Software Requirements

This document outlines the functional and non-functional requirements for PLUMP

Functional Requirements

Functional requirements describe the specific actions or functions that the system must be able to perform.

1. The system must allow users to create a unique account using an email address.
2. Each user account must have a unique system-generated identifier in addition to the email address.
3. Users must be able to initiate the account creation process by providing their organization email address.
4. The system must verify the organization affiliation of the provided email address during registration.
5. New user join requests must be sent to team administrators for approval.
6. Users will only gain access to the application upon approval from a team administrator.
7. The system must support user authentication via login credentials (email and password).
8. User passwords must be stored using JWT encryption for security.
9. Users must be able to change their password after logging in.
10. The system must implement a secure logout feature that terminates the user's session.
11. Users must be able to delete their own account and all associated data through a 'Danger Zone' feature.
12. Confirmation must be required before a user account and data are permanently deleted.
13. The system must maintain a distinction between user roles: Team Member, Project Lead, and Project Manager.
14. User roles must determine the level of access and permissions within the application.
15. A user's access level and role(s) must be clearly displayed within their profile or relevant sections of the application.
16. Users can have varying levels of privileges for different projects based on their assigned role within each project.
17. The system must ensure that varying access levels and roles across different projects work seamlessly together without conflicts.
18. Project Managers must be able to create new projects within the system.
19. When creating a project, Project Managers must be able to define the project name, description, and initial phase.
20. Project Managers must be able to assign Project Leads and Team Members to projects during or after project creation.
21. Each project must have a designated Project Lead.

22. Projects must have a visible 'phase' indicator, displaying the current stage of the project.
23. Project Leads must be able to update the project phase as the project progresses.
24. Each project must include sub-charters to store specific project-related information.
25. Users must be able to access and view the sub-charters for projects they are part of.
26. A resource planning feature must be available for each project.
27. The resource planning feature must allow teams to allocate and manage various types of resources.
28. Project health status must be visible for each project.
29. Project Leads must be able to update the project health status to indicate progress and potential issues.
30. The project health status should provide a visual indicator (e.g., color-coded) of how close the project is to meeting expectations.
31. A project dashboard feature must allow users to search for projects they are involved in.
32. The project dashboard search should provide quick details about the retrieved projects.
33. Project Managers must have the ability to see all active projects within the organization.
34. Team Members and Project Leads can only see the projects they are explicitly assigned to.
35. Each project must include a task list.
36. Members must be able to generate new tasks within a project's task list.
37. Project Leads must be able to assign tasks to team members within the project.
38. Project Leads must be able to provide all necessary details for each task, including description, due date, and priority.
39. Task management features must include the ability to assign priority levels to tasks.
40. Tasks must have fields for specifying an expected completion date and an actual completion date.
41. Users assigned to a task must be able to mark the task as completed.
42. The system must allow for listing the team members who worked on a specific task.
43. Users should be able to view tasks assigned to them.
44. Users should receive notifications for new task assignments or updates to assigned tasks.
45. Tasks can be linked to specific project phases or sub-charters.
46. The allocated budget for each project must be clearly visible.
47. The system must allow for tracking expenses against the allocated budget for each project.
48. Users with appropriate permissions should be able to input project expenses.
49. Budget allocation and spending should be visible.

50. Visual representations (e.g., charts, graphs) of budget allocation and spending should be available.
51. The resource planning feature should integrate with budget tracking to show resource costs.
52. A 'timesheet' feature must be available for relevant users to submit hours worked on specific projects.
53. The timesheet feature must be user-friendly and intuitive.
54. The timesheet feature must allow users to select the specific project they worked on.
55. Users must be able to enter the number of hours worked for a given project and date.
56. The timesheet feature should allow for overwriting or adding on to hours.
57. Users should be able to update and add to their time entries on the fly.
58. The timesheet data should be used by employers to keep track of working hours for compensation purposes.
59. Project Leads and Project Managers should be able to view the aggregated time entries for team members on their projects.
60. The system should provide reports on logged hours per project and per user.
61. The 'home' page of the application, shown after a user logs in, must provide a summary of all projects the user is working on.
62. The homepage summary should include key information for each project, such as project name, current phase, and perhaps pending tasks or deadlines.
63. The homepage must provide clear links to navigate into each project listed in the summary.
64. The homepage layout and content must be different based on the user's hierarchy and role (e.g., Project Manager vs. Team Member).
65. Project Managers' homepage should display a summary or list of all active projects within the organization.
66. Team Members' homepage should only display a summary or list of projects they are assigned to.
67. A clear navigation menu should be available on all pages to easily access different sections of the application (e.g., Home, Projects, Calendar, Team).
68. A dedicated 'team page' must provide information about all members assigned to a relevant project.
69. The team page should display basic information about each team member (e.g., name, role in the project).
70. Team Leads and Project Managers should be able to view the roles and assignments of all team members on their projects.
71. Users should be able to view the team members and their roles for projects they are part of.
72. A comprehensive calendar feature must be integrated into the application.

73. The calendar should display tasks due on any particular date for the logged-in user.
74. The calendar should show arranged meetings related to the user's projects.
75. The calendar should display project deadlines.
76. Users should be able to view their personal schedule, including tasks, meetings, and deadlines, within the application calendar.
77. The calendar should support different views (e.g., daily, weekly, monthly).
78. Users should be able to create new meeting events directly within the application calendar.
79. Users should receive notifications for upcoming tasks, meetings, and deadlines via the in-app calendar or notifications feature.

Non-Functional Requirements

These requirements define system performance, security, and usability expectations.

1. The application database must perform regular automated backups to prevent data loss.
2. The database backup strategy should allow for point-in-time recovery.
3. The user interface (UI) must be fully responsive and function seamlessly across different browsers and devices.
4. The software architecture must be scalable to handle an increasing number of users, projects, and data without significant performance degradation.
5. The system must maintain high reliability and function consistently as intended without frequent crashes or errors.
6. The software must be secure, implementing measures to protect user anonymity and data privacy.
7. Security measures should include protection against common web vulnerabilities (e.g., XSS, SQL injection).
8. The system should have clear and informative error messages for users.
9. The application should have a consistent design and user experience across all features.
10. Performance benchmarks should be established and monitored to ensure the application remains responsive under load.
11. Access control mechanisms must be strictly enforced based on user roles and project assignments to prevent unauthorized access to data or features.