Project Backlog
Online Resource Management Software

Team Members : Aritra Samanta**,** Cody Phrampus, Hemil Desai, Pranav Punjabi

Problem Statement

As teams within an organization begin to grow, the amount of online resources shared by the team members rise with each passing day. The resources can range from a link to a file; and managing and categorising these resources can sometimes be a mess. It can get increasingly difficult to refer back to a resource unless the user knows exactly what they are searching for.

Background Information

While working in a large organization, developers and other company personnel often stumble upon resources that could be used for future use or is worth reading. Our platform will provide the user with complex search and organization tools for all their resources.

AUDIENCE

Our target segment consists of users who work in teams spread across organizations as well as individual users working on open source or personal projects. With our application, we plan to provide ways to save and search resources for all users, essentially providing a service to meet the requirements of all organizations through an essential platform that is not only user friendly but also has great functionality.

SIMILAR APPLICATIONS

There are many team collaboration and communication platforms used commonly across various organizations and industries. These include communication apps like **Slack**, task management apps like **Trello** and wiki apps like **Confluence**. All these can be used to indirectly share resources and links, but there is no way to organize or search through the shared resources. As time goes by, most people are lost trying to find those resources shared in the past and there is no reuse. Our platform provides a way to organize and search resources for reusability and efficiency in teams, and integrates seamlessly with the platforms already in use.

EXISTING LIMITATIONS

Currently there are a lot of team collaboration tools, but none of them have the ability to organize and search resources. Team members always find themselves searching for links or files that were shared in the past, but might be useful later. Our system tries to solve this problem, making information reusable across teams and also increasing the efficiency of different teams.

ENVIRONMENT

The backend API will be built using Node.js and express. JSON web tokens will be used for authorization and access to the API. The API will communicate with a database server. MySQL or postgreSQL will be used to provide persistent storage. The API will use Knex (Query Builder) and Bookshelf (ORM) to abstract the database communication. The front end web app will be built using modern frontend frameworks (React or Angular). A chrome extension will be built using JS for easy access to the platform. There could potentially be more front end clients consuming the API, but only if time permits.

Functional Requirements

1. As a user, I would like to be able to login and logout of my account.
2. As a user, I would like to be able change my password for my designated account.
3. As a user, I would like to be able to save links as a resource.
4. As a user, I would like to be able to share links with other team members.
5. As a user, I would like to be able to tag links.
6. As a user, I would like to be able to search for links using tags or simple commands.
7. As a user, I would like to be able to filter my search results based on tags.
8. As a user, I would like to be able to search based on link text.
9. As a user, I would like to be able to search by teams and projects.
10. As a user, I would like to search for links based on the user who added it.
11. As a team member, I would like to be able to see the other members of my team(s).
12. As an admin user, I would like to be able to add users to a team I am an admin for.
13. As an admin user, I would like to be able to create teams.

14. As an admin user, I would like to be able to create project threads.
15. As an admin user, I would like to be able to add user to a team.
16. As an admin user, I would like to be able to remove users from a team.
17. As an admin user, I should have access to all information of teams below me.
18. As a user, I would like to be able to create a sub-thread(collection) for specific topics within a project.
19. As a user, I should not be able to see/search information that is not accessible to me.
20. As a user, I would like to have a tab view associated for each team I am a part of.
21. As a user, I should be able to see a preview of the resource I am being linked to.
22. As a user, I shall be able to add a description to the link I am adding.
23. As a user, I shall be able to link documents of all file formats hosted on an online service.
24. As a user, I shall be able to favorite a link for future reference.
25. As a user, I shall be able to update the link, description or the tags associated with it at any time.
26. As a user, I would like to add comments to an already added link in the repository.
27. As a user, I would like to receive notifications when a new link is added to the team(s) I am a part of.
28. As a user, I would like to be able to mute notifications for the team(s) I am a part of.

Non Functional Requirements

1. We must be able to use the software as a web application.
2. The interface must provide flexibility and be user friendly on different browsers on different devices.
3. We must be able to use the software as an iOS application, if time permits.
4. As a developer, I would like the APIs to have publicly available documentation.
5. Must be able to integrate with Slack, if time permits.
6. As a developer, I would like to use the server - client model in our application.
7. As a developer, I would like to use Node.js for backend development.
8. As a developer, I would like to use a local server to store data.
9. As a developer, I would like to use Express as a framework to develop the web application.
10. As a developer, I'd like to use MySQL as our database software.

11. As a developer, I would like to use JSON web tokens for authorization and access to APIs.
12. As a developer, I would like to use GitHub as a repository.
13. The system should have fast response times and efficient latency and bandwidth. The system should have an online response time of no more than 10 seconds.
14. The API should be efficient. If deployed on cloud, it should integrate well with services like Amazon AWS or Microsoft Azure.
15. The server must be able to handle concurrent users.
16. The app should aim to have minimum downtime.
17. The API and database should be scalable to handle occasional overloads and several users.