with **CurtinTalentTrack** we have focus mostly on dashboards, interactive quizzes and practice task creation which a typical type of Use Cases that benefit from UI component reusability. Every part (public speaking, written communication etc) can be separate components and more maintainable and scalable.
    - **State Management**: **React** has good options for state management such as hooks (useState, useEffect), and if you need a more advanced way of managing states like when doing an API call to track which user have answered something or submitting some feedback we would be able to use Redux that is specially centralized store -> this will manage all the dynamic process in our course app including keeping up with users progress / any submissions made by user / personal dashboards.
    - **Interactivity**: **React** can easily manage user inputs, events, and live updates so as for recording/uploading video or answering a quiz question/live feedback by assessors. Fast UI Update by using Virtual DOM.

2.6.2 Documentation and Support:

- **Is there good help available if you need it?**

  - **Great Documentation**: **React** has a superb set of well-maintained documentation catered to an easy-to-follow guide for beginners and also seasoned pros. It covers all from basic concept to advance patterns and best practices.

  - **Massive Community and Eco System**: **React** has a massive developer community. There are a lot of tutorials, forums like Stack Overflow or Reddit and official resources where we can get help when was needed. Moreover, open-source libraries and tools such as **React** Router or Material-UI can help to create development even easier.

  - **Corporate Support**: Meta, the company behind **React** provides a robust backing for support and ongoing development. Problems are fixed fast and additional features come highly documented.

2.6.3 Liveliness:

- **Is the technology still being updated and used?**

  - **Regularly Enhanced**: **React** is being regularly updated to offer better user experience, which ensures that it will remain one of the top front-end frameworks. It evolves continuously because it is updated and maintained by an open-source community.

  - **Adoption: React** is in wide use at large companies, such as Facebook (of course), Instagram and Airbnb but also small startups. Its extensive ecosystem has plenty of third-party libraries, frameworks like Next.js etc centres around a Node.js server-side runtime which is pretty fast (considering V8 engine)and with tools like Gatsby also in the mix, React provides a strong foundation that ensures its relevance and viability well into the future.

  - **Future prospects**: There is a good demand for **React** developers, which assures that you will be able to stick around in the market. **React** remains a major player in front-end development, even as new frameworks crop up all the time.

2.7 Security Considerations for CurtinTalentTrack

When creating a platform like CurtinTalentTrack, which will receive user content and contain sensitive information, it is essential that the application security be taken very seriously. While React has numerous security features in-built, there are some context-specific risks as well that could create an obstacle. In this blog, we will delve into some of the security concerns and how it pertains to CurtinTalentTrack.

2.7.1 Cross-Site Scripting (XSS)

- **Application to CurtinTalentTrack:** The platform that I am designing, Culley-Being you has user generated content such as written communication tasks and public speaking videos. If user inputs, such as text in written tasks for example, are not correctly sanitized then an attacker can escape this and enter some malicious scripts that will run when a certain other user or evaluator sees the content.

- **React Front-End (Update Number):** Due to React escaping strings when they are rendered you get some protection by default. But as with dangerously set inner HTML, it comes at the risks of leaving yourself open to XSS. To prevent this, you must sanitize all incoming HTML content before displaying it in the application using libraries like DOM Purify [R13].

2.7.2 Component-Based Architecture Risks

- **What this means for CurtinTalentTrack**: Modularisation — Many components are reusing with different feature as **CurtinTalentTrack** have modular architecture. Max: Still, the dependency hell in that space is a threat as always — using third-party libraries or components pose risks of introducing vulnerabilities not vetted up front.

- **React Front-End Application:** The biggest weakness in the third-party React component that we can use to develop, on top of helping us, is also our own security hole where they are hacked. To help mitigate these risks, those dependencies should be audited regularly to ensure they are not dangerously out of date [R14].

2.7.3 State Management Vulnerabilities

- **Relevance to CurtinTalentTrack:** The platform will probably interact with sensitive user data profiling metrics, and a session may choose to cache this in the state for transient reasoning. Unfortunate that unsecured data could be accessed by rampant unauthorized parties.

- **React Front-End Application:** One must be cautious when working with sensitive data whilst using tools like Redux or the React Context API for state management. You should absolutely NOT save any sensitive data in this client-side state or local storage. Encrypt data only when necessary and store it in a proper secure way on the server side [R14].

2.7.4 Cross-Site Request Forgery (CSRF)

- **Relevance to CurtinTalentTrack:** For example, if you allow users or evaluators to interact with the platform (such as submitting feedback, updating profiles etc.) these actions could all be vulnerable to CSRF attacks IF the application does not authenticate these requests properly at all.
- **React Front-End Application:** All the forms and API requests should have Anti-CSRF tokens implemented to prevent CSRF. This makes sure that only transactions by valid, authenticated requests are made thereby decreasing the possibility of malicious action to be performed in user's context [R14].

2.8 Security Best Practices

Security management: Given the sensitive nature of user generated content and interactions, effectively managing security is critical to develop a platform like CurtinTalentTrack. So, let us discuss the Five Must Security Best Practices and Implementations in your React Application:

1. **Sanitize User Inputs**
   - Dangerously Set Inner HTML can potentially open up XSS attacks if you are not extremely careful. First and foremost, always sanitise user inputs using libraries like DOM Purify before rendering them. You will likely had to do this before, especially with any user-generated content; the communication tasks or comments haven't gone away.
   - **Source:** Snyk and Dev.to
2. **Implement Content Security Policy (CSP)**
   - Control from what origins scripts, styles and other resources can be loaded with CSP headers. This leads to the execution of harmful scripts which might be injected into your app.
   - **Source:** Dev.to
3. **Prevent Cross-Site Request Forgery (CSRF)**
   - Use anti-CSRF tokens in your forms and API requests to protect your application from CSRF attacks. Also please note that you can use same site cookies to avoid cookie headers being sent along with cross-site requests.
   - **Source:** JsJungle and Dev.to
4. **Secure API Communications**
   - Be sure to always move data between the client and server in an encrypted manner using HTTPs. Use token-based authentication (e.g., JWTs) to access APIs securely and make sure that the tokens are transmitted over a secure channel and have short expiration periods which reduce risks.
   - **Source:** JsJungle

5. **Keep Dependencies Up-to-Date**

    - Unpatched dependencies can lead to security issues. Testing tools like npm audit, or yarn audit can be used to check your project for vulnerabilities on a regular basis. Default integration helps us patch security vulnerabilities in third-party libraries as quickly as possible

    - **Source:** Snyk and Relevant Software

2.9 References

References:

[R1] - ELITEX - Front-End Architecture: In-Depth Analysis

[R2] - BrilWorks - Best Front-End Frameworks

[R3] - BitCot - React vs Angular
[R4] - Dev.to - The React Ecosystem in 2024 (Part 1)
[R5] - I Love ReactJS - 2024 Year in Review: The React Ecosystem
[R6] - Progress Blog - Sitefinity React Next.js Renderer Going Live October 2024
[R7] - FreeCodeCamp - JSX in React: Introduction
[R8] - Dev.to - The Magic of Hot Reloading in React
[R9] - Masai School - Frontend Web Development with ReactJS
[R10] - React Blog - React 19 and Beyond
[R11] - BrowserStack - Angular vs React vs Vue
[R12] - JavaScript Conference - Svelte vs Angular vs React vs Vue
[R13] - Snyk - 10 React Security Best Practices
[R14] - Dev.to - Securing Your React Applications: Best Practices and Strategies

3. Middleware Technology

This section discusses the rationale for choosing **Express.js (Version 4.18.2)** in detail as the best middleware technology for our CurtinTalentTrack project. The decision to go for Express.js was chosen after a careful analysis of different middleware technologies where criterions like functionality, popularity, simplicity, flexibility, and compatibility were taken into consideration. It also highlights comparisons with other middleware technologies and the reasoning behind choosing Express.js, with the precise version numbers of all applications covered for accuracy.

3.1 Overview

Express.js (Version 4.18 2) is an open-sourced Web application framework that has minimal design and great flexibility; It provides a powerful solution to create Web and Mobile applications. It is meant to assist people in the creation of web apps and APIs much easier by offering an abstracted API that works well with Node.js. Designed and created by TJ Holowaychuk and is now maintained by the Node.js Foundation, Express.js also has the responsibility of handling the middleware stack and all the routing that takes place in the application.

3.2 Technology Options Researched

There are various middleware technologies that we took into account based on the goals and requirements of CurtinTalentTrack. The following technologies were evaluated:

- **Express.js (Version 4.18.2)**: A basic and versatile open-source, fast, Node.js web application with a huge number of features for web and mobile apps. It is well known for its simplicity in usage, favourable community support, and its ability to expand on these aspects.
- **Koa.js (Version 2.14.2)**: A new generation web framework developed by the team behind Express.js, is more modular and supports async functions, but is comparatively less developed than Python.
- **Django (Version 4.2.5)**: A Python web application framework that proposes the usage of high-level synchronous components and many features to accelerate the development process maintaining the applications clean, reusable and pragmatic. Django has many features, but these may not be relevant if a project is being done using Node.js and JavaScript.
- **Apache Camel (Version 3.20.4)**: An open-source integration framework focused on routing and mediation rules, suitable for enterprise-level integration but less ideal for web application development in a JavaScript environment.

3.3 Architecture

Express.js adopts a level of middleware where each function deals with a request before passing it to the other function in the stack. This extensible basic structure can be extended with middleware functions like logging or body parsing or authentication or routing, depending on the needs of the application.

3.4 Advantages

The decision to adopt Express.js as the preferred middleware technology for **CurtinTalentTrack** is grounded in several pivotal factors:

- **Lightweight Architecture**: Express.js is often praised for the fact that it has an unobtrusive style of programming, which makes it very basic to read or comprehend, modify and find mistakes in. It also makes it possible for the developers to work on the essential aspects of a project without having to worry about complications that are not central to the project. By being lightweight, it also supports low request-response times, an aspect that is important in enhancing user interaction.
- **Ease of Learning**: Based on JavaScript, Express.js. takes advantage of developer's familiarity with this popular computer languages. The language has a plain structure, which loosely requires time to learn, thus enabling the developers to focus on adding end points of different functionalities of the CurtinTalentTrack system.
- **Middleware Support and Routing**: Express.js shines in working with middleware which can be used for a request validation, authentication, and logging, etc. Its effective routing mechanism provides a suitable way to handle HTTP requests, which is a core feature for the modularity of our application.
- **Integration with Databases**: Express.js is very effective in integrating with various databases, especially the MongoDB, which we are employing in our project. This makes the results easier to manipulate and retrieve and is essential when running the user information and the content of the platform.
- **Scalability**: The fact that Express.js is non-blocking and uses an event-driven model makes sure that the application remains functional and can accommodate many users at the same time. This scalability is important especially as the platform expands and more people join the platform.

3.5 Mitigation Strategies

While Express.js offers numerous advantages, it is essential to address some potential limitations:

- **Lack of Structure and Convention**: Express.js has a lot of freedom of choice but does not have a clear structure, which can cause disorder in programming. To counter this, we will implement the Model-View-Controller (MVC) design pattern and conform to coding standards.

- **Overhead Performance**: To optimize performance, we will implement best practices such as profiling to identify bottlenecks and using caching mechanisms to reduce server load. Deploying Express.js behind a reverse proxy can further enhance performance.

- **Limited Built-In Features**: Although Express.js is lightweight, it doesn't include many features out-of-the-box. This can be mitigated by leveraging npm packages and third-party middleware to extend the functionality of the framework as needed.

- **Lack of Strong Typing**: To improve code quality, especially in larger applications, we will integrate TypeScript (Version 5.2.0) with Express.js. TypeScript provides compile-time type checking, which helps in early detection of errors and enhances the overall maintainability of the codebase.

3.6 Comparison

3.6.1 Koa.js (Version 2.14.2)

Koa.js is a lightweight and modular web framework created by the same team behind Express.js. It uses async functions to streamline asynchronous operations, providing greater control over the flow of middleware. Koa.js is designed to be more flexible than Express.js, but this comes at the cost of a smaller ecosystem and fewer third-party tools and middleware. For our project, which requires a robust and well-supported framework, Express.js is preferred due to its extensive community support and comprehensive documentation.

3.6.2 Django (Version 4.2.5)

Django is a Python based framework that provides tools for web development which enhances the speed of development, and the applications are designed to be clean and pragmatic. It provides many out-of-the-box functionalities, such as Facility to manage control panel, ORM, and to handle Forms. But Django framework is monolithic, and it is built for the Python language, which is less optimal when you have most of our project in JavaScript. In addition, to the points mentioned above, integrating Django with a Node.js environment would only complicate things in this case as it would make Express.js is a little more appropriate for now for our requirements.

3.6.3 Apache Camel (Version 3.20.4)

Apache camel is an open-source integration framework which largely deals with routing and mediation rules and is based on Java DSL. Although better for large integration jobs, Apache Camel is not well suited to be used for web application development and not at all for a JavaScript environment as we have here. This section is going to describe the challenges that come with integrating Apache Camel with Node.js and JavaScript, plus its emphasis on enterprise connectivity over the web construction leads to Express.js. so is a more suitable choice for our project.

3.7 Conclusion

After choosing and comparing different kinds of middleware technologies, **Express.js (Version 4.18.2)** was determined to be the most appropriate middleware framework for CurtinTalentTrack. The main advantages being its lightweight architecture, lower learning curve, ample middleware support and powerful routing capabilities correlate with the end goal of our project. Also, compatibility with our selected database to integrate it into our program and the capability to scale up the program can make Express.js the ideal choice for creating a dynamic and responsive platform for the Curtin community.

3.8 Best Practices

For further insights into middleware technologies and best practices, the following resources are recommended:

1. Django Middleware
2. Guide to Django Middleware
3. Express.js Middleware: A Complete Guide
4. Python Best Practices for Better Code
5. Python Best Practices, Tips, and Tricks
6. Express.js Best Practices
7. Ruby on Rails Best Practices
8. Rails Best Practices

4. Database Technology

3.1 Introduction

In the CurtinTalentTrack project it's crucial to have a user friendly, accurate and efficient database system to handle large amounts of information that is processed and used on the platform. Since CurtinTalentTrack shall facilitate improvement in communication skills within the Curtin Community, it will process multiple types of data; profiles of the users, multimedia contents, written communication tasks and results of the evaluations. The database is going to be incorporated as a core of the platform under consideration, because this data is to be accumulated and stored in such a manner that would allow for its fast access, proper storage, and efficient retrieval.

4.2 Technology

4.2.1 Options Researched

- Three database technologies were evaluated to determine the best fit for the project

1. MongoDB: An acronym for ‘not only structured query language’, a type of database that is flexible and scalable and stores data in a document form.

2. MySQL: It is a widely used Relational DBMS with very good data consistency.

3. PostgreSQL: Very well-developed Relational Database with basic and important calculations and inquiries for complex questions and answers.

4.2.2 Option Selected

As a result of performing a comprehensive comparison and assessing consequent options, MongoDB was adopted to be the database remedy for the CurtinTalentTrack Project.

4.3 Overview of the Option Selected.

MongoDB is one of the most widely used databases belonging to NoSQL movement and known for flexibility, opportunity for growth and simplicity. It records data in a document-oriented manner through the Binary JSON (BSON) to address the issues of structural relationship complexity. MongoDB’s document-based model with no fixed schema specifically defines it as flexible, and that is exactly what is required for contemporary applications to deal with various and constantly evolving data types.

Of course, it is effective with the new version 6. 0, MongoDB presents improved characteristics including high-performance upgraded indexing options, and better security measures. It also supports extensive data analysis capabilities, capping and accessing features making it ideal for large scale applications which require high availability and real time computations. MongoDB’s distribution also makes it easy to incorporate data in a horizontal way – meaning, vertically, the system can just add more servers, which makes the system highly scalable for projects that expect to grow and processes big data sets.

4.4 Justifications

The decision to go with MongoDB was based on several factors:

1. Flexibility: MongoDB does not have a fixed database schema, which is relevant to the variety of multimedia resources that will be used by the platform, user profiles, evaluation records, and feedback entries.

2. Scalability: MongoDB also fits into the application's needs well due to its distributed nature and horizontal scalability, needed to accommodate the increasing amount of user interactions and resources.

3. Performance: MongoDB is preferred for high write and high read applications such as real-time communication applications, applications that store user generated content and applications that require rapid data access.

4. Ease of Integration: MongoDB also fits very well with the chosen technology stack for the project combined especially with Node.js for backend development.

5. High Availability: MongoDB has a replication technique, and it also has automatic failover that enables CurtinnTalentTrack to run even when the server is down or being recharged. This is important if a platform is to provide always active resources to the users especially during the peak usage time.

6. Strong Community Support and Continuous Development: It has high popularity, and it has good community, which would mean that the industry is full of best practices, best tools and best resources. Also, the continual development and the community support mean that CurtinnTalentTrack can take advantage of the new advancements and keep in touch with the best practices for a familiar industry.

4.5 Security

Curtin Talent Track ensures that users' data is highly secured when they are using the platform.

The following security measures will be implemented:

1. Encryption: While operating and while on the move, the data will be encrypted with the help of Transport Layer Security (TLS) and secure sockets layer (SSL).

2. Access Control: Implementing access control means that restricted data will only be accessible and more so can be modified by only those with authority to do so.

4.6 Best Practices

To maintain optimal performance and data integrity, the following best practices will be followed:

1. Use Indexes Wisely: Make sure that you are defining fields that are most subjected to search creating indexes on them to enhance the search. This assistance to MongoDB in its ability to access the data quickly when working with big data.

2. Implement Data Validation: MongoDB's schema validation allows user to define restrictions on the documents, including the structure and the values of the fields. This way, any data that is entered into the database is valid thus minimizing errors and inconsistency.

3. Regular Backups: It is essential to work with backups of your MongoDB database because it allows you to minimize the loss of data in case of an emergency. This way you can easily recover from failure or disasters calamitous to your business.

4.7 Conclusion

Upon careful analysis of all the feasible solutions and with the right consideration for the CurtinTalentTrack project, MongoDB was chosen as the database solution. The support of the unstructured data and great variability, and scalability also meet the requirements of the project, as it provides an ability of the storage and management of the rather different kinds of the data. MongoDB high performance with incredibly fast querying and indexing will allow the platform to scale to the extent of projected users' and data's increase without any decline in velocity and steadiness. Furthermore, the security options of MongoDB incorporate reliable and secure encryptions and access controls essential in offering appropriate shield to other important details concerning the users thus making it the most suitable choice to use in CurtinTalentTrack project.

4.8 References

More on these Links that will take into consideration,

1. MongoDB Schema Design Best Practices
2. MongoDB Configuration
3. MongoDB Best Practices

Securing your MongoDB database essential best practices

5. Backend Technology

5.1 Introduction

The CurtinTalentTrack a web application aimed at improving communication skills within the Curtin Community is fundamentally based on the backend technology. Backend deals with data and function as well as user control; though the frontend contains multiple aspects such as practice, interviewing, written communication, quizzes, critical thinking, and events, backend focuses on supporting and handling these all factors including but not limited to user authentication and security, data management, and scalability.

In choosing the backend technology one has to get it right to meet the efficiency, reliability and ability to scale of the app goals set. This entails choosing the best server-side framework, the right and reliable DBMS and optimal cloud hosting solutions. Further, the backend should enable real-time interaction, secure video and data processing, smooth user experience and compatibility with other systems of the university.

This focuses on the selected backend this technology for CurtinTalentTrack and assesses and analyses the appropriateness of the selected technology in delivering reliable, structured, and cost-effective environment for the various users of the postgraduate talent tracking system; the users encompass students, academic and non-academic staff among others.

5.2 Technology

5.2.1 Options Researched

- Python
- Node.js
- Express.js
- Django
- .Net with C#

5.2.2 Option Selected

Comparing different backend technologies and, having discussed with our server side, the Front-end side we decided to Node. JS to be used as the backend framework of CurtinTalentTrack. Node.js is a fast and scalable cross platform runtime which uses JavaScript to create server-side applications. Since the server is non-blocking, this makes it very efficient in the handling of multiple connections at once, an aspect that is quite relevant for an application such as CurtinTalentTrack as it deals with real time interactions, video uploads and data processing for a significantly large number of users.

5.2.3 Justification

Performance and Scalability

Compared to the other options Node.js has emerged as preferable choice since it does not involve the use of extensive third-party libraries and databases. Node.js employs non-synchronous, event driven execution which made it possible for it to support and execute multiple requests concurrently but in greater efficiency. This means it is well suited to CurtinTalentTrack as this needs to process data in real time, perform video uploads and work with a significant number of users at one point in time.

Single Language for Full-Stack Development

Other options that can be used such as Python or Django Through the use of JavaScript on both the frontend as well as the backend, Node.js allows for the creation of a single development environment for both the frontend and the backend, which in turn helps to reduce time spent as well as help the two teams collaborate while they work. This saves time in development and makes all the code base to be similar in a big project.

Rich Ecosystem and Support of Community

Node.js also has many libraries and modules which can be installed through the node package manager that helps in the development process by providing functionalities for almost everything ranging from authentication, video processing and data handling. There is also the consistently strong support and involvement of the community to the project such as through updates and security patches.

Cost Efficiency

Node.js is free from licensing fees as it is an open source, thus cutting the general expenditure of the activity. Also, resource management is improved enabling efficient cutting down of cost of part from servers especially when maintaining and expanding the platform.

5.3 Security

Built-in Security Features

Node.js has inbuilt modules like the crypto module that can be used for instance in data encryption and also implementing secure hash algorithms. Such features are crucial when it comes to preserving information that should not be accessed by other people such as user credentials, personal details of users and content of messages within CurtinTalentTrack.

Regular Security Updates

Node.js community contributes towards the framework by frequently updating with security features and fixes. This assist in early identification and correction of vulnerability issues and sustain a secure app environment.

Secure Session Management

Node.js provides safe session management through utilities that can be found among others in *"express-session*". These libraries offer such functionalities as session expiration, session storage, and secure cookie management that are very useful in preserving the contents of the session, especially in an application that may involve sensitive activities as may be rated speaking delivery practices, interview rehearsals among others.

Monitoring Tools

Node.js offers options with a variety of monitoring and logging mechanisms that help get information about a breach and address it as soon as possible, including Winston and Bunyan. These tools can monitor, and we searched abnormal activities patterns and this can go long way in protecting the system by notifying the administrators of a potential threat.

5.4 Best Practices

1. https://www.tatvasoft.com/blog/node-js-best-practices/
2. https://www.tutorialspoint.com/expressjs/expressjs_best_practices.htm
3. https://www.codingdojo.com/blog/python-best-practices

6.Intergration and Compatibility

6.1 Technology Stack Overview

The foundation for the CurtinTalentTrack application has been chosen to be built with The MERN stack, which includes React, Express.js, Node.js, and MongoDB. These all add up to a strong services platform that incorporates JavaScript from the ground on up the stack: frontend, middleware, backend and database. This consistency allows a development to be more streamlined, provides us with little learning curve, and promotes greater ease of communication between different parts of the application.

Each technology of the MERN plays a huge roll in the overall architecture of CurtinTalentTrack.

6.2 Integration Consideration

Good integration of these technologies is a must to successfully deploy CurtinTalentTrack to its users. The following guidelines will be followed.

6.2.1 API Design

The Frontend (React) & The Backend (Node.js); RESTful APIs are the bridges that allow these 2 major parts of your application talk to each other js). Express. These APIs are to be created and managed using js framework. It had to satisfy the requirements of moving data between the client and server, with transparent endpoints for modifying records.

- **API Versioning:** With an eye on the future, versioning will be in place to regulate potential changes in the API. This in itself paves the way for having multiple API versions to coexist, which ensures backward compatibility without breaking changes between REST endpoints as the app matures.

- **Error Handling:** We will be implementing error handling right into the API design where we provide clear feedback to the client and ensure errors are well logged so that they can be easily tracked on server side.

6.2.2 Dataflow and Integration

Data will be passed from Frontend, Middleware, Backend to Database in JSON format. This is even more helpful because MongoDB being a NoSQL database natively supporting JSON-like documents.

- **State Management:** React uses any state management too Redux or Context API on the front end to manage states of application. As a result the UI gets updated as and when updates comes from backend, which gives a good user experience.

- **Data Synchronization:** In a distributed system, we need to make sure data consistency which could be maintained using the technique like versioning of documentation, that will be used in MongoDB. This way it allows the conflicts that may arise from simultaneous data updates of more than one user.

6.2.3 CORS Handeling

Since frontend and backend can be hosted under different domains or subdomains, Cross-Origin Resource Sharing (CORS) will be configured for the client to safely utilize the server. Express.js will give middleware to manage CORS settings, enabling secure cross-domain requests while preventing unauthorized access.

6.2.4 Session Management

User sessions will be managed on the server side using Express.js. Cookies and JWTs (JSON Web Tokens) will be used for maintaining session state and managing user authentication and authorization. JWTs, stored in HTTP-only cookies, provide a secure mechanism for session management, reducing the risk of cross-site scripting (XSS) attacks.

6.3 Compatibility Consideration

It is essential to make sure all the different components of MERN stack are compatible with each other, so that CurtinTalentTrack application can run without a hitch. Will focus on below points.

- **Frontend and Middleware:** React provides a modular architecture like Express.js, in its focus on separation of concerns and the reusability rather than object-oriented js. Interact for the frontend with Express using RESTful APIs Routing & Middleware functions in express with js React components are modular, makes it easy to manage data, handle side effects and update the UI due to backend responses.

- **Middleware and Backed Compatibility:** Express js is built on Node js, which makes the integration between middleware and backend almost natural. The compatibility allows easy management HTTP request, routing and server-side logic. Additionally, the event-driven design of Node.js is particularly well-suited to handling multiple concurrent requests in a world where web applications are increasingly asynchronous.

- **Backend and Database Compatibility:** MongoDB is a NoSQL database, and usually requires NoSQL for backend development. Direct connection backend and database This compatibility ensures data operations (CRUD) Create Read Update Delete are fast and have minimal latency. The document-oriented structure of MongoDB very much fits with the JavaScript object model that keeps data manipulation simpler and less dependency on ORMs.

7.Performance and Scalability

7.1 Optimization Strategies

Several optimization techniques will be applied throughout the stack to guarantee that the CurtinTalentTrack application operates effectively in a variety of scenarios. These tactics concentrate on lowering server load, minimizing latency, and guaranteeing a responsive user experience:

7.1.1 Frontend Performance Optimization

React comes with a number of built-in techniques to improve frontend performance. Among them are:

- **Virtual DOM:** By batching updates, React's Virtual DOM reduces the amount of direct DOM manipulations. This method drastically lowers the overhead related to DOM operations, which results in quicker rendering speeds and a more responsive user interface.

- **Code Splitting:** The program can load only the JavaScript bundles required for the active view by employing code splitting. Performance is enhanced and the initial load time is decreased, particularly for large applications.

- **Lazy Loading:** Components that are not required for the initial load will be loaded slowly using a technique called lazy loading. This method significantly shortens the initial load time and enhances user experience by delaying the download of certain components until they are needed.

- **Effective State Management:** Redux and Context API, two of React's state management capabilities, will be utilized to effectively manage global state. By reducing pointless re-renders, these techniques aid in even more performance optimization.

7.1.2 Middleware and Backend Optimization

To handle numerous requests at once, Express.js and Node.js will use asynchronous, non-blocking operations. The middleware and backend will be optimized by the following strategies:

- **Load Balancing:** To disperse incoming requests among several server instances, load balancing will be used. This tactic makes sure that no single server becomes a bottleneck, which enhances the capacity of the program to manage heavy traffic.

- **Clustering:** On multi-core systems, multiple instances of the server will be created using Node.js's clustering module. By using all of the CPU cores that are available, this method improves performance and scalability of the application.

- **Caching:** In order to reduce the frequency of database queries, server-side caching will be used to store frequently accessed data in memory. We'll take into consideration programs like Redis or in-memory caching libraries.

7.1.3 Database Optimization

MongoDB will be enhanced to efficiently manage substantial datasets and elevated query traffic. Among the optimization techniques are:

- **Indexing:** To enhance query performance, MongoDB's indexing features will be utilized. Indexes cut down on the amount of time needed to get data by enabling the database to find documents fast based on the indexed fields.

- **Denormalization:** To minimize the amount of read operations needed to retrieve related data, data may occasionally be denormalized. Denormalization might result in redundant data, but it can also greatly enhance read efficiency in NoSQL databases such as MongoDB.

- **Aggregation Framework:** To handle massive datasets effectively, MongoDB's aggregation framework will be employed. Aggregations provide intricate data manipulations and calculations, allowing the program to draw conclusions from massive amounts of data.

7.2 Scalability Consideration

For the CurtinTalentTrack application to support an increasing user population over time, scalability is essential. The scalability solutions listed below will be taken into account to make sure the application can accommodate growing loads without experiencing performance issues:

- **Horizontal Scalability:** The application architecture will facilitate horizontal scaling, which will enable the addition of more servers to manage growing loads. Since load balancers and clustering make scaling the Node.js backend simple, this is especially crucial. Because it enables the application to take use of the elastic scaling properties of cloud infrastructure, horizontal scaling is an affordable method of increasing capacity.

- **Database scaling:** As the database expands, read and write speed will be enhanced by distributing data over several servers thanks to MongoDB's sharding capabilities. To guarantee redundancy in data and high availability, replica sets will also be put into place.

- **Load balancing:** To avoid any one server from becoming a bottleneck, incoming traffic will be divided among several server instances via a load balancer. This will guarantee steady performance even with high loads.

8. Security Considerations

8.1 Frontend Security

The security of the frontend is most important, especially in protecting user data and ensuring safe interactions with the backend:

- **Input Validation:** Client-side validation will be used for input validation in order to stop harmful data from being uploaded. Before submitting queries to the server, this entails confirming that the data types, formats, and ranges are valid.

- **Secure Communication:** HTTPS will be used to encrypt all communications between the frontend and backend, guaranteeing that information sent over the network is shielded against interception and manipulation.

8.2 Middleware Security

Security in the middleware focuses on protecting the API endpoints and ensuring safe data handling:

- **API Authentication and Authorization**: To secure API endpoints and guarantee that only authorized users can access particular resources, JWT (JSON Web Tokens) will be employed. Token verification and user role management will be handled via Express.js middleware methods.

- **Rate restriction:** Rate restriction, which caps the quantity of queries a user can submit in a specified amount of time, will be put in place to safeguard the API against misuse. By doing this, denial-of-service assaults will be less common.

8.3 Backend Security

Strong security measures will be implemented by the backend to safeguard private information and server resources:

- **Authorization and authentication:** User authentication will be managed by the backend, guaranteeing that only authorized users are able to carry out particular tasks. Role Based Access Control will be used to enforce this.

- **Data protection:** Before being stored, sensitive information, including passwords, is hashed using techniques like bcrypt. Data will be encrypted both in transit and at rest to prevent unwanted access.

8.4 Database Security

In order to protect stored data against breaches and illegal access, database security is essential:

- **Encryption:** Industry-standard encryption algorithms will be used to encrypt data that is at rest. We'll make use of MongoDB's built-in encryption mechanisms to protect your data.

- **Access Control:** Role-based access control will be used to restrict database access to individuals who are permitted. This will stop unauthorized individuals from changing or accessing private information.

- **Frequent Backups:** To guarantee that data can be recovered in the case of corruption or loss, regular backups will be planned. Redundancy will be provided via both on-site and off-site storage in backup plans.

8.5 References

1. React Documentation. (n.d.). Retrieved from https://reactjs.org/docs/getting- started.html
2. Express.js Documentation. (n.d.). Retrieved from https://expressjs.com/en/starter/installing.html
3. Node.js Documentation. (n.d.). Retrieved from https://nodejs.org/en/docs/
4. MongoDB Documentation. (n.d.). Retrieved from https://docs.mongodb.com/manual/
5. Shrestha, S. (2023). Full-Stack Web Development with the MERN Stack: Mongoose for MongoDB. Retrieved from https://www.digitalocean.com/community/tutorials/full-stack-web-development-with-mern
6. Myers, K. (2023). Best Practices for Secure API Development. Retrieved from https://www.owasp.org/index.php/Top_10

9. Conclusion

Based on the technology research conducted for the CurtinTalentTrack project, the MERN stack - which consists of React, Express.js, Node.js, and MongoDB was chosen to serve as the web application's base. Because of this stack's performance, scalability, security features, and consistency with the project's goals, it was selected. The program will be able to provide a top-notch user experience while upholding strong security and scalability thanks to the integration and optimization techniques described.

Part Two Agile

10. Agile

10.1 Agile Development Process

We adopted agile process throughout planning the CurtinTalentTrack application. It outlines the Agile practices we followed including client interactions, internal team meetings.

10.2 Internal Communications and Sprint Planning

10.2.1 Daily Communication

We made a WhatsApp group and comminated through the group to discuss progress and have daily updates on the project.

10.2.2 Meeting Documentation

We uploaded all our group meeting minutes to a Google Drive and also sent to all member and as well as emailed to our client and supervisor.

Meeting 3
https://drive.google.com/file/d/1w0iKe4EQ43A1wUbocTTIPbGTchkBDwji/view?usp=share_link

Meeting 4

https://drive.google.com/file/d/1AwEiYHEYLbwdIGe7v_VtoOR9wlJ9GIcA/view?usp=sharing

10.3 Progress Tracking and Metrics

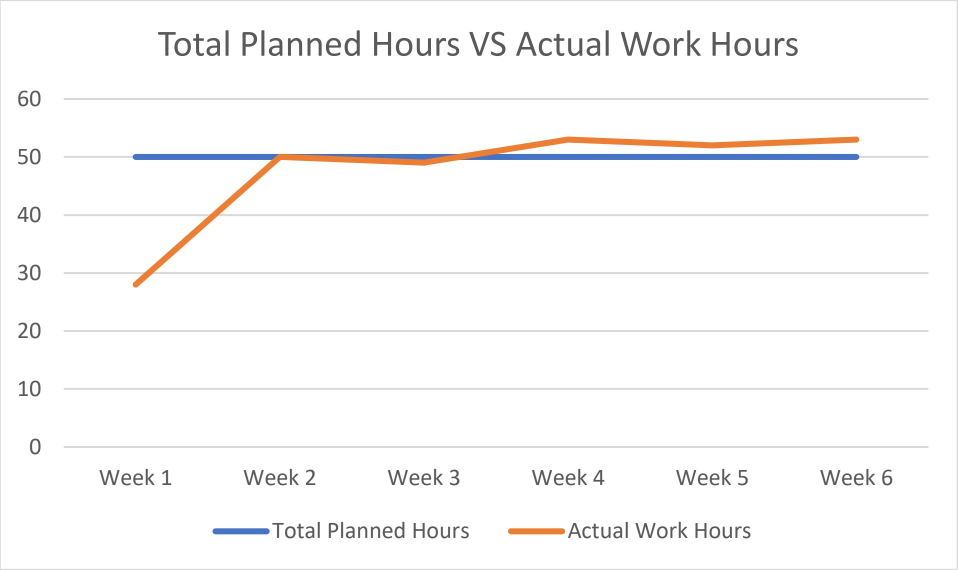
All charts are in a spreadsheet in the Google Drive link below:

https://docs.google.com/spreadsheets/d/1-FvvdftIMFHd4Zb7EnxQZLxN099e--wt/edit?usp=share_link&ouid=100909382276998164235&rtpof=true&sd=true

10.3.1 Time Tracking

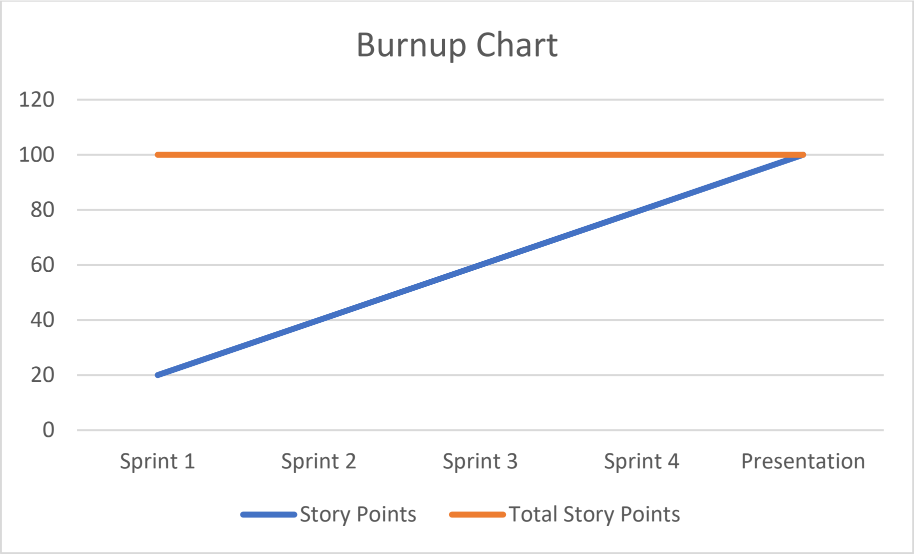
We used togglTrack to keep track the time spent. This allowed us to monitor our progress accurately and ensure we were on track to meet deadlines. All our toggl reports are uploaded to a Google Drive. (Link below)

https://drive.google.com/drive/folders/1G16F20JoTuJBAHq7w_Pja6vcH3mCNsK4?usp=share_link



10.3.2 Burn Up Chart

We created a burnup chart to track our progress and gave story points to get an idea on how things are going.

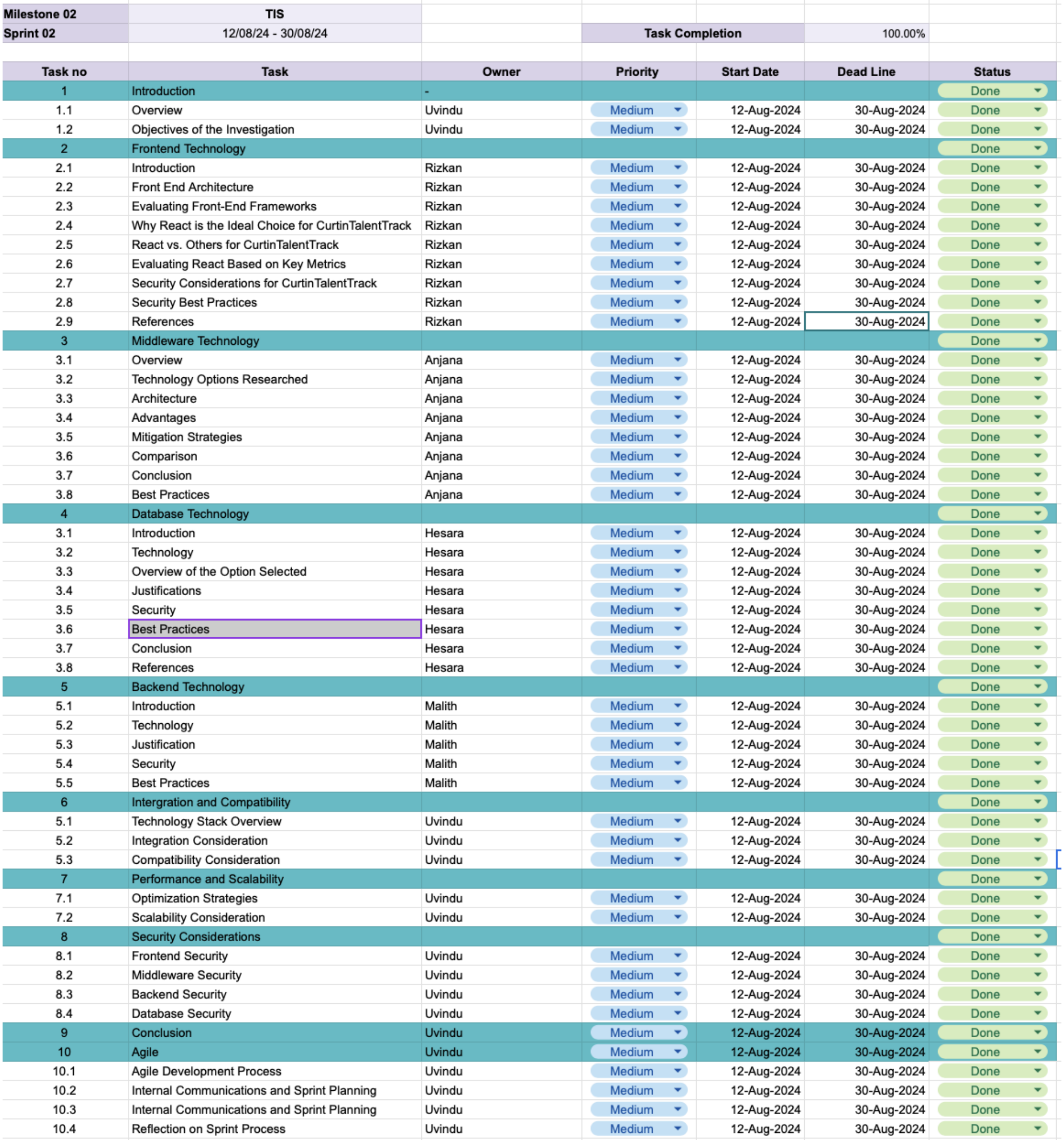


https://docs.google.com/spreadsheets/d/1-FvvdftIMFHd4Zb7EnxQZLxN099e--wt/edit?usp=share_link&ouid=100909382276998164235&rtpof=true&sd=true

10.3.4 Sprint Backlog

We used the following sprint backlog to give tasks to members and keep track of the progress.

https://docs.google.com/spreadsheets/d/1Wz0nsmCQ8rboeHLidMYOsBZSNGszHoLRJshrOCSdATQ/edit?gid=0#gid=0



Milestone 02	TIS					
Sprint 02	12/08/24 - 30/08/24		Task Completion		100.00%	
Task no	Task	Owner	Priority	Start Date	Dead Line	Status
1	Introduction	-				Done
1.1	Overview	Uvindu	Medium	12-Aug-2024	30-Aug-2024	Done
1.2	Objectives of the Investigation	Uvindu	Medium	12-Aug-2024	30-Aug-2024	Done
2	Frontend Technology					Done
2.1	Introduction	Rizkan	Medium	12-Aug-2024	30-Aug-2024	Done
2.2	Front End Architecture	Rizkan	Medium	12-Aug-2024	30-Aug-2024	Done
2.3	Evaluating Front-End Frameworks	Rizkan	Medium	12-Aug-2024	30-Aug-2024	Done
2.4	Why React is the Ideal Choice for CurtinTalentTrack	Rizkan	Medium	12-Aug-2024	30-Aug-2024	Done
2.5	React vs. Others for CurtinTalentTrack	Rizkan	Medium	12-Aug-2024	30-Aug-2024	Done
2.6	Evaluating React Based on Key Metrics	Rizkan	Medium	12-Aug-2024	30-Aug-2024	Done
2.7	Security Considerations for CurtinTalentTrack	Rizkan	Medium	12-Aug-2024	30-Aug-2024	Done
2.8	Security Best Practices	Rizkan	Medium	12-Aug-2024	30-Aug-2024	Done
2.9	References	Rizkan	Medium	12-Aug-2024	30-Aug-2024	Done
3	Middleware Technology					Done
3.1	Overview	Anjana	Medium	12-Aug-2024	30-Aug-2024	Done
3.2	Technology Options Researched	Anjana	Medium	12-Aug-2024	30-Aug-2024	Done
3.3	Architecture	Anjana	Medium	12-Aug-2024	30-Aug-2024	Done
3.4	Advantages	Anjana	Medium	12-Aug-2024	30-Aug-2024	Done
3.5	Mitigation Strategies	Anjana	Medium	12-Aug-2024	30-Aug-2024	Done
3.6	Comparison	Anjana	Medium	12-Aug-2024	30-Aug-2024	Done
3.7	Conclusion	Anjana	Medium	12-Aug-2024	30-Aug-2024	Done
3.8	Best Practices	Anjana	Medium	12-Aug-2024	30-Aug-2024	Done
4	Database Technology					Done
3.1	Introduction	Hesara	Medium	12-Aug-2024	30-Aug-2024	Done
3.2	Technology	Hesara	Medium	12-Aug-2024	30-Aug-2024	Done
3.3	Overview of the Option Selected	Hesara	Medium	12-Aug-2024	30-Aug-2024	Done
3.4	Justifications	Hesara	Medium	12-Aug-2024	30-Aug-2024	Done
3.5	Security	Hesara	Medium	12-Aug-2024	30-Aug-2024	Done
3.6	Best Practices	Hesara	Medium	12-Aug-2024	30-Aug-2024	Done
3.7	Conclusion	Hesara	Medium	12-Aug-2024	30-Aug-2024	Done
3.8	References	Hesara	Medium	12-Aug-2024	30-Aug-2024	Done
5	Backend Technology					Done
5.1	Introduction	Malith	Medium	12-Aug-2024	30-Aug-2024	Done
5.2	Technology	Malith	Medium	12-Aug-2024	30-Aug-2024	Done
5.3	Justification	Malith	Medium	12-Aug-2024	30-Aug-2024	Done
5.4	Security	Malith	Medium	12-Aug-2024	30-Aug-2024	Done
5.5	Best Practices	Malith	Medium	12-Aug-2024	30-Aug-2024	Done
6	Intergration and Compatibility					Done
5.1	Technology Stack Overview	Uvindu	Medium	12-Aug-2024	30-Aug-2024	Done
5.2	Integration Consideration	Uvindu	Medium	12-Aug-2024	30-Aug-2024	Done
5.3	Compatibility Consideration	Uvindu	Medium	12-Aug-2024	30-Aug-2024	Done
7	Performance and Scalability					Done
7.1	Optimization Strategies	Uvindu	Medium	12-Aug-2024	30-Aug-2024	Done
7.2	Scalability Consideration	Uvindu	Medium	12-Aug-2024	30-Aug-2024	Done
8	Security Considerations					Done
8.1	Frontend Security	Uvindu	Medium	12-Aug-2024	30-Aug-2024	Done
8.2	Middleware Security	Uvindu	Medium	12-Aug-2024	30-Aug-2024	Done
8.3	Backend Security	Uvindu	Medium	12-Aug-2024	30-Aug-2024	Done
8.4	Database Security	Uvindu	Medium	12-Aug-2024	30-Aug-2024	Done
9	Conclusion	Uvindu	Medium	12-Aug-2024	30-Aug-2024	Done
10	Agile	Uvindu	Medium	12-Aug-2024	30-Aug-2024	Done
10.1	Agile Development Process	Uvindu	Medium	12-Aug-2024	30-Aug-2024	Done
10.2	Internal Communications and Sprint Planning	Uvindu	Medium	12-Aug-2024	30-Aug-2024	Done
10.3	Internal Communications and Sprint Planning	Uvindu	Medium	12-Aug-2024	30-Aug-2024	Done
10.4	Reflection on Sprint Process	Uvindu	Medium	12-Aug-2024	30-Aug-2024	Done

10.4 Reflection on Sprint Process

10.4.1 Reflections

We have completed the TIS report with comprehensive description according to the guideline on time. All team members have worked more than allocated hours as well. Rather than the last sprint all team members have put more effort and time to research and documentation

10.4.2 Improvements

Moving forward, we will utilize new tools and have more meeting to streamline our project.

Attributions (Only for Documentation)

Task No	Task	Anjana	Hesara	Malith	Rizkan	Uvindu
1	Introduction				[redacted]	
1.1	Overview				[redacted]	
1.2	Objectives of the Investigation				[redacted]	
2	Front End Technology				[redacted]	
2.1	Introduction				[redacted]	
2.2	Front End Architecture				[redacted]	
2.3	Evaluating Front-End Frameworks				[redacted]	
2.4	Why React is the Ideal Choice for CurtinTalentTrack				[redacted]	
2.5	React vs. Others for CurtinTalentTrack				[redacted]	

2.6	Evaluating React Based on Key Metrics				■	
2.7	Security Considerations for CurtinTalentTrack				■	
2.8	Security Best Practices				■	
2.9	References				■	
3	Middleware Technology	■				
31	Overview	■				
3.2	Technology Options Researched	■				
3.3	Architecture	■				
3.4	Advantages	■				
3.5	Mitigation Strategies	■				
3.6	Comparison	■				

3.7	Conclusion	■				
3.8	Best Practices	■				
4	Database Technology		■			
4.1	Introduction		■			
4.2	Technology		■			
4.3	Overview of the Option Selected		■			
4.4	Justifications		■			
4.5	Security		■			
4.6	Best Practices		■			
4.7	Conclusion		■			
4.8	References		■			

5	Backend Technology			■		
5.1	Introduction			■		
5.2	Technology			■		
5.3	Justification			■		
5.4	Security			■		
5.5	Best Practices			■		
6	Intergration and Compatibility					■
6.1	Technology Stack Overview					■
6.2	Integration Consideration					■
6.3	Compatibility Consideration					■
7	Performance and Scalability					■

7.1	Optimization Strategies					■
7.2	Scalability Consideration					■
8	Security Considerations					■
8.1	Frontend Security					■
8.2	Middleware Security					■
8.3	Backend Security					■
8.4	Database Security					■
9	Conclusion					■
10	Agile					■
10.1	Objectives of the Investigation					■
10.2	Internal Communications and Sprint Planning					■

10.3	Internal Communications and Sprint Planning					[redacted]
10.4	Reflection on Sprint Process					

Software Requirements Specification

For

CurtinTalentTrack

Version 1.0

Prepared By

SD06

Anjana Wishwajith Premadasa - 20463382
Malith Pramuditha - 20926076
Hesara Yasaswin Pathirana – 20928386
Uvindu Gamage - 20528142
Rizkan Rhazes – 21309380

August 1, 2024

Part One - Main Document (SRS)

1. Introduction

1.1 Purpose

The purpose of this Software Requirements Specification (SRS) document is to detail the requirements and specifications for the CurtinTalentTrack system. This system aims to enhance communication skills within the Curtin Community, focusing on three main areas: Public Speaking, Written Communication, and Critical Thinking/Problem Solving. This document acts as a complete guide for all stakeholders, including developers, designers, and project managers, outlining the functional and non-functional requirements, constraints, and other essential aspects of the system.

1.2 Scope

1.2.1 Software Product Identification

We're developing a web platform called "CurtinTalentTrack" for everyone in the Curtin community. This includes students, teachers, other staff members, and alumni.

1.2.2 Product Capabilities and Limitations

1.2.2.1 CurtinTalentTrack will:

- Offer tools and resources to boost communication skills, focusing on public speaking, writing, and critical thinking/problem-solving.
- Let users register, create profiles, and personalize their experience.
- Provide features for recording and uploading speeches, practicing interview questions, and submitting written work for feedback from evaluators.
- Include quizzes and exercises to boost critical thinking and problem-solving skills.
- Allow administrators to manage content, events, and user interactions through an admin dashboard.
- Support event creation and promotion, with functionalities for user registration and reminders.

1.2.2.2 CurtinTalentTrack will not:

- Include project development plans like cost estimates, staffing details, or schedules.
- Provide detailed design specifications or go beyond the outlined functional and non-functional requirements in terms of software development.

1.2.3 Application of the Software

CurtinTalentTrack is intended for use by the Curtin Community members to develop and improve their communication skills. The benefits and objectives are:

Enhanced Communication Skills: Access to resources and practice opportunities in public speaking, written communication, and critical thinking.
Professional Development: Assist users in landing a job via interviews and; Enhance the presentation of user's reports, proposals and any written work in professional settings.
Community Engagement: Promoting the culture of lifelong learning and staff and alumni skill enhancement among the existing community members.
Structured Feedback: Introducing a feature by which users may get suggestions from professional evaluators and what they should do to change for the better.

1.2.4 Consistency with Higher-Level Specifications

The SRS concurs with the higher-level requirements such as the system requirements, if there are any; that is, it is consistent with defining the project scope, objectives, and intended functionalities to concur with supporting the overall goals of the Curtin Community about improving communication skills.

1.3 Definitions, Acronyms, and Abbreviations

- **Curtin Community**: All the students, academic staff, non-academic staff, and alumni of Curtin University.
- **User**: A student, staff, or alumnus that operates the system.
- **Administrator (Admin)**: System operator with the appropriate rights regarding the management of user accounts, content, and events.
- **Evaluator**: Employer, who is paid to evaluate submissions by users under the administrator and give his/her opinion about them, preferably in the fields of public speaking and writing.
- **Public Speaking**: Public speaking involves speaking in front of a group of people or an audience in one format or another such as giving a speech or a presentation. It fosters a specific talent in the same through practice and feedbacks to ensure that the users are confident while presenting the expressions before the audience.
- **Written Communication:** Writing messages like emails, memos, and reports.
- **Critical Thinking and Problem Solving:** Thinking clearly about problems or situations to make good decisions. The program helps you get better at this with tests and activities.
- **SRS**: Abbreviation of Software Requirements Specification. A document that describes all data, functional, and behavioural requirements of a software system in detail.

1.4 References

1.4.1 List of Referenced Documents

This section lists all the documents referenced in this Software Requirements Specification (SRS) for the CurtinTalentTrack project.

1.4.2 Identification of Documents

1.4.2.1 IEEE-STD-830-1998

- **Title:** IEEE Recommended Practice for Software Requirements Specifications
- **Report Number:** IEEE-STD-830-1998
- **Date:** 1998
- **Publishing Organization:** Institute of Electrical and Electronics Engineers (IEEE)

1.4.2.2 Client Brief for CurtinTalentTrack

- **Title:** Client Brief for CurtinTalentTrack Project
- **Client Meeting 01 – Requirement Gathering**
- **Name of Meeting/Project:** CurtinTalentTrack: A System to Improve Communication Skills in the Curtin Community
- **Day & Date:** 26th July 2023
- **Time:** 1:15 pm - 2:00 pm
- **Location:** Curtin Colombo

1.4.3 Sources of References

1.4.3.1 IEEE-STD-830-1998

Source: Available from the Institute of Electrical and Electronics Engineers (IEEE). Also referenced in the weekly topic section on Blackboard regarding how to write an SRS.

1.4.3.2 Client Brief for CurtinTalentTrack

Source: Provided directly by the client, Umanga Pilapitiya. The document was emailed by Miss Ann.

1.4.3.3 Client Meeting Minutes

Source: Emailed to the client and Miss Ann, containing details from the meeting on 26th July 2023.

1.5 Overview

1.5.1 Contents of the SRS

The requirements for the **CurtinTalentTrack** system are contained in this Software Requirements Specification document. It contains a broad description of the system, functional and non-functional requirements, design constraints, and other information.

1.5.2 Organization of the SRS

The **SRS** is structured with the following major sections:

- **Introduction**: Describes what the project is all about; its purpose, scope, and references.
- **General Description**: Describes the general functionality of the system and outlines characteristics of the users and other constraints.
- **System Features and Interface**: Listing of the functional and non-functional requirements that describe the system characteristics and how it is expected to operate.
- **User Interfaces**: Describes the design constraints for the user interface on the system.
- **Document Appendices**: Any additional information that would be considered necessary in the support of the document.
- **Index**: For quick reference, a table of contents is included at the beginning of the document.

2. Overall Description

2.1 Product Perspective

To Develop the Curtin Community's communication, a web-based application named CurtinTalentTrack will be developed. It will be an important part of the professional and educational life of students as well as members of the faculty. This Web Application Will be covering a vast list of issues connected with communication and including such topics as Written Communication, and Public Speaking, Critical Thinking/Problem-Solving. What separates the solution from other tools is that the work includes video recording, practice tasks, and the comments of the evaluator.

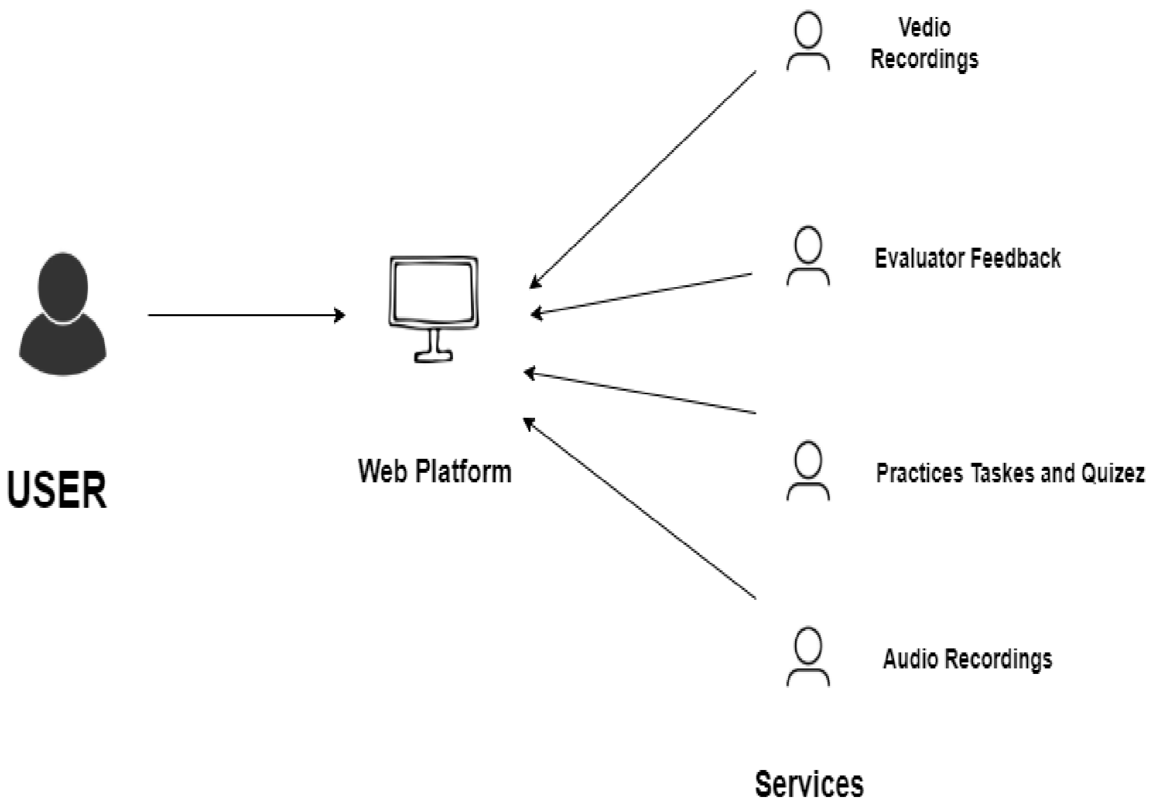


Figure 6 - Basic Plan

2.2 Product Features

2.2.1 User Registration

With respects to safety of the data that users input into the CurtinTalentTrack platform, the sign-up and login process use will be secure. A particular view of the possibility of changing account settings to match users' preferences and needs can be provided. To maintain the environment formal and strict, administrators will incorporate evaluators into the system.

2.2.2 Public Speaking and Interview Practice

There are many selections for the public speaking films that present tips on how to enhance the ability as well as the previews of both good and bad speeches. Users of the site add their talks and discussions that are posted and can be accessed by everyone; the maximum length of the talk is 5 minutes. Possible actions during the usage of the application may include recording and saving practice sessions while observing examples as well as Job-Specific Questions. In order to ensure independent and fair assessments only an automatic selection allocates the listeners and gives non-confidential opinions.

2.2.3 Written Communication

several written communication formats such as emails, CVs, and letters can be accessed from this web application. Users can practice on different writing tasks that are implemented in workplace. Members can get private comments on how they can further build up their writing skills and the system finds evaluators for theses compositions.

2.2.4 Critical Thinking and Problem Solving

This web Application has the ideas of having drills with quizzes as ways of enhancing the decision making and problem-solving skills. Taking structured and multiple-choice questions, the answers will mark by an automatic grading system. It means that users can pass through several levels of degree of difficulty and their performance is being observed and this enhances the aspects of development of critical thinking.

2.2.5 Event Creation

A few specific possibilities of using an event calendar to plan seminars and other events to advertise them are stated below. To ensure that the users are informed of these possibilities, there may be options to register for the event with alerts concerning the event's updates automatically.

2.2.6 Dashboard

There are 3 Dashboards for each category,

User – All the three core features along with their sub features should be displayed.

Evaluator – All Pending, completed jobs should be displayed.

Admin – All Administrative Functions and statics should be included in a Dashboard.

2.3 User Classes and Characteristics

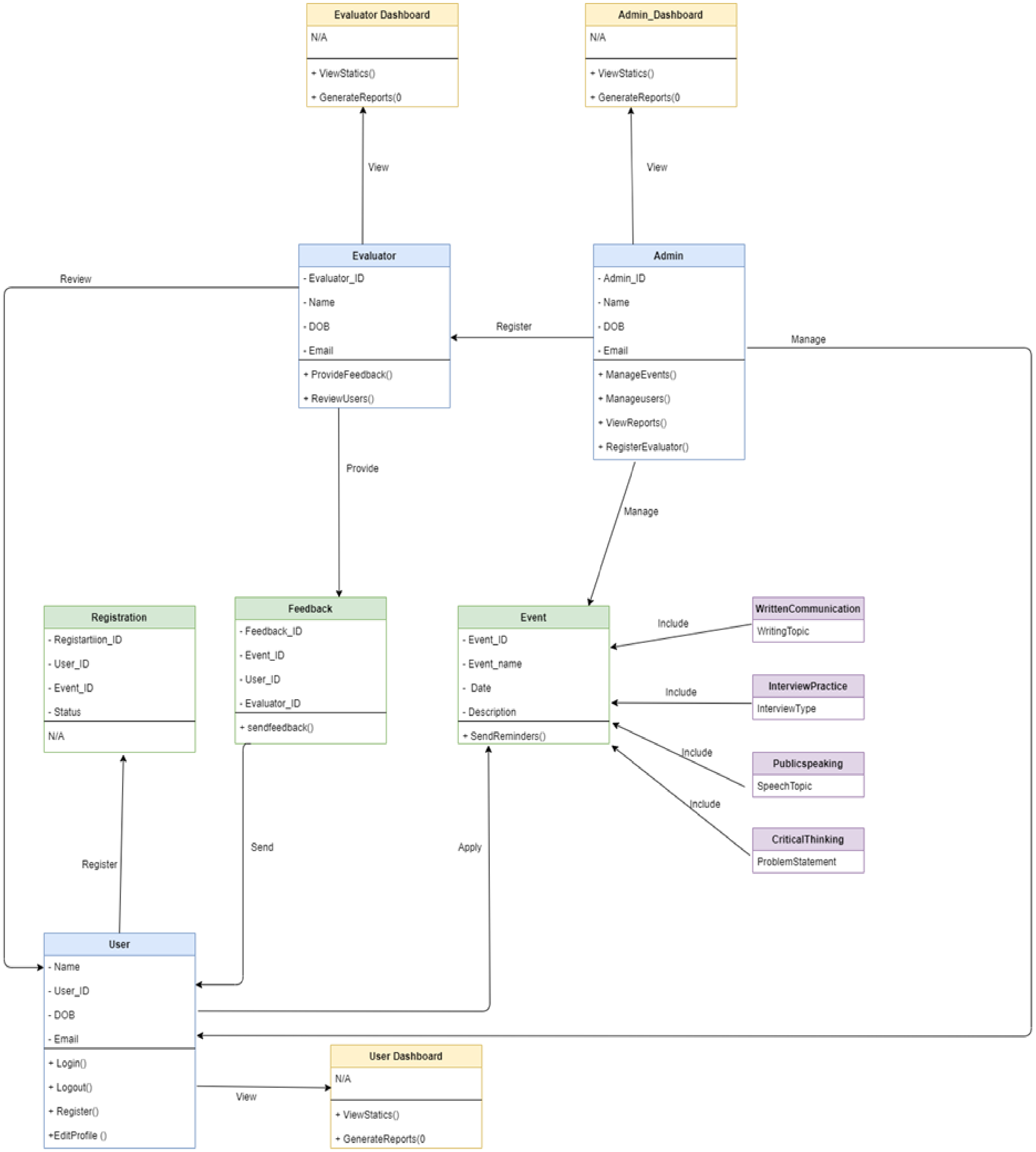


Figure 7 - UML Diagram

2.4 Operating Environment of Web Application

This Platform will be web based application. The platform will be able to sync with different operating systems such as window, Mac, Linux, iPhone, and android.

This application will be hosted in Linux server thus making it very reliable and fast. it will also use MERN stack.

As for the server side of this Web Application, the concepts of data-based servers provided by the leading companies in this sphere, including MongoDB and MySQL, will be used. Proven and efficient paradigms such as Express. js and Node. js will be implement when it comes to the creation of the backend services, which speaks of the safe and responsive experiences for the users.

Web server – Node.js good web server that is known for its performance and scalability. This is well-suited real-time applications.

Database Server – MongoDB and MYSQL will be used for Database also popular for store large amount of data.

Frontend – React will be used to build frontend. It's also popular frontend framework. Also, powerful and widely used.

2.5 Design and Implementation Constraints

2.5.1 Data Privacy and Security

Concerning the development of this Web, security will be considered a priority through number of measures. The platform respects and adheres to the rules of data protection, thus, ensuring proper handling of the given data. User accounts and users' data are safeguarded by the mechanisms of secure authentication and authorization.

2.5.2 Technological Performance

CurtinTalentTrack will be working with the video upload and playback which are crucial for the interview and speech rehearsals. The reliability of the system will also be guaranteed and further enhanced to accommodate more than one user in given instance.

2.5.3 System Usability

The functionality is a major priority that takes usability into consideration, and this means that the web site's design will be relatively simple, and its navigation must be intuitive to allow users to effectively engage with the different components of the website. However, this Web App will be available to all Curtin Community because of the incorporation of accessibility options for the impaired users.

2.6 User Documentation of Web Application

One of top priority thing is User should need to understand system correctly. Or there will be errors because user's misunderstandings. there are some Aid provided with the software,

2.6.1 User Manual

For all the users in CurtinTalentTrack, there will be user guides for the students, administrators, evaluators, academic staff, and the non-academic candidates. These comprehensive manuals contain detailed written, critical thinking and spoken word instructions in terms of usage of the system's functions, that include written communication exercises, critical thinking tests, and public speaking practice among others. Each manual is adjusted to the characteristics of the target user audience, ensuring that users can make the most of the opportunities within the platform.

2.6.2 Help Desk

To assist the users even more, there will be Implement online support centre. to help customers gather fast solutions to such issues that they face often, this portal has a dedicate FAQ section and tips and tricks. The contact details for the technical support of the company will be provided for more complex technological problems so that the consumers can get the want as soon as possible.

2.6.3 Tutorials and Training

This Application Will be aware that education does not stop after getting a job and that course on boarding needs to be effective. To ensure that first time users can know how to use the platform, the platform will has a video tutorial and walkthrough. There will be also specific training courses available, which would ensure that administrators and evaluators have enough experience to lead the system and provide better assistance. Such tools are supposed to build the users' competency and confidence, thus would contribute to the overall success of the platform.

2.7 Assumptions and Dependencies

Assumptions

2.7.1 Users have internet access and good training with web browsers and Servers.

First of all, as for the requirements, users should have prior experience in using web browsers and also, they should have internet connection. This must be attained to ensure that the different functions of the platform including writing, recording, and use of other resources are accessible and can be operated by the user. Without these skills, users may have complications to gain the maximum benefits out of the website.

2.7.2 Evaluators are willing to provide good feedback at time.

The approach is therefore based on the assumption that academic personnel, business executives or other suitably competent persons who are to be engaged as evaluators will be able to offer their comments within a short time. Such timely and helpful feedback is beneficial in understanding how CurtinTalentTrack enhances the user's communication experience.

2.7.3 Regular updates and maintenance

With such a system in place, it is expected that the regular maintenance and updating processes would be conducted to ensure the system is both up-to-date and fully functional. This include error correction in the program, enhancements of the security measures as well as implementing features that may be missing in the program. The technical work to ensure that the platform remains useful, sound and contemporary needs some constant attention**.**

Dependencies

2.7.4 Depend on service providers for hosting and storage.

There expected that perennial service and renovation would be implement to ensure the specific system is up to date and functional. This include rectifying program errors, improving security measures, and implementing new modalities as circulating. These all need constant updates and care since the platform must always be functional, protected, and up to date for its users from a modern technological standpoint.

2.7.5 Reliance on third-party tools and libraries

Other tools and libraries are also necessary for such things as text manipulation, video manipulation, and the alerting system. Such features like reliable alerting systems, Interface for text editing. The services and utility of the system relies immensely on these third party technologies in terms of effectiveness and reliability.

2.7.6 Resources and Budget

The availability of resources budget, experienced Developers, Support Platform and Knowledge are all critical dependencies for this success of this project.

3. System Features

3.1 User Authentication and Login Feature

3. 1. 1 Description and Priority:

With this feature, user can safely enter their login details into the web-based application. This is a high priority task since the first steps towards personalizing the users' experiences and their data are taken here.

3. 1. 2 Stimulus/Response Sequences:

- **Stimulus:** The user gets to the home page where there is a link to the login page
  - **Response:** The system presents the login form to the user.
- **Stimulus:** The user re-enters their details and clicks on the submit button or submits the form.
  - **Response:** It checks the credentials for the authenticity, and hence awards the user with access to their account.

3. 1. 3 Functional Requirements:

• **REQ-1:** The system shall present the login page for the secure access with textbox for the username/Email and password.

• **REQ-2:** The system shall check the identity and authority of the user and allow him/her to access resources if okay.

• **REQ-3:** Inability to enter the correct password and any other related errors should be accompanied by relevant error messages in the system.

• **REQ-4:** Password hashing and secure session management must also be managed in the system to enhance security of the data.

3.2 User Registration and Sign-Up Feature

3.2.1 Description and Priority:

This feature allows new users to register for the system. It is of High priority, as it facilitates user engagement and expands the user base.

3.2.2 Stimulus/Response Sequences:

- **Stimulus:** New user clicks on "Sign Up."
  - **Response:** The system introduces a registration form.
- **Stimulus:** The user fills out the form and submits it.
  - **Response:** The system checks form data, and this form data is used to create account for the new user.

3.2.3 Functional Requirements:

- **REQ-1:** It is required to allows the user to register; the registration form should have fields for information that is required to be filled by the user.
- **REQ-2:** The system shall check for the validity of the fields in form and if correct, the form should create a new user.
- **REQ-3**: The system shall generate and forward an auto reply email to the newly registered user.
- **REQ-4:** The system shall retrieve and manage user's information stored securely in the application and link the user to the correct role.

3.3 User Profile Access and Management Feature

3.3.1 Description and Priority:

This feature provides the users with the profiles with own information and other information concerning the system. It is considered high priority, due to the direct interaction with the end-users and the opportunity to provide more relevant content.

3.3.2 Stimulus/Response Sequences:

- **Stimulus:** Logged-in user selects "My Profile."
    - **Response:** The system displays a complete overview of user's information.
- **Stimulus:** The user chooses to update profile information.
    - **Response:** It offers input fields for profile data that can be edited by the user.
- **Stimulus:** The user modifies the profile and send changes to the application.
    - **Response:** The system checks and then modifies the profile information saved in the database.

3.3.3 Functional Requirements:

- **REQ-1:** The system will present a feature commonly known as "My Profile", where the users can find all relevant data related to them.
- **REQ-2:** The system shall allow the user to view the profile they created with personal information as well as information acquired from the system.
- **REQ-3:** Profile information update and validation of such changes should be made possible by the system.
- **REQ-4:** The system should be able to store information about users that are specific to them and should be regularly updated.

3.4 Public Speaking Practice Feature

3.4.1 Description and Priority:

This feature gives users a chance to develop their speaking skills by offering the possibility to submit speeches as well as evaluates them. It is of High priority as it is one of the core features which helps in the formation of good communication skills.

3.4.2 Stimulus/Response Sequences:

- **Stimulus:** User selects "Public Speaking Practice."
    - **Response:** The above system shows options for submitting speech in addition to options for evaluating already submitted speeches
- **Stimulus:** User submits a speech for evaluation.
    - **Response:** The speech is saved to the system and informs an evaluator through a notification.
- **Stimulus:** Evaluator reviews the speech and provides feedback.
    - **Response:** The feedback is then shown to the user through the system.

3.4.3 Functional Requirements:

- **REQ-1:** The system should have a functionality whereby users can submit speeches for assessment.
- **REQ-2:** The system will notify evaluators when a speech is uploaded.
- **REQ-3:** Allows evaluators to give feedback on speeches uploaded to the system.
- **REQ-4:** The system should be able to display evaluator feedback to the users.
- **REQ-5:** The system should show samples of good and bad speeches a tips to users.

3.5 Interview Practice Feature

3.5.1 Description and Priority:

It allows the users to prepare for interview sessions through simulation interview and follow up feedback. This is also high priority as this being one of the core features of the web-app.

3.5.2 Stimulus/Response Sequences:

- **Stimulus:** User selects "Interview Practice."
    - **Response:** The system shows the slots for mock interview that is available.
- **Stimulus:** User asks for a schedule of a mock interview.
    - **Response:** The system schedules the mock interview and notifies the user.
- **Stimulus:** User takes the mock interview.
    - **Response:** The interview is recorded, and the feedback is saved in the system.

3.5.3 Functional Requirements:

- **REQ-1:** The system shall provide a scheduling interface for booking mock interview slots.
- **REQ-2:** The system shall notify users of their scheduled mock interviews.
- **REQ-3:** The system shall record the mock interview sessions.
- **REQ-4:** The system shall store and display feedback from the mock interviews.
- **REQ-5:** The system should show samples of good and bad interviews a tips to users.

3.6 Written Communication Practice Feature

3.6.1 Description and Priority:

This feature allows the users to apply written communication by submitting written tasks and getting feedback. It is of High priority, as this is the third core feature of the web-app.

3.6.2 Stimulus/Response Sequences:

- **Stimulus:** User chooses "Written Communication Practice".
  - **Response:** A list of writing assignments that can be done via the system is shown.
- **Stimulus:** User submits a written task for evaluation.
  - **Response:** The system saves the submission and informs an evaluator.
- **Stimulus:** Evaluator reviews the written task and provides feedback.
  - **Response:** The system displays the feedback to the user.

3.6.3 Functional Requirements:

- **REQ-1:** The system should offer selections of writing activities that users may select from.
- **REQ-2**: The system should enable users to submit written assignments for grading.
- **REQ-3**: An alert should be generated to the evaluators each time written tasks have been submitted.
- **REQ-4**: Feedback from the evaluator to the users must be displayed by the system.

3.7 Critical Thinking/Problem-Solving Practice Feature

3.7.1 Description and Priority:

This standardized feature assists the users in enhancing their critical thinking and problem-solving abilities through practice as well as assessments. It is of High Importance as it boosts one's mental capacity.

3.7.2 Stimulus/Response Sequences:

- **Stimulus:** User picks “Critical Thinking/Problem Solving Practice “.
  - **Response:** Available exercises are presented to the system (selected from question pool)
- **Stimulus:** Users participate and submit an exercise for grading.
  - **Response:** The system saves the submission to the database.
- **Stimulus:** The system grades the tests using the mark sheets stored in system.
  - **Response:** The system displays the feedback/grade to the user.

3.7.3 Functional Requirements:

- **REQ-1:** The system should contain a question pool of recommended critical thinking and problem-solving exercises.
- **REQ-2:** Should allow the users to submit their completed exercises for grading.
- **REQ-3:** The system should be able to mark the exercises using the mark sheets sorted in database.
- **REQ-4:** The system should show the graded exercise to the user.

3.8 Event Creation and Management Feature

3.8.1 Description and Priority:

This feature enables the administrators to add extracurricular events to the dashboard and allows users to sign-up and register to them. This feature makes it easier to organize and attend events.

3.8.2 Stimulus/Response Sequences:

- **Stimulus:** Admin selects "Create Event."
  - **Response:** The system has an event creation form that comes up.
- **Stimulus:** Admin fills out the form and submits it.
  - **Response:** The system provides the confirmation to the creation of the event.
- **Stimulus:** Users register for the event.
  - **Response:** The system acknowledges registration and modifies the list of participating events.

3.8.3 Functional Requirements:

- **REQ-1:** It must enable administrators to create an event through a form
- **REQ-2:** The system will be responsible for verifying and generating new events based on the provided information.
- **REQ-3:** Allows user to register events.
- **REQ-4:** The system should update the participants list for the events.
- **REQ-5:** The system should show the created events in a calendar form to users (in dashboard).

3.9 Admin Dashboard Feature

3.9.1 Description and Priority:

This feature offers administrators a way of managing users, events, and evaluators through a dashboard. This is a high priority feature as this dashboard gives admin whole control of the system.

3.9.2 Stimulus/Response Sequences:

- **Stimulus:** Admin logs in and selects "Admin Dashboard."
    - **Response:** The system is presented with a dashboard that shows numerous management possibilities.
- **Stimulus:** Admin selects an option (e.g., manage users).
    - **Response:** A management interface pertaining to the displayed task appears on the screen.

3.9.3 Functional Requirements:

- **REQ-1:** The system will have to offer a detailed dashboard for administrators.
- **REQ-2:** Managing users, event, and evaluators should be made possible through the created dashboard.
- **REQ-3:** Depending on the selected option the system should show the required management interfaces.

REQ-4: The system shall include security measures that guarantee the access to the admin dashboard as well as its various functionalities.

4. External Interface Requirements

4.1 User Interfaces

4.1.1 Login and Sign up Interfaces

Secured login and sign up interfaces for users as well as evaluators and administrators.

Elements

- Input fields for username, password, and personal details
- Buttons for login, sign-up, and forgot password.
- Error messages for invalid inputs.

Behaviour

- Users can register by providing necessary details and receive a confirmation email.
- Secure login using username and password.
- Password reset functionality through email verification.

4.1.2 User Profile Interface

Interface for users to view and edit their profile information.

Elements

- Display fields for user details
- Editable fields for certain profile information.
- Button for saving changes.

Behaviour

- Users can update their profile information.
- Changes are saved and updated in the database immediately.

4.1.3 Public Speaking and Interview Practice Interface

Interface for viewing public speaking/interview videos and recording practice sessions.

Elements

- Video player for viewing sample videos.
- Recording tool for users to record their speeches or interview responses.
- Upload button to save recordings.

Behaviour

- Users can view tips and sample videos.
- Users can record and upload their practice sessions.
- Notifications sent to evaluators upon submission.

4.1.4 Written Communication Practice Interface

Interface for viewing sample documents and practicing writing tasks.

Elements

- Text editor for composing written tasks (e.g., emails, memos)
- Sample document viewer.
- Save button for submitting written tasks.

Behaviour

- Users can view sample documents.
- Users can write and save their practice tasks.
- Notifications sent to evaluators upon submission.

4.1.5 Critical Thinking and Problem-Solving Interface

Interface for participating in quizzes and problem-solving exercises.

Elements

- Quiz questions (MCQs and structured questions).
- Answer input fields.
- Submit button for quiz answers.
- Score display at the end of the quiz.

Behaviour

- Users can take quizzes and receive immediate feedback.
- Scores and standings are displayed upon completion.

4.1.6 Event Management Interface

Interface for administrators to create and promote events.

Elements

- Event creation form with fields for event details (title, date, description).
- Event calendar viewer.
- Registration and reminder options.

Behaviour

- Administrators can add, edit, and delete events.
- Users can register for events and receive reminders.

4.1.7 Admin Dashboard Interface

Comprehensive dashboard for administrators to manage the system.

Elements

- Statistics and reports on user activities.
- Management tools for adding content (videos, samples, quizzes)
- User management tools (adding evaluators, reviewing users).

Behaviour

- Administrators can view system metrics and perform management tasks.
- All actions and updates are logged.

4.2 Hardware Interfaces

4.2.1 Client Devices

Users will access the system via web browsers on various devices.

Requirements

- Desktop and laptop computers with modern web browsers.
- Mobile devices (smartphones and tablets) with responsive design support.

4.2.2 Servers

The system will be hosted on web servers that handle requests and store data.

Requirements

- Web server with sufficient processing power and memory to handle user requests.
- Database server for storing user data, videos, written tasks, and system logs.

4.3 Software Interfaces

4.3.1 Web Browser Compatibility

The system should be compatible with major web browsers.

Supported Browsers (Latest Updates)

- Google Chrome
- Safari
- Mozilla Firefox
- Microsoft Edge

4.3.2 APIs for Video Processing

APIs for handling video uploads, processing, and playback.

Examples

- YouTube API or a similar service for video hosting and playback.
- Custom API for video recording and storage.

4.3.3 Email Services

Integration with email services for notifications and password resets.

Examples

- SMTP servers for sending emails.
- APIs for managing email communications.

4.4 Communication Interfaces

4.4.1 Network Protocols

In CurtinTalentTrack, strong network protocols are essential to ensure secure, reliable, and efficient communication between the users and the server. The application will use HTTPS to make the transferring of data secure.

4.4.2 Notification System

Internal messaging and notification system for evaluators and administrators.

Mechanism

- In-system notifications for new review requests.

Email notifications for important updates.

5 Non-Functional Requirements

5.1 Performance Requirements

5.1.1 Responsive Interface

This platform will be designed to ensure that the consumer remains engaged has a responsive layout so that several people can use it at the same time. It also entails factors such as response time delay between actions and minimal interruption when flipping through the pages as well as page loading time.

Mechanism

- Use multiple servers to handle user requests.
- Check the system regularly to fix problems quickly.
- Keep frequently used data ready for quick access.

5.1.2 Efficient Multimedia Handling

Multimedia files such as audio recordings will also be well handled on the platform. This requires low delay while playing the videos, fast uploading and ease in streaming.

Mechanism

- Reduce file sizes to make uploads and streaming faster.
- Use smart database techniques to find and load multimedia quickly.
- Temporarily store frequently used media for quicker access.

5.2 Security Requirements

5.2.1 Secure Login

The measures that shall be put in place to ensure that users' data are secured with the best of securities include features multi-factor authentication (MFA) and encrypted storage of sensitive data and data login.

Mechanism

- Multi-factor authentication (MFA) for added security.
- Regular updates to authentication protocols.

5.2.2 Role-Based Access Control (RBAC)

Using RBAC to manage users, evaluators, and administrators to provide them with adequate access to their functions and to meet their needs.

Mechanism

- Setting up user roles with specific permissions (e.g., Users, Evaluators, Administrators).
- Restricting access to sensitive data based on user roles.

5.2.3 Regular Security Audits

Security testing of the platform will be conducted periodically in order to determine the system's weaknesses. Such a manner is preventive in nature meaning that any possible security risks are identified and addressed before they cause significant harm.

Mechanism

- Conducting periodic security assessments.
- Implementing findings from audits to improve security.

5.2.4 Data Encryption

All personal data and recordings will be ensured by implemented encryption both at the transfer level and when it is stored.

Mechanism

- Regular updates to encryption protocols.
- Encrypting data both in transit (using HTTPS) and at rest (using encryption algorithms).

5.3 Usability Requirements

5.3.1 Intuitive Design

The design of the platform will be simple and easy to use to reduce the complexity of usage for various categories of people. This encompasses simple menu, features that are easily located and content arrangement that is always well structured.

Mechanism

- Using responsive web design techniques to adapt to different devices and screen sizes.
- Testing across various devices to ensure consistent user experience.

5.3.2 Comprehensive Help Resources

Interactive tools and guides in cases when users will need help and would not be able to solve a problem by themselves.

Mechanism

- Create a central place with guides and FAQs.
- Provide easy-to-follow video or text tutorials.
- Offer options for users to chat with support if they need immediate help.
- Implement a searchable library of articles and tips.

5.3.3 User Feedback Mechanism

It has been observed that it is important to integrate a user feedback module in the platform to enhance it using suggestions from the users.

Mechanism

- Adding feedback forms and surveys within the platform.
- Analysing feedback to make continuous improvements.

5.3.4 Search and Navigation

To make a search easier, there are such advanced options as a basic and an advanced search besides regular options like star ratings and tags to help users easily find the best tools and resources.

Mechanism

- Implementing advanced search features with filters.
- Designing intuitive menus and navigation paths.

5.4 Reliability Requirements

5.4.1 High Availability

It will be highly available platform so that it will be almost always online and ready for usage by the users. This includes correct infrastructure and backups/fail-over solutions.

Mechanism

- Regularly testing recovery plans.

5.4.2 Regular Backups

It will always back up data through the platform for the protection of customer data and content. Data backups are created so that in the event of a problem like a hardware malfunction or perhaps a hack; data can be recovered.

Mechanism

- Automated backups of data and system configurations.
- Regular testing of backup restorations.

5.5 Maintainability Requirements

5.5.1 Regular Updates

An important fact to note is that the platform will be updated regularly with new versions to enhance its performance and feature set and to synchronise with emerging technologies. Such routine keeps the platform in check in terms of updates and security among other attributes.

Mechanism

- Scheduled Maintenance (Plan regular times to update the system and fix issues.)
- Update the platform based on user feedback and new needs.

5.5.2 Detailed Documentation

Ensuring that all versions of the code are appropriately documented and recording all the steps to help with easy sending of new developers or maintenance of the code.

Mechanism

- Creating and maintaining comprehensive user guides and developer documentation.
- Updating documentation with each system change.

5.5.3 Modular Design

Ensuring the platform has a modularity that enables changes to be made on one part of a system that does not cause alterations to the whole system.

Mechanism

- Designing the system in separate, interchangeable modules.

Part Two Agile

6. Agile

6.1 Agile Development Process

We adopted agile process throughout planning the CurtinTalentTrack application. It outlines the Agile practices we followed including client interactions, internal team meetings.

6.2 Client Feedback and Meetings

6.2.1 Regular Meeting with Client

We conducted weekly meetings with our client to get feedback and refine our planning process.

- We took meeting minutes of every meeting we had, which were emailed to our client and team member and took feedback and copied to our supervisor as well.
- We collected all our email communications and stored it in a Google Drive. https://drive.google.com/drive/folders/1LpLpZ6vbE6eJdyTUZ4nHk_435HEKvWhO?usp=share_link
- We stored all our meeting minutes in a Google Drive and shared with all group members linked below https://drive.google.com/drive/folders/1Gnv8TSwU4Eo2BxgdB0MmzhgK4sr8RhNT?usp=sharing

The first client meeting was held on 26th July 2024 and our briefed us about features and gave an overview of the required application.

Meeting minutes - Client

Meeting Minutes for Capstone Computing Project 1

Name of meeting/Project	**CurtinTalentTrack: A system to improve communication Skills in Curtin Community**
Day & Date	**26th July 2024**
Time	**1:15 pm - 2.00 pm**
Location	**Curtin Colombo**
Meeting Objective	
Client Meeting 01 – Requirement Gathering	
Attendees	
PRESENT:	1. Anjana Wishwajith Premadasa - 20463382 2. Malith Pramuditha - 20926076 3. Hesara Yasaswin Pathirana – 20928386 4. Rizkan Rhazes - 21309380
ABSENT:	5. Uvindu Gamage - 20528142

Agenda	
Topic	**Discussion**
Project overview	Client gave a brief introduction to the project overview.
User Registration and Profiles	The client highlighted that any stakeholder involved in Curtin community should be able to register as users. Evaluators are registered into the system by admins (Evaluators cannot self-register like users) **Users**: User (Common user), Administrator, Evaluator
Public Speaking and Interview Practice *(First Core Feature)*	The client explained how the section has two parts one being public speaking and other one being interview prep. The public speaking part allows the user to see samples of correct and incorrect speeches as well as tips. The user should also be able to upload their own recording (max 5 mins) of a speech to the system to be evaluated by an evaluator. The client also suggested a load balancing technique to be applied to the evaluator so that the same evaluator does not get all jobs. The second part of this is interview prep, the system allows users to view examples of proper and improper interviews, practice for interviews, and select job categories to receive relevant questions. Users can answer these questions, record their responses, and save them in the system. Same evaluator system applies here.
Written Communication *(Second Core Feature)*	The client explained how the system should provide samples of proper and improper letters, emails, memos, and CVs. Users can practice written communication through activities like writing an email to a superior or creating a memo. They can write their responses in the system and save them. When ready for review, an evaluator is assigned based on load balancing system. Evaluators receive notifications, review the work, and post private feedback below the user's writing.
Critical Thinking and Problem Solving *(Third Core Feature)*	The client explained how the system offers quiz-style scenarios for users to make decisions, answer general problems, and complete critical thinking exercises. Users provide answers, and the system marks them based on correctness. Quizzes can include multiple-choice or structured questions from a question bank. At the end, users receive their scores (# correct/# incorrect). The system features multiple levels of difficulty. No requirement of an evaluator here(All done by system).
Admin Privileges	The client explained how only the admin should be able to add all the sample materials(sample videos, quizzes etc..) in to the system.

<table>
<tr><td></td><td colspan="2">administrators can create and promote events via the system, this includes an event calendar where users can register and get reminders on events.</td></tr>
<tr><td>Dashboard</td><td colspan="2">The client explained how there should be 3 dashboards for each login category.
Admin - All administrative functions and statistics should be included in a dashboard.
Evaluator – All pending, completed jobs should be displayed.
User – All the three core features along with their sub features should be displayed.</td></tr>
<tr><td colspan="3"><b>Action items</b></td></tr>
<tr><td><b>Action</b></td><td><b>Assigned</b></td><td><b>Due Date</b></td></tr>
<tr><td>-</td><td></td><td></td></tr>
<tr><td>-</td><td></td><td></td></tr>
<tr><td>-</td><td></td><td></td></tr>
<tr><td><b>Next meeting</b></td><td colspan="2"><b>TBD</b></td></tr>
</table>

The second client meeting was held the 8th of August 2024 to clarify concerns with the draft of the SRS report. During the meeting, the client reviewed the requirements stated and approved the current version of the SRS to move forward.

Meeting minutes - Client

Meeting Minutes for Capstone Computing Project 1

Name of meeting/Project	**CurtinTalentTrack: A system to improve communication Skills in Curtin Community**
Day & Date	**8th August 2024**
Time	**1:40 pm - 2.10 pm**
Location	**Curtin Colombo**
Meeting Objective	
Showing the Draft SRS to the client	
Attendees	
PRESENT:	1. Anjana Wishwajith Premadasa - 20463382 2. Malith Pramuditha - 20926076 3. Hesara Yasaswin Pathirana – 20928386
ABSENT:	4. Uvindu Gamage – 20528142 5. Rizkan Rhazes - 21309380

<table>
<tr><td colspan="4"><b>Agenda</b></td></tr>
<tr><td><b>Topic</b></td><td colspan="3"><b>Discussion</b></td></tr>
<tr><td>Showing the SRS</td><td colspan="3">Client read the draft SRS and pointed out 2 missing requirements.
Approved the rest of the document</td></tr>
<tr><td colspan="4"><b>Action items</b></td></tr>
<tr><td colspan="2"><b>Action</b></td><td><b>Assigned</b></td><td><b>Due Date</b></td></tr>
<tr><td colspan="2">-</td><td></td><td></td></tr>
<tr><td colspan="2">-</td><td></td><td></td></tr>
<tr><td colspan="2">-</td><td></td><td></td></tr>
<tr><td><b>Next meeting</b></td><td colspan="3"><b>TBD</b></td></tr>
</table>

6.3 Internal Communications and Sprint Planning

6.3.1 Daily Communication

We made a WhatsApp group and comminated through the group to discuss progress and have daily updates on the project.

6.3.2 Meeting Documentation

We uploaded all our group meeting minutes to a Google Drive and also sent to all member and as well as emailed to our client and supervisor.

https://drive.google.com/drive/folders/1Gnv8TSwU4Eo2BxgdB0MmzhgK4sr8RhNT?usp=sharing

6.4 Progress Tracking and Metrics

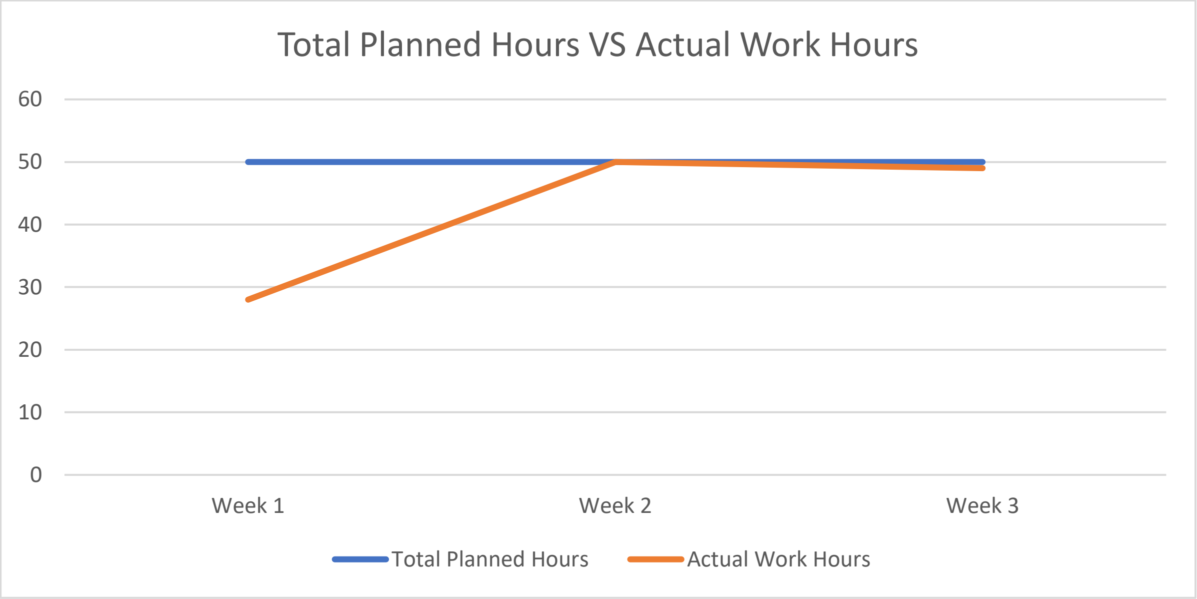
All charts are in a spreadsheet in the Google Drive link below:

https://drive.google.com/drive/folders/1vCffqvIiset-II8YIyZ1WUhkEgIsR4IW?usp=share_link

6.4.1 Time Tracking

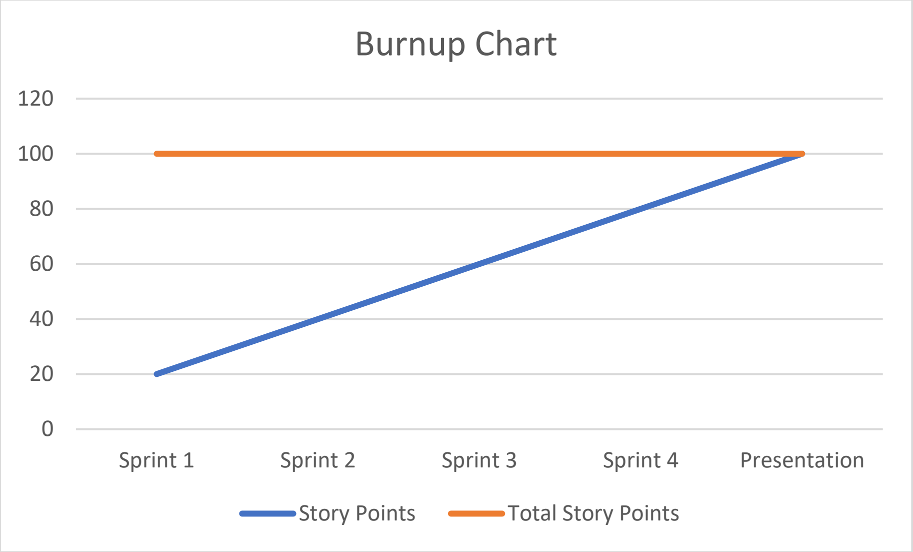
We used togglTrack to keep track the time spent. This allowed us to monitor our progress accurately and ensure we were on track to meet deadlines. All our toggl reports are uploaded to a Google Drive. (Link below)

https://drive.google.com/drive/folders/14qB6keXZ3R2sTrzM8od7fPYuIlJtkWDS?usp=share_link



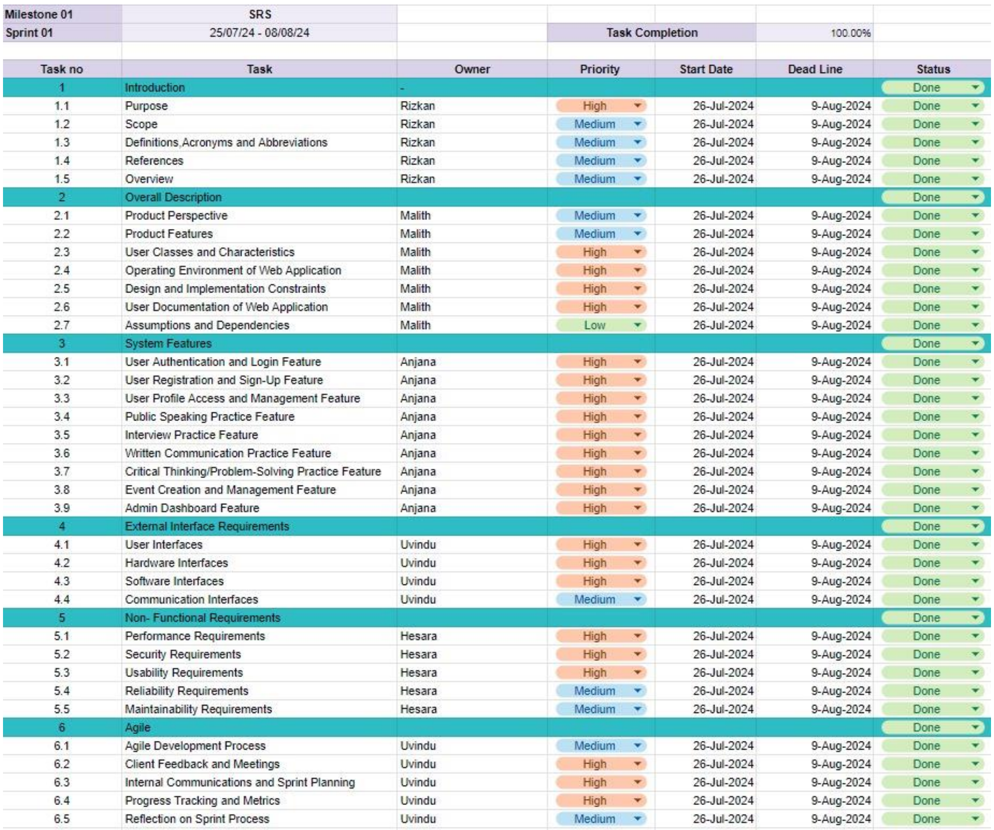
6.4.2 Burn Up Chart

We created a burnup chart to track our progress and gave story points to get an idea on how things are going.



6.4.3 Sprint Backlog

We used the following sprint backlog to give tasks to members and keep track of the progress.



Milestone 01	SRS					
Sprint 01	25/07/24 - 08/08/24		Task Completion		100.00%	
Task no	Task	Owner	Priority	Start Date	Dead Line	Status
1	Introduction	-				Done
1.1	Purpose	Rizkan	High	26-Jul-2024	9-Aug-2024	Done
1.2	Scope	Rizkan	Medium	26-Jul-2024	9-Aug-2024	Done
1.3	Definitions,Acronyms and Abbreviations	Rizkan	Medium	26-Jul-2024	9-Aug-2024	Done
1.4	References	Rizkan	Medium	26-Jul-2024	9-Aug-2024	Done
1.5	Overview	Rizkan	Medium	26-Jul-2024	9-Aug-2024	Done
2	Overall Description					Done
2.1	Product Perspective	Malith	Medium	26-Jul-2024	9-Aug-2024	Done
2.2	Product Features	Malith	Medium	26-Jul-2024	9-Aug-2024	Done
2.3	User Classes and Characteristics	Malith	High	26-Jul-2024	9-Aug-2024	Done
2.4	Operating Environment of Web Application	Malith	High	26-Jul-2024	9-Aug-2024	Done
2.5	Design and Implementation Constraints	Malith	High	26-Jul-2024	9-Aug-2024	Done
2.6	User Documentation of Web Application	Malith	High	26-Jul-2024	9-Aug-2024	Done
2.7	Assumptions and Dependencies	Malith	Low	26-Jul-2024	9-Aug-2024	Done
3	System Features					Done
3.1	User Authentication and Login Feature	Anjana	High	26-Jul-2024	9-Aug-2024	Done
3.2	User Registration and Sign-Up Feature	Anjana	High	26-Jul-2024	9-Aug-2024	Done
3.3	User Profile Access and Management Feature	Anjana	High	26-Jul-2024	9-Aug-2024	Done
3.4	Public Speaking Practice Feature	Anjana	High	26-Jul-2024	9-Aug-2024	Done
3.5	Interview Practice Feature	Anjana	High	26-Jul-2024	9-Aug-2024	Done
3.6	Written Communication Practice Feature	Anjana	High	26-Jul-2024	9-Aug-2024	Done
3.7	Critical Thinking/Problem-Solving Practice Feature	Anjana	High	26-Jul-2024	9-Aug-2024	Done
3.8	Event Creation and Management Feature	Anjana	High	26-Jul-2024	9-Aug-2024	Done
3.9	Admin Dashboard Feature	Anjana	High	26-Jul-2024	9-Aug-2024	Done
4	External Interface Requirements					Done
4.1	User Interfaces	Uvindu	High	26-Jul-2024	9-Aug-2024	Done
4.2	Hardware Interfaces	Uvindu	High	26-Jul-2024	9-Aug-2024	Done
4.3	Software Interfaces	Uvindu	High	26-Jul-2024	9-Aug-2024	Done
4.4	Communication Interfaces	Uvindu	Medium	26-Jul-2024	9-Aug-2024	Done
5	Non- Functional Requirements					Done
5.1	Performance Requirements	Hesara	High	26-Jul-2024	9-Aug-2024	Done
5.2	Security Requirements	Hesara	High	26-Jul-2024	9-Aug-2024	Done
5.3	Usability Requirements	Hesara	High	26-Jul-2024	9-Aug-2024	Done
5.4	Reliability Requirements	Hesara	Medium	26-Jul-2024	9-Aug-2024	Done
5.5	Maintainability Requirements	Hesara	Medium	26-Jul-2024	9-Aug-2024	Done
6	Agile					Done
6.1	Agile Development Process	Uvindu	Medium	26-Jul-2024	9-Aug-2024	Done
6.2	Client Feedback and Meetings	Uvindu	High	26-Jul-2024	9-Aug-2024	Done
6.3	Internal Communications and Sprint Planning	Uvindu	High	26-Jul-2024	9-Aug-2024	Done
6.4	Progress Tracking and Metrics	Uvindu	High	26-Jul-2024	9-Aug-2024	Done
6.5	Reflection on Sprint Process	Uvindu	Medium	26-Jul-2024	9-Aug-2024	Done

6.5 Reflection on Sprint Process

6.5.1 Reflections

Over the course of the sprint our team improved estimated task duration as shown in the charts above. For the 1st week of sprint 1, the time spent on work was low of all the members as it was the first week starting up and getting to know the team member. Furthermore, a new group member was added to the group in the first week of the sprint. Afterwards we improved our pace of working and got tasks completed as per the estimated time. Moreover, we could not get time track data as a collective of the whole group since it has been made a paid feature in toggl.

6.5.1 Improvements

Moving forward, we will utilize new tools and have more meeting to streamline our project.

Part Three - Attributions

Attributions (Only Documentation).

Task No	Task	Anjana	Hesara	Malith	Rizkan	Uvindu
1	Introduction					
1.1	Purpose				■	
1.2	Scope				■	
1.3	Definitions, Acronyms, and Abbreviations				■	
1.4	References				■	
1.5	Overview				■	
2	Overall Description					
2.1	Product Perspective			■		
2.2	Product Features			■		
2.3	User Classes and Characteristics			■		
2.4	Operating Environment of Web Application			■		
2.5	Design and Implementatio n Constraints			■		
2.6	User Documentation of Web Application			■		
2.7	Assumptions and Dependencies			■		

3	System Features					
3.1	User Authentication and Login Feature	■				
3.2	User Registration and Sign-Up Feature	■				
3.3	User Profile Access and Management Feature	■				
3.4	Public Speaking Practice Feature	■				
3.5	Interview Practice Feature	■				
3.6	Written Communication Practice Feature	■				
3.7	Critical Thinking/Proble m-Solving Practice Feature	■				
3.8	Event Creation and Management Feature	■				
3.9	Admin Dashboard Feature	■				
4	External interface Requirements					

4.1	User Interfaces					■
4.2	Hardware Interfaces					■
4.3	Software Interfaces					■
4.4	Communication Interfaces					■
5	Non-Functional Requirements					
5.1	Performance Requirements		■			
5.2	Security Requirements		■			
5.3	Usability Requirements		■			
5.4	Reliability Requirements		■			
5.5	Maintainability Requirements		■			
	Attribution table		■			