1. **Technological Requirements**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Chosen Technology** | **Pros** | **Cons** | **Why Best Fit for This Project** |
| **Frontend** | ReactJS | * Light Weight. * Maintained by Facebook group. * Built upon popular Java Script library, Modular. * Easily integrates with Axios library for http request. * Several templating engines are available * Works in component level. * Does not require the complete page reload when contents are modified. * Browser compatible. | * As it is a library, we need to import third party modules for better programming experience. * As it works on component level, difficult in SEO optimization. * Needs third party libraries to enhance the features. * Difficult to keep track of components if many of them are implemented. | * Great for developing complex web application. * Can implement freely available libraries. * Uses JavaScript, which most of us are familiar with. * Can use both the function and class components according to our ease. * Modular, so every member in the group can easily share their contribution. * Seamlessly compatible with popular browsers. |
| **Backend** | Node.JS | * Made from popular JavaScript language. * Scalable. * Offers the development of asynchronous APIs, which allows background processing for processes that take time. * Large Active Community support. * Data exchange in JSON format, that can easily be handled in frontend. * Seamlessly integrates with ReactJS – a JavaScript front end development tool. * Can reduce time loading by quick caching. | * Reduces performance when handling Heavy computing tasks. * Asynchronous programming sometimes become difficult to maintain code. | * Ours is a simple web based temporary workspace management project, does not need heavy transactions to carry out so it has a better performance. * Easy to design APIs that can be used in ReactJS to consume and provide data. Can run on any of the popular OS based servers like Linux, Windows, and others. |
| **Database** | MySQL | * Owned by Oracle though still a kind of open-source database. * It is a relational database. * Important information can be hashed and stored in tables. * Provides security of the data. * Data migration is easy. * Light weight relational database management software. * Works on almost all the popular server architecture. * Uses SQL, a popular database language. | * Although it is best for RDBMS databases, difficult to handle big data. * Issues with complex SQL operation (hash). * Data replication and backup is difficult. * Suitable for small database needs. * Database optimization now needs money. * Not suitable for big data. * Limited capabilities for stored procedures. | * We found MySQL to be the best fit for our project due to the nature of the data our project will be handling and the team members’ existing proficiency in it. * We are developing a medium-scale web project that will not store huge amount of complex data. It also does not require ‘critical’ performance constraints. Thus, MySQL is best suited in this scenario. * The database can easily be scaled up or can be migrated to other popular database in future if needed. * We will not have complex data. Therefore, the performance of the database will not decrease. * MySQL is free, robust, and very well documented open-source database management system. * MySQL supports many interfaces and programming languages, and it is easier to learn. It follows SQL syntax which most of us are familiar with. |
| **Programming Languages** | JavaScript | * JavaScript is relatively simple to learn and implement. * JavaScript can be used in both front-end and back-end. * It is widely used and has great support to integrate it in a huge variety of applications. * Client-side JavaScript is very fast as it saves the time required to connect to the server. * JavaScript offers wide range of frameworks and libraries for making all type of applications, because it is very popular and that is why has great supply of libraries coming out in the market constantly. It also has a very strong community support for the same reason. | * JavaScript runs on client side and can be used for malicious purposes. * Different browsers interpret JavaScript differently, programmers should be equally watchful on it. | * Most of us are familiar with the technology. * Easy to generate codes and huge support in the web in case we have difficulties. |
| **Frameworks** | We are using different libraries and engines like ReactJS, Axios, Express, EJS, and Bootstrap | * Express is a popular backend framework for Node.js. * Axios simplifies working with RESTful APIs. * EJS is a templating engine to develop the view. | * JavaScript libraries release frequent updates. As we are using several JavaScript third-party libraries and frameworks, if any of these libraries needs updates in future, then there is always a chance of breaking the dependencies. Some libraries might no longer support the compatibility with all the other units we are using. Fixing and/or rebuilding of several units of the application might be needed due to this and that might bring unanticipated amount of effort. | * Several frameworks, libraries and templating engines make our job easier and help us finish our task with high standard coding practices in a limited time. |
| **Hardware** | Platform independent | * JavaScript is a platform independent technology. We can work on all the major hardware and OS platforms. |  | * As we are using platform independent scripting language, our application can run on any OS and hardware platforms. |
| **Version Control** | Git | * Git is free and open source. * Fast and works seamlessly on all major platforms. * Ability to create branches and merge those changes cleanly and in an organized way makes team collaboration easier. * Each developer has their own local copy of branches, so it minimizes mistakes being made. * Rollback is extremely useful if something goes wrong and we can go back to the last stable version. * Git provides a complete history of all code changes. So, it is a powerful tool for change management and avoiding breaking code making its way into production. | * Git has a little bit more of a learning curve compared to other source control solutions, especially the advanced features and functionalities, which takes time to master. * Some specific situations can arise where it is difficult to figure out the right way to do things, for example merging conflicts. * If any of the members does not follow the agreed upon Git-flow properly, it can disrupt the codebase and can be a hassle to fix the central repository. | * Since we are developing the project in a group, we need a common repository, that can keep track of our submissions and change history. Git is the perfect choice for this because all of us can work separately and push our works in the central repository. * Despite having some cons, Git is so far the best version controlling protocol in industry, that is intuitive, and provides the most needed features to maintain stable releases. So, this is the best suited technology for our project. |

1. **Learning Plan**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **Responsibility** | **Existing Skill Level** | **Learning Plan** | **Start Date** | **End Date** | **Resources** |
| **Kamrun Nahar Liza** | Product Owner  Middleware | 60%  60% | Take online courses | 18/12/2020 | 28/01/2021 | Online courses  Publisher documentations |
| **Sudeep Manandhar** | Scrum Master  Front-End | 70%  60% | 18/12/2020 | 28/01/2021 |
| **Rajesh Bista** | Front-End | 75% | 18/12/2020 | 28/01/2021 |
| **Suvash Sharma** | Back-End | 75% | 18/12/2020 | 28/01/2021 |
| **Shishir Das** | Back-End  Quality Control | 60%  70% | 18/12/2020 | 28/01/2021 |