ReqTrackManager

Project Specifications

Table of contents

1 Introduction

ReqTrackManager is aiming to fill the market gap for teams (predominatly product development teams) that need a formal engineering requirements management system, but cannot afford propietory systems, and find other requirements software “not quite right” (generally due to lack of formal change management).

A Formal Engineering Requirements Management System (ERMS) is a structured process and associated tools designed to manage and document the requirements of an engineering project. It ensures that all stakeholders have a clear understanding of what the product should do, how it should perform, and under what conditions it should operate.

So what is a Formal Engineering Requirements Management System?

- Requirements Documentation
  - This involves the systematic collection and documentation of all functional, non-functional, performance, regulatory, and interface requirements for a product.
- Change Management
  - Processes to handle changes in requirements. This includes evaluating the impact of changes, documenting them, and updating the relevant documentation and stakeholders.
- Stakeholder Collaboration
  - Facilitating communication and collaboration among all stakeholders (for example, business leaders, engineers, designers and users).
- Traceability
  - The ability to track each requirement from its origin through the design, and development. This ensures that all requirements are met and any changes are managed effectively.
- Verification and Validation
  - Ensuring that the product meets the specified requirements (verification) and that the final product fulfills its intended purpose (validation).

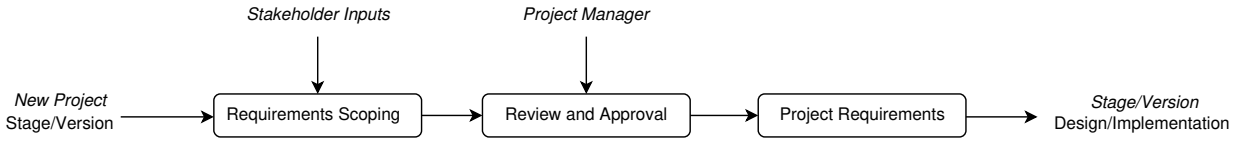
1.1 How it works

Requirements projects are organised by project component and category and also split by project version. Projects sit within an organisation.

	Component A		**Component B**	
	Category α	**Category β**	**Category γ**	**Category δ**
Stage/Version 1	Requirement 1 Requirement 2 ... Requirement n	Requirement 1 Requirement 2 ... Requirement n	Requirement 1 Requirement 2 ... Requirement n	Requirement 1 Requirement 2 ... Requirement n
Stage/Version 2	Requirement 1 Requirement 2 ... Requirement n	Requirement 1 Requirement 2 ... Requirement n	Requirement 1 Requirement 2 ... Requirement n	Requirement 1 Requirement 2 ... Requirement n

This allows ease of management of the project’s lifecycle while similtaniously accommodating the complexity of requiements and their sequential nature.

For each project version, there is a requirements scoping stage, then these get reviewed and refined and finally approved to form the actual project requirements.

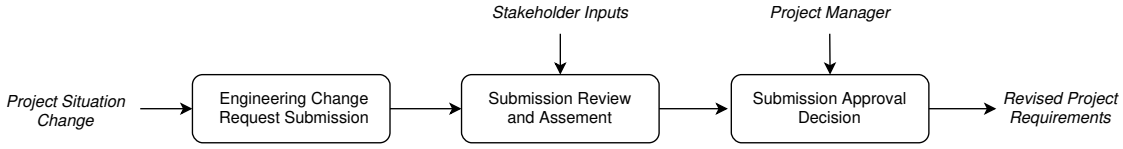


During the scoping stage of the project, requirements can be added without change requests, and by any user that is authorised to add requirements. By default, this is limited to project managers, project administators and project stakeholders.

Once the requirements for a particular version has been approved, all changes to the requirements must go through a change management process.

The change management process consists of sumbitting a change request, having the change request reviewed, and based on the review, either being accepted and approved or being rejected. If the change is approved, the project requirements are updated.

The submission must include what is being requested (a new requirement or a change to an existing requirement), and all required attributes. There must also be a reason for why this submission is being made, and wasn't identified in the original scope or previous project requirements.



1.2 Background

Hardware teams that I have been apart of, have lacked proper engineering requirement systems and formal change request processes.

The main reason for this is cost. IBM DOORS is arguably the industry standard for this type of software, but it comes with a heafty cost. Small to medium size teams simply can't afford it, and so either try alternatives (such as Jira Requirements), manual process or go without.

However due to the complexity of tracking requirements, most alternatives simply don't have all the features required by hardware teams (they are developed for agile software). Manual processes seemingly end up in the too hard basket and as such teams generally revert back to a static document which doesn't change throughout the project and only gets revised at the end (generally changing the requirements to match what was made).

As such, I wanted to develop my own system that I could work on in my spare time, using industry best practices and looking like a modern software package.

2 Core Functionality Specifications

2.1 General

ID	Requirement	Level	Target
C-G-01	User Authorisation is needed for login **Reasoning:** This feature enables administrators to enforce access controls and permissions based on user roles, safeguarding sensitive engineering requirements and project data. Implementing user authorization enhances accountability and traceability within the system, promoting a secure and compliant environment for managing requirements. **Clarification:** To view or perform any data modification, a user must authenticate into the the system with an authorised user crediential	Requirement	Ossa (v1)
C-G-02	Users must have the ablility to create, edit, and delete requirements. **Reasoning:** This is a key feature of an Engineering Requirements System **Clarification:** A users ability to perform these actions is to be limited by their permissions roles	Requirement	Ossa (v1)
C-G-03	Users must have the ability to create, edit, and delete engineering change requests. **Reasoning:** This is a key feature of an Engineering Requirements System **Clarification:** A users ability to perform these actions is to be limited by their permissions roles	Requirement	Ossa (v1)
C-G-04	Requirements must be sorted by Projects (top level), project components and categories. **Reasoning:** Sorting requirements by projects, project components, and categories enhances clarity and organization, making it easier for stakeholders to navigate and understand the scope of each project. It improves traceability and impact analysis, allowing for efficient management and communication. This structured approach ensures compliance with standards and facilitates effective collaboration and productivity.	Requirement	Ossa (v1)
C-G-05	There must be no artificially imposed limit on the number of projects, components, categories, requirements or change requests that can be created. **Reasoning:** This ensures scalability and accommodates varying user needs, from small teams to large organizations. This flexibility supports the growth and dynamic nature of engineering projects, allowing users to manage multiple projects without restrictions. It ensures the system remains versatile and widely applicable across diverse use cases. This should also be more appealing to users. **Clarification:** Limitations may still occur due to deployment characteristics such as memory or storage limits.	Requirement	Ossa (v1)
C-G-06	Requirements must have unique identifiers within the project that they reside **Reasoning:** Having unique identifiers for requirements within a project ensures precise tracking, referencing, and management of each requirement, eliminating confusion and reducing errors. Unique identifiers	Requirement	Ossa (v1)

ID	Requirement	Level	Target
	facilitate clear communication among team members and stakeholders, allowing for unambiguous discussions and documentation. **Clarification:** This incudes any archieved (removed) requirements.		
C-G-07	Project components and categories must have settable identifiers which get used as prefixes for requirement unique identifiers. **Reasoning:** This ensures clear and organized tracking of requirements within a project. This feature enhances the traceability and categorization of requirements, making it easier to identify and manage related items. By providing a structured and consistent naming convention, the system improves clarity and efficiency in navigating and referencing requirements, which is crucial for effective project management and communication.	Requirement	Ossa (v1)
C-G-08	Each project must have the ability to have multiple project stages. **Reasoning:** This enables detailed tracking and management of the project's lifecycle, accommodating the complexity and sequential nature of engineering processes. This feature helps in organizing tasks, monitoring progress, and ensuring that requirements are met at each stage before moving forward, enhancing project control and quality assurance.	Requirement	Ossa (v1)

2.2 User Management

ID	Requirement	Level	Target
C-U-01	Organisations must have at minimum the permission roles of: Organisation Administrator, Project Creators, Member **Reasoning:** These roles provide a structured way to manage permissions and responsibilities, ensuring that users have appropriate access based on their involvement in the organisation. This prevents unauthorized access and enhances overall project management and governance. **Clarification:** Organisational Admins can manage properties of the organisation, manage projects (but is not guarenteed project access), and can create projects. Project Creators can create projects. Members roles provide general access, but cannot create projects. No Project access is guarenteed from any roles (apart from org admin being able to manage project settings)	Requirement	Ossa (v1)
C-U-02	All Project users, must be an organisation user. **Reasoning:** This allows easy determination of the users that are using the system. This also allows ease of removing a user from all access.	Requirement	Ossa (v1)
C-U-03	Projects must have at minimum the permission roles of: Project Manager, Project Administator, Stakeholder, Member. **Reasoning:** These roles provide a structured way to manage permissions and responsibilities, ensuring that users have appropriate access based on their involvement in the project. This prevents unauthorized access and enhances overall project management and governance. **Clarification:** A project administators can perform the following tasks: project setup, adding project versions/stages, modification of components and categories, and generate reports. Stake holders can add requiements during the scoping stage; submit, view and provide	Requirement	Ossa (v1)

ID	Requirement	Level	Target
	feedback for change requests and view and generate reports. Members view requirements and generate reports only. Project Managers, can perform all project administator and stakeholder tasks. Project Managers can also provide approvals for change requests and approval of project requirements from scoping review to project requirements.		
C-U-04	Users must be able to be deactivated. **Reasoning:** Deactivating users is essential for security and access control, ensuring that former team members or users who no longer require access cannot compromise sensitive data or functionalities.	Requirement	Ossa (v1)
C-U-05	Deactivated Users can be archieved, such that they no longer show as users, but their previous contributions remain attributed to them. **Reasoning:** preserves the integrity of the project's historical data and maintains transparency in the system's records. Retaining their contributions ensures that past work and insights are accessible for reference and analysis, even after a user's account is deactivated, aiding in continuity and knowledge retention within the engineering team.	Requirement	Ossa (v1)
C-U-06	The implementation of users, must allow for different backends to be used. **Reasoning:** This allows future expanability beyond basic in-built users, such as for single sign-on users.	Requirement	Ossa (v1)
C-U-07	Users in an organisation can be grouped (organisation groups). **Reasoning:** Groups simplifies administration, especially in large teams, by enabling bulk permissions management. Organisational groups can be used to easily group users over many projects.	Requirement	Ossa (v1)
C-U-08	A Project must have at least one project manager user. **Reasoning:** A project needs at least one person to run a project. If there is only one project member then they must be able to manage the project.	Requirement	Ossa (v1)
C-U-09	If a project has all users removed from the organisation, then the user who removed them, becomes the project manager on the projects for which would otherwise lack any users. **Reasoning:** A project needs at least one person to run a project. If there is only one project member then they must be able to manage the project.	Requirement	Ossa (v1)
C-U-10	On creation of a new project (unless using a template project), there should be project groups created for each standard project permission roles, and the project creator should default to being in the project manager group. **Reasoning:** This structure provides a clear framework for permissions and responsibilities from the outset, facilitating effective project management. Defaulting the creator to the project manager role ensures there is an initial accountable person to oversee and coordinate project activities.	Requirement	Ossa (v1)
C-U-11	Groups can be used to manage user authorisations for each project (project groups)	Requirement	Ossa (v1)

ID	Requirement	Level	Target
	Reasoning: Groups simplifies administration, especially in large teams, by enabling bulk permissions management. Project Groups can be used to easily add many users with the same permissions.		
C-U-12	Organistion Groups can be used inside Project Groups. **Reasoning:** Organisational groups can be used to easily group users over many projects. **Example Use:** This feature can be used as if the organisation groups are in fact teams (such as development team and client relations teams).	Requirement	Ossa (v1)
C-U-13	The ability of a project “member” to submit a change request can be enabled or disabled in the project configuration. **Reasoning:** ensures that change management is controlled and aligned with the project's governance policies. This capability prevents unauthorized or excessive change requests, maintaining project focus and reducing administrative overhead. For example, when an external consultantancy has access to a project, it might be desireable for them to have read-only access. **Clarification:** This should default to enabled.	Recommended	Pelion (v2)
C-U-14	There must be the ability to enable two-factor authentication (2FA) for users not using SSO **Reasoning:** This enhances the security of user accounts by adding an extra layer of protection against unauthorized access. Users can benefit from modern security practices, reducing the risk of breaches and data loss.	Requirement	Pelion (v2)
C-U-15	Users can be in multiple organisations. **Reasoning:** This allows: 1) Organisations may be used as teams for some deployments; 2) Helps with multi-tenant deployments where a single user could be part of multiple organisations (eg contractors)	Recommended	Pelion (v2)
C-U-16	Users with organisational admin role, must have the ability to lock display names of users in their organisation. **Reasoning:** This ensures consistency and professionalism in user identification across the system. This feature prevents unauthorized or whimsical changes that could lead to confusion, misidentification, or lack of accountability within the organization. By maintaining control over display names, organizational admins can enforce naming conventions and standards, enhancing clarity and communication among users.	Recommended	Pelion (v2)
C-U-17	Users must have email, a display name and a password. **Reasoning:** This ensures secure and identifiable access to the system, enhancing both security and accountability. The email allows for communication and password recovery, while the display name provides a consistent identity for collaboration and tracking activities within the system.	Requirement	Ossa (v1)
C-U-18	Users may have the ability to set their pronouns and an avatar. **Reasoning:** This fosters an inclusive and personalized user experience, supporting diverse identities and preferences within the system.	Recommended	Pelion (v2)

2.3 Reviewing

ID	Requirement	Level	Target
C-R-01	There must be the ability to have discussion threads with comments on requirements and change requests. **Reasoning:** This allows a central location for discussions and helps with understanding reasoning, issues and intracacies of a requirement or change request	Requirement	Ossa (v1)
C-R-02	Tasks should be able to be assigned during a change request's review stage. **Reasoning:** This allows reviews to include investigations or simply requesting a user to join the review discussion	Recommended	Massif (v3)
C-R-03	Stakeholders should be able to vote on change request approval. **Reasoning:** ensure that all invested parties have a say in significant project changes, promoting collaborative decision-making. This process helps balance differing interests and perspectives, leading to more informed and accepted decisions. Allowing stakeholder votes also enhances transparency and accountability in the change management process.	Recommended	Massif (v3)
C-R-04	Tasks should have the ability to have due dates set. **Reasoning:** This ensures that project timelines are effectively managed and deadlines are met, enhancing overall project scheduling and accountability. By setting due dates, the system provides clear expectations and facilitates timely completion of tasks, contributing to the successful execution of projects.	Recommended	Massif (v3)
C-R-05	Once a project stage has entered review after being scoped, a deadline can be set for stakeholders to provide a resonse review, after which time if they have not provided a response, then it’s assumed they have approved the requirements. **Reasoning:** This feature establishes clear expectations and accountability, encouraging stakeholders to engage and respond within the specified timeframe. Assuming approval if no response is provided streamlines the review process, reducing bottlenecks and enabling smoother transitions between project stages.	Recommended	Massif (v3)

2.4 Auditing

ID	Requirement	Level	Target
C-A-01	Dates and Time of creation of requirements must be logged. **Reasoning:** This allows a timeline of changes to be derived. This can also be used for project auditing.	Requirement	Ossa (v1)
C-A-02	Dates, Times and any attribute modifications of any change of a requirement must be logged. **Reasoning:** This allows a timeline of changes to be derived. This can also be used for project auditing.	Requirement	Ossa (v1)
C-A-03	Dates and Times of any creation, of change of engineering change requests must be logged	Requirement	Ossa (v1)

ID	Requirement	Level	Target
	Reasoning: This allows a timeline of changes to be derived. This can also be used for project auditing.		
C-A-04	Dates, Times and any attribute modifications of any change of engineering change requests must be logged. **Reasoning:** This allows a timeline of changes to be derived. This can also be used for project auditing.	Requirement	Ossa (v1)
C-A-05	Dates, Times and creation/modifications of any change of a organisational or project group must be logged. **Reasoning:** This allows a timeline of changes to be derived. This can also be used for project auditing.	Requirement	Ossa (v1)
C-A-06	If a requirement is removed, it is archieved such that the timeline of the requirement is still maintained. **Reasoning:** This allows a timeline of changes to be derived. This can also be used for project auditing.	Requirement	Ossa (v1)
C-A-07	Dates, Times, IP address and location of IP address of user log-ins must be logged. **Reasoning:** This helps with tracking any potential malicious activity.	Requirement	Ossa (v1)
C-A-08	Dates and Time of when a notification was read must be recorded. **Reasoning:** This helps with determining if a user has interacted with a requirement, such that it can be determined if they had reviewed it. This helps if there is a disagreement on if a user claims they had not seen a requirement.	Requirement	Ossa (v1)
C-A-09	The UI must show a change log for requirements and change requests. **Reasoning:** This ensures transparency and traceability of modifications. This allows stakeholders to understand the history and rationale behind changes, facilitating better decision-making and accountability. It also helps in auditing and resolving disputes by providing a clear record of who made changes and when. **Clarification:** The UI should not have comments in discussion thread in this change log.	Requirement	Ossa (v1)
C-A-10	The UI should have the ability to see project changes over time. **Reasoning:** This ensures transparency and traceability of modifications. This allows stakeholders to understand the history and rationale behind changes, facilitating better decision-making and accountability. It also helps in auditing and resolving disputes by providing a clear record of who made changes and when. **Clarification:** The UI can have a filter attached to this view, such as to allow a time period to be specified. Display changes in discussion threads should be optional, and determined by filters.	Requirement	Pelion (v2)
C-A-11	On creation of a requirement, a project managers should be able to assign the creator of the requirement to another user **Reasoning:** This ensures flexibility in accurately reflecting the ownership and responsibility of the requirement. This feature accommodates changes in project roles and tasks, ensuring that the correct user is recognized for the creation and subsequent management of the requirement.	Requirement	Pelion (v2)

ID	Requirement	Level	Target
C-A-12	On creation of a change request, a project managers should be able to assign the creator of the change request to another user **Reasoning:** This ensures flexibility in accurately reflecting the ownership and responsibility of the requirement. This feature accommodates changes in project roles and tasks, ensuring that the correct user is recognized for the creation and subsequent management of the requirement.	Requirement	Pelion (v2)

2.5 Metadata

ID	Requirement	Level	Target
C-M-01	Requirements should have the ability to have keywords (aka tags) attached such that the keywords can be used for search. **Reasoning:** This enhancess searchability, allowing users to quickly locate relevant requirements based on specific terms, which improves efficiency in navigating and managing complex projects.	Recommended	Ossa (v1)
C-M-02	There must be the ability to upload and attach files (supporting documentation) to a requirement. **Reasoning:** This enhances documentation completeness and clarity by providing additional context and supporting materials. This capability allows users to include relevant diagrams, specifications, or supporting documents directly within the requirement.	Requirement	Pelion (v2)
C-M-03	There should be the ability to upload files as shared resources to an organisation, such they can be shared between projects. **Reasoning:** This allows streamlines access to centralized documentation repositories.	Recommended	Pelion (v2)
C-M-04	There should be the ability to link an organisation’s shared resource file to a requirement. **Reasoning:** This streamlines access to centralized documentation repositories, ensuring that all stakeholders can easily reference relevant materials without redundancy. This capability promotes consistency and version control by leveraging existing resources and updates across projects, reducing the risk of outdated or conflicting information.	Recommended	Pelion (v2)

2.6 Ease of Use

ID	Requirement	Level	Target
C-E-01	The order of project components in a report should be able to be set by a project manager. **Reasoning:** This enables customized and logical presentation of information, tailored to specific needs and preferences. This flexibility supports clearer communication and easier comprehension of the project's structure and priorities for different stakeholders. Additionally, it enhances usability and efficiency by allowing users to highlight the most critical components in a manner that aligns with their workflow and reporting requirements.	Requirement	Ossa (v1)
C-E-02	The order of requirement categories in a report should be able to be set by the project manager.	Requirement	Ossa (v1)

ID	Requirement	Level	Target
	Reasoning: This enables customized and logical presentation of information, tailored to specific needs and preferences. This flexibility supports clearer communication and easier comprehension of the project's structure and priorities for different stakeholders. Additionally, it enhances usability and efficiency by allowing users to highlight the most critical components in a manner that aligns with their workflow and reporting requirements.		
C-E-03	The order of requirement in a report should be able to be set by the project manager, while the project stage is in scoping state. **Reasoning:** This enables customized and logical presentation of information, tailored to specific needs and preferences. This flexibility supports clearer communication and easier comprehension of the project's structure and priorities for different stakeholders. Additionally, it enhances usability and efficiency by allowing users to highlight the most critical components in a manner that aligns with their workflow and reporting requirements. **Clarification:** By limiting this to the scoping stage, this ensures that requirement identifiers don't get duplicated or become out of order.	Requirement	Ossa (v1)
C-E-04	There should be the ability to set a template project as the default template when creating a new project. **Reasoning:** Setting a template project as the default when creating a new project standardizes project setup, ensuring consistency and adherence to best practices across the organization. This capability saves time by streamlining the initial configuration process, reducing the need for repetitive manual setup for each new project. Additionally, it helps maintain uniformity in project structure and documentation, which facilitates easier management, comparison, and tracking of multiple projects.	Recommended	Pelion (v2)
C-E-05	There should be the ability to use an existing project as a template when creating a new project **Reasoning:** This streamlines project setup, saving time and ensuring consistency by reusing established structures and practices. This capability enhances efficiency and reduces the likelihood of errors or omissions, as successful frameworks and methodologies can be replicated easily. Additionally, it promotes standardization across projects, facilitating easier management and comparison of multiple projects within the organization. **Clarification:** This will copy project groups, members of project groups, project configuration, requirement attribute setup and requirements within the project.	Recommended	Pelion (v2)

2.7 Notifications

ID	Requirement	Level	Target
C-N-01	There should be notifications for joining a project, new project stage entering scoping stage (unless brand new project), project stage entering review, project stage approved, project stage completed, change request submitted, stakeholder input requested, change request approval/rejection, updated project requirements, user password changed, user granted (or ungranted) permissions. **Reasoning:** Implementing detailed notifications for various project activities and user actions ensures that all stakeholders are promptly	Requirement	Pelion (v2)

ID	Requirement	Level	Target
	informed about significant events, enhancing transparency and coordination.		
C-N-02	Notifications must be viewable within the UI. **Reasoning:** This ensures that users are promptly informed about important updates, changes, and actions required, thereby enhancing responsiveness and collaboration.	Requirement	Pelion (v2)
C-N-03	There must be the ability to have notifications sent as emails. **Reasoning:** This ensures that users receive timely updates even when they are not actively using the software, enhancing responsiveness and engagement.	Requirement	Pelion (v2)
C-N-04	For each notification type, users should be able to set if they get notifications and how (email and/or via UI). **Reasoning:** This provides flexibility to match individual preferences and work styles, enhancing user satisfaction and productivity.	Requirement	Pelion (v2)
C-N-05	Users must be able to switch email notifications between daily digest, instantanious and no email notifications. **Reasoning:** This provides flexibility to match individual preferences and work styles, enhancing user satisfaction and productivity.	Recommended	Pelion (v2)

2.8 Customisations

ID	Requirement	Level	Target
C-C-01	For each project, the standard fields (attributes) for a requirement or change request can be customised, and new fields may be added. **Reasoning:** Customizing fields for requirements and change requests on a per-project basis allows the system to adapt to the unique needs and workflows of different projects, enhancing flexibility and relevance. This capability ensures that all necessary information is captured accurately and comprehensively, tailored to the specific context and requirements of each project. **Clarification:** Customisations should include (but may not be limited to) number of attributes, name of attributes, type of attributes and setting if they are required or optional.	Requirement	Pelion (v2)
C-C-02	Custom attributes can be of types short text, long text, checkbox or list (enum). **Reasoning:** This provides the necessary flexibility to capture diverse requirements and change request details effectively within different projects. This capability accommodates varying data formats and structures, ensuring that users can tailor attribute types to suit their specific project needs and workflows. By offering a range of attribute types, the system enhances usability and data integrity, empowering users to accurately capture and manage project information according to their requirements.	Requirement	Pelion (v2)
C-C-03	Ability to change terminology per project. **Reasoning:** This flexibility allows organizations to tailor the system to their preferred terminology, reducing confusion and streamlining communication across teams and projects. Additionally, it supports consistency and coherence within project documentation.	Requirement	Pelion (v2)

ID	Requirement	Level	Target
	Examples: Project stages may be the standard term for a consulting delivery project, while products use versions and agile development projects use the term horizons.		

2.9 Project Management

ID	Requirement	Level	Target
C-P-01	Projects can be archieved, such that no longer show in the active projects list, but the project information still remains. **Reasoning:** This preserves the integrity of the project's historical data and maintains transparency in the system's records. Retaining a project ensures that requirements and change requests are accessible for reference and analysis, aiding in continuity and knowledge retention within the engineering team.	Requirement	Pelion (v2)
C-P-02	Project stages should be able to be marked as completed. If they are completed, a time and by whom should be logged **Reasoning:** Allowing requirements to be marked as completed with a logged time and responsible individual provides clear tracking and accountability, ensuring transparency in project progress. This further helps with understanding the state of the overall project (or product) which can help with external business areas such as marketing and sales. **Clarification:** By default, this should not mark requirements within the project stage to have been completed, however there may be a user option to do so.	Recommended	Massif (v3)
C-P-03	Requirements should be able to be marked as completed. If they are completed, a time and by whom should be logged. **Reasoning:** Allowing requirements to be marked as completed with a logged time and responsible individual provides clear tracking and accountability, ensuring transparency in project progress. This further helps with understanding the state of the overall project (or product) which can help with external business areas such as marketing and sales.	Recommended	Massif (v3)

3 Reporting

3.1 General

ID	Requirement	Level	Target
R-G-01	There must be the ability to add custom markdown text to the end of generated reports (such as appendices). **Reasoning:** This ensures flexibility and completeness in documentation by enabling users to include additional context, explanations, or supplementary information. **Clarification:** Section Titles must be included in markdown.	Requirement	Ossa (v1)
R-G-02	There must be the ability to add custom markdown text at the beginning of generated reports (such as introduction and body chapters). **Reasoning:** This ensures flexibility and completeness in documentation by enabling users to include additional context, explanations, or supplementary information. **Clarification:** Section Titles must be included in markdown.	Requirement	Ossa (v1)
R-G-03	There should be the ability to set filters on requirements for custom generated reports. **Reasoning:** This enables users to tailor the report content to specific criteria, improving relevance and clarity for different stakeholders. This capability enhances efficiency by focusing on pertinent requirements, reducing information overload, and facilitating targeted analysis and decision-making. **Example Use:** Printed Versions can specify range of horizons to show (so doesn't need to show already met versions and doesn't go to far in future)	Requirement	Pelion (v2)
R-G-04	There should be the ability to use an organisation’s shared resource file as a report section. **Reasoning:** Integrating an organization's shared resource file as a report section ensures that reports are comprehensive and consistently aligned with standardized documentation and resources. This capability enhances the accuracy and relevance of reports by directly incorporating up-to-date and authoritative information from shared resources.	Requirement	Pelion (v2)
R-G-05	There should be the ability to use an organisation’s shared resource file as a report section. **Reasoning:** This ensures that organizations can tailor report formats to their specific branding, compliance, and presentation standards. This capability enhances the professionalism and consistency of reports, aligning them with organizational guidelines and stakeholder expectations.	Requirement	Massif (v3)

3.2 Formats

ID	Requirement	Level	Target
R-F-01	Generated Reports must have the ability to output as PDF **Reasoning:** This ensures compatibility and accessibility across various devices and platforms, as PDFs are widely accepted and easily shareable. This capability enhances document integrity and	Requirement	Ossa (v1)

ID	Requirement	Level	Target
	professionalism by preserving the report's formatting and content exactly as intended.		
R-F-02	Generated tables of requirements should have the ability to output as CSV or Spreadsheet. **Reasoning:** This enhances data portability and flexibility, enabling users to manipulate, analyze, and share data easily using various tools.	Recommended	Ossa (v1)

4 User Interface

4.1 Design Principals

ID	Requirement	Level	Target
U-P-01	The UI should have fast response times for all user interactions. When there is a task that is inherently slow, a loading indicator or loading spinner should be shown. **Reasoning:** This enhances the overall user experience, increasing productivity and satisfaction by minimizing delays and frustration. Displaying a loading indicator or spinner for inherently slow tasks provides users with clear feedback that the system is processing their request, maintaining transparency and managing expectations.	Requirement	Ossa (v1)
U-P-02	The UI must be fully responsive and adapt seamlessly to different screen sizes and orientations, including but not limited to mobile devices, tablets, and desktop monitors. **Reasoning:** This ensures accessibility and usability across various devices, allowing users to interact with the system effectively whether they are on mobile devices, tablets, or desktop monitors. This capability enhances user experience by providing a consistent and intuitive interface regardless of the device used, supporting flexibility and convenience for users working in diverse environments.	Requirement	Ossa (v1)
U-P-03	The UI must be designed to optimize workflows to minimize the number of steps required to complete tasks. **Reasoning:** This improves user satisfaction and reduces the likelihood of errors, promoting smoother operation and better overall user experience. Additionally, a streamlined UI fosters user adoption and encourages continued engagement with the software, benefiting both individual users and the project as a whole.	Requirement	Ossa (v1)
U-P-04	The UI must incorporate clean and easy-to-understand icons to enhance user experience and facilitate intuitive navigation. **Reasoning:** This simplifies navigation, reducing the cognitive load on users and enhancing overall usability. This approach fosters a more enjoyable and productive user experience, promoting engagement and adoption of the engineering requirements management system.	Requirement	Ossa (v1)
U-P-05	The project overview UI page, should have metrics of the project shown. **Reasoning:** This provides stakeholders with a comprehensive and real-time snapshot of the project's status, facilitating informed decision-making and effective project management. enhances transparency and allows team members to quickly assess progress and identify areas needing attention. This holistic view promotes accountability and efficiency by ensuring all relevant project data is easily accessible and actionable from a single, centralized interface. **Clarification:** This should include: Number of requirements in project, Percentage of requirements Completed, number of files in project (if files are implemented), and Number of Change Requests in each state (proposed, approved, rejected).	Requirement	Ossa (v1)
U-P-06	The UI should perform lazy loading of large data sets. **Reasoning:** By only loading necessary data as needed, the system remains responsive and efficient, improving usability and scalability.	Recommended	Ossa (v1)

4.2 User Preference

ID	Requirement	Level	Target
U-U-01	The UI must provide both a dark theme and a light theme **Reasoning:** This feature enhances accessibility and user comfort, accommodating individual preferences and reducing eye strain under different lighting conditions.	Requirement	Ossa (v1)
U-U-02	The UI must be designed to allow ease of language translations being used in the interface. The dynamic data is not required to have the ability to allow translations. **Reasoning:** This ensures accessibility to a diverse user base, enhancing inclusivity and usability across different regions and languages. Separating dynamic data from translation requirements minimizes complexity, allowing for smoother development and maintenance processes without compromising user experience.	Requirement	Ossa (v1)
U-U-03	The UI must allow users to set the first page seen after login to be a specific project, the project overview page or automatic (project if there is only one project, else overview). **Reasoning:** This aligns with personalized workflow preferences, enhancing user satisfaction and efficiency. This feature promotes a user-centric approach, catering to individual needs while optimizing the user experience within the engineering requirements management system.	Requirement	Ossa (v1)
U-U-03	Users may have the ability to set favourite projects. Favourite projects should be displayed at the top of list of projects that is displayed to the user. **Reasoning:** This enhances productivity by providing quick access to frequently used projects, streamlining workflow and reducing navigation time.	Requirement	Pelion (v2)

4.3 Ease of Use

ID	Requirement	Level	Target
U-E-01	Search should be able to filter by specific attributes. **Reasoning:** This allows users to quickly and precisely locate relevant requirements, improving efficiency and productivity. By providing advanced filtering options, the system enhances user experience and effectiveness in managing and navigating large volumes of requirements. **Clarification:** Search should default to name and unique id only.	Recommended	Ossa (v1)
U-E-02	There may be the ability to utilise AI to help with writing requirement attributes such as reasoning. **Reasoning:** This helps save time and reducing the cognitive load on users. AI-driven suggestions can help standardize the quality and format of requirement entries, ensuring consistency and clarity across the system. This capability improves productivity and supports users in generating more thorough and well-reasoned requirements, contributing to the overall effectiveness.	Recommended	Murchison (v4)
U-E-03	There must be a project list view, which shows active projects across all organisations for which the user has access too. This must show the projects name, the project summary/description, current project	Requirement	Ossa (v1)

ID	Requirement	Level	Target
	stage, last modification datetime and the users permission roles within each project. **Reasoning:** This functionality enables users to quickly locate active projects and provide a quick overview of it's status.		
U-E-04	In the project list view, there should the ability to list archieved projects across all organisations for which the user has access too **Reasoning:** This allows users to access historical project data, which can be crucial for reference, audits, and compliance purposes. It ensures that important information from past projects is not lost and can be reviewed when needed.	Requirement	Ossa (v1)
U-E-05	The projects shown in the project list view should have the ability to be filtered. **Reasoning:** Filtering options enable users to sort projects based on various criteria, such as status, date, or assigned roles, making it easier to manage large numbers of projects.	Requirement	Pelion (v2)

4.4 Customisation

ID	Requirement	Level	Target
U-C-01	UI Colour scheeme must use CSS variables. **Reasoning:** This will make theme modification easier to match the hosting provider.	Requirement	Ossa (v1)
U-C-02	There should be the ability to add an organisation’s logo to the UI. **Reasoning:** This enables organizations to customize the interface to reflect their brand identity, reinforcing their presence and values within the software. This feature enhances the professional appearance of the system and ensures consistency with the organization's branding across all platforms.	Requirement	Pelion (v2)

5 Infrastructure

5.1 API Interface

ID	Requirement	Level	Target
I-A-01	The UI will be loosely coupled to the backend. **Reasoning:** This enhances system flexibility, facilitating easier updates and modifications to either component independently. This separation of concerns allows for more efficient development and maintenance workflows, enabling developers to focus on improving specific aspects of the software without disrupting the entire system. Loose coupling also promotes scalability and interoperability, supporting future integrations and extensions.	Requirement	Ossa (v1)
I-A-02	The API will be RESTful **Reasoning:** RESTful APIs are preferred for their simplicity, scalability, and flexibility. They utilize standard HTTP methods for CRUD operations, making them easy to understand and use. Their statelessness enables seamless scalability, and their uniform interface promotes interoperability across different technologies. Additionally, RESTful APIs leverage caching mechanisms and promote a separation of concerns between the client and server, resulting in efficient communication and maintainable architectures.	Requirement	Ossa (v1)
I-A-03	The API will be OpenAPI Complient. **Reasoning:** OpenAPI provides a clear, machine-readable description of the API, enabling automatic generation of documentation, client libraries, and server stubs. This not only streamlines development but also enhances communication between teams and stakeholders by providing a common reference point. Moreover, adherence to the OpenAPI specification promotes consistency and interoperability, ultimately reducing errors and accelerating the development process.	Requirement	Ossa (v1)

5.2 Maintenance

ID	Requirement	Level	Target
I-M-01	Product services (frontend and backend services) must be deployable by docker containers. **Reasoning:** This is the industry standard. **Success Criteria:** The project can be pushed to a container registery and easily deployed via docker compose.	Requirement	Ossa (v1)
I-M-02	The database must be easily initilised on first install. **Reasoning:** This Ensures easy initialization of the database on first installation simplifies the onboarding process for server managers, reducing setup time and potential barriers to adoption. This requirement streamlines deployment and configuration.	Requirement	Ossa (v1)
I-M-03	The database must have a way of determining schema version. **Reasoning:** This helps facilitate database maintenance and upgrades, ensuring compatibility with evolving software versions.	Requirement	Ossa (v1)

ID	Requirement	Level	Target
I-M-04	There must be a simple method to backup all data from the deployment, including the database and any metadata (such as files). **Reasoning:** This ensures disaster recovery preparedness for the system. By including database and metadata backups, users can safeguard their entire dataset, reducing the risk of data loss and facilitating seamless restoration in case of failures or cyber attacks.	Requirement	Ossa (v1)
I-M-05	Users with server admin permission role, can create organisations, create users in organisations, and receive deployment notifications. This permission role does not give access to data within organisations. **Reasoning:** This ensures effective system-wide management and oversight, especially in a multi-tenant environment. This role allows for centralized control and administration across all tenants, facilitating efficient setup and management without requiring the server admin to be part of individual organizations. By restricting access to organizational data, this role maintains the privacy and security of each tenant's data, ensuring that sensitive information remains protected while still providing necessary administrative capabilities. **Clarification:** This permission needs to be able to create users in organisations, such that they can create the initial organisation user.	Requirement	Ossa (v1)
I-M-06	There should be a user that is setup in the deployment config, that has the server admin permission role. This user can assign any user on the system, the server admin permission role. This user can be disabled in the deployment config. **Reasoning:** This promotes flexibility and scalability in user management. Providing the option to disable this user offers an additional layer of security and control, aligning with best practices for access control.	Requirement	Ossa (v1)
I-M-07	Any data that is being pushed or pulled from the database must be sanitised. **Reasoning:** Sanitizing data ensures data integrity and protects against security vulnerabilities such as SQL injection attacks, safeguarding the system from malicious exploitation. By enforcing data sanitization, the system mitigates risks associated with unauthorized access or manipulation of sensitive information, enhancing overall system security.	Requirement	Ossa (v1)
I-M-08	On first install, if enabled in the deployment config, the server admin user as defined in the deployment config, should also have the organisational administrator role of an automatically generated organisation. **Reasoning:** This setup facilitates efficient initial configuration and management, allowing the admin to quickly establish organizational settings and user permissions. Enabling this feature in the deployment config streamlines the onboarding process, enhancing usability and ensuring a smooth start for new deployments. **Clarification:** This functionality can specifically be disabled in the deployment configuration.	Requirement	Ossa (v1)
I-M-09	There must be the ability to set an email (or email group) that will be messaged upon notifications related to the deployment. **Reasoning:** This ensures timely communication of critical system events to designated personnel or groups. This capability facilitates swift response to potential issues.	Recommended	Pelion (v2)

ID	Requirement	Level	Target
	Example: Example notifications are disk full or database issue notifications.		
I-M-10	File storage should be implemented such that there can be different backends (file system, minIO etc). **Reasoning:** The system accommodates different scalability needs and data management strategies. This approach promotes interoperability and future-proofing, empowering users to tailor their file storage solutions to their specific requirements.	Requirement	Pelion (v2)
I-M-11	When using a file system for file storage, a utilility should be provided to monitor disk usage, and if a usage threshold is met, send notifications to the deployment management user or configured deployment email address. **Reasoning:** This ensures proactive management of storage resources, preventing potential system downtime or data loss. This utility enhances system reliability and performance by allowing administrators to promptly address storage issues before they escalate.	Recommended	Pelion (v2)

6 Non-Functional

6.1 Development

ID	Requirement	Level	Target
N-E-01	Architecture diagrams should be developed for software functional flow and how modules fit together. **Reasoning:** This aids developers in understanding the software's functional flow and module interactions. These diagrams serve as invaluable documentation for both current and future developers, ensuring maintainability and scalability of the requirements management system.	Recommended	Ossa (v1)
N-E-02	Tools used to develop the system should be open source. **Reasoning:** Utilizing open-source tools promotes interoperability and reduces dependency on proprietary software, ensuring long-term sustainability and flexibility in the development process.	Requirement	Ossa (v1)
N-E-03	The backend service, must be written in Python. **Reasoning:** Selecting Python for the backend ensures streamlined development and maintenance processes due to its simplicity and extensive libraries, accelerating the project timeline. Leveraging a familiar language like Python reduces the learning curve for developers, promoting efficient implementation and minimizing errors or setbacks during development.	Requirement	Ossa (v1)
N-E-04	All code written in Python must follow the Google Python Style Guide. **Reasoning:** This ensures consistency and readability across the codebase, making it easier for multiple contributors to understand and collaborate on the project. This uniformity enhances maintainability, reducing the likelihood of errors and simplifying code reviews and debugging processes. By following a well-established style guide, the project promotes best practices and fosters a high standard of code.	Requirement	Ossa (v1)
N-E-05	All code written in Python must use docstrings to document all packages, modules, classes and functions **Reasoning:** This practice enhances code readability and maintainability, making it easier for new contributors to onboard and for existing developers to collaborate and debug.	Requirement	Ossa (v1)
N-E-06	The frontend UI, must be written in React. **Reasoning:** Choosing React for the frontend UI offers a robust and efficient development framework, providing a rich ecosystem of components and tools to streamline interface design and implementation. Leveraging React's component-based architecture enhances code modularity and reusability, facilitating rapid development and iteration of UI features.	Requirement	Ossa (v1)
N-E-07	All Javascript based code (eg React), must follow the Google Javascript Style Guide. **Reasoning:** This ensures consistency and readability across the codebase, making it easier for multiple contributors to understand and collaborate on the project. This uniformity enhances maintainability, reducing the likelihood of errors and simplifying code reviews and	Requirement	Ossa (v1)

ID	Requirement	Level	Target
	debugging processes. By following a well-established style guide, the project promotes best practices and fosters a high standard of code.		

6.2 Documentation

ID	Requirement	Level	Target
N-D-01	There must be comprehensive documentation on how to deploy and configure a new instance of the project. **Reasoning:** This ensures that users and administrators can easily set up and customize new instances of the project, reducing the entry barrier for adoption.	Requirement	Pelion (v2)
N-D-02	There must be comprehensive documentation on how to use the user interface **Reasoning:** Documentation empowers users to effectively navigate and utilize the system's features, maximizing their productivity and satisfaction. Clear instructions and examples reduce the learning curve, enabling users to quickly become proficient and leverage the full potential of the software.	Requirement	Pelion (v2)

7 Enterprise Features

7.1 User Management

ID	Requirement	Level	Target
E-U-01	There must be the ability to alter an organisation to predominatly use a Single-Sign-On (SSO) user authentication system. The SSO may provide the authorisations avaliable to that user. **Reasoning:** This simplifies user management and enhances security by centralizing authentication processes. SSO integration reduces the need for multiple passwords, streamlining the user experience and improving compliance with organizational security policies.	Requirement	Massif (v3)
E-U-02	The provisioning service, should provide a SCMI interface. **Reasoning:** This ensures interoperability with other systems and tools that support SCMI, enhancing automation and integration capabilities. By using SCMI, the system promotes consistency, reduces complexity, and facilitates efficient project creation and administrative tasks by allowing management to be automated.	Requirement	Massif (v3)

7.2 Provisioning

ID	Requirement	Level	Target
E-P-01	Allow multiple organisations (projects belong to organisations) **Reasoning:** This allows a single deployment to have multiple tenants. This feature can also be used to segregate teams if required.	Requirement	Massif (v3)
E-P-02	There should be a provisioning service on a seperate TCP/IP port that allows all user, group, permissions work to be conducted via an API. Projects should be able to be created using this interface. **Reasoning:** To provide an API that can be connected to other enterprise services for automated creation of projects. Different port allwos a firewall or routing rules to easily secure this service to only be accessible by autherised services.	Requirement	Massif (v3)
E-P-03	Each organisation should be able to have their own login page (if there is only one organisation, it is the main login page) **Reasoning:** This ensures a tailored and branded user experience, enhancing user satisfaction and engagement. This customization supports the unique identity and requirements of different organizations, such as different SSO redirections.	Recommended	Massif (v3)

7.3 Compatibility

ID	Requirement	Level	Target
E-C-01	Data Import/Export to IBM Doors **Reasoning:** This ensures compatibility with a widely-used requirements management tool, facilitating seamless integration and data migration for users. This capability allows organizations to	Recommended	Murchison (v4)

ID	Requirement	Level	Target
	leverage existing data and workflows, and ease of transistioning between the two requirements systems.		

8 Appendix A - Project Stages

Project stages are the “horizons” (agile term), that features will be divided into.

Major version names, come from Tasmanian Ables.

Version Name	Description
Ossa (v1)	A minimum viable product for the project.
Pelion (v2)	Focus on Customisation features.
Massif (v3)	Focus on enterprise features.
Murchison (v4)	Focus on Nice to haves.

At the completion of Ossa, a requirements system will be functional and easy to deploy. The minimum viable product version is named after Mt Ossa, as to achieve it, will be the equivilent of climbing a very tall mountain and Mt Ossa is Tasmania’s highest mountain.

9 Appendix B - Terminology

9.1 Standard Fields

- **Requirement Name:** The requirement name serves as a clear and concise identifier for each specific requirement. It allows stakeholders to easily reference and discuss individual requirements without confusion. A well-defined name facilitates communication and understanding throughout the development process.
- **Reasoning:** The reasoning behind a requirement provides context and justification for its inclusion. It helps stakeholders understand why a particular requirement is necessary for the product. This understanding is crucial for making informed decisions, prioritizing requirements, and resolving conflicts or discrepancies during the development process.
- **Requirement Level:** The requirement level indicates the importance or priority of a requirement in relation to others. It helps stakeholders understand the criticality of each requirement and guides decision-making regarding resource allocation, scheduling, and trade-offs. Clear requirement levels ensure that development efforts are focused on fulfilling the most essential needs of the project.
- **Timeline/Horizon:** The timeline or horizon specifies when the requirement needs to be implemented or achieved. It provides a clear understanding of the project's timeline and helps stakeholders coordinate activities, plan resources, and manage expectations. Additionally, it allows for the prioritization of requirements based on their deadlines or dependencies, ensuring that critical milestones are met on time.
- **Success Criteria:** Success criteria define the conditions that must be met for a requirement to be considered successfully implemented or fulfilled. They provide objective benchmarks for evaluating the quality and completeness of deliverables, guiding testing, validation, and acceptance processes. Clear and measurable success criteria ensure alignment between stakeholder expectations and project outcomes, facilitating transparency and accountability throughout the development lifecycle.
- **History of Requirement:** This section documents the changes, updates, or modifications made to the requirement over time. It includes a timeline of change requests, both approved and disapproved, providing a comprehensive view of the requirement's evolution. Tracking the history of requirements helps stakeholders understand the rationale behind changes, assess the impact on project timelines and resources, and maintain transparency and accountability in requirement management. This section is auto-generated based on change requests and updates.

9.2 Additional/Optional Fields

- **Example(s):** Including an "Example:" detail for a requirement provides clarity and context, helping users understand the specific application and intent of the requirement. This enhances comprehension and ensures consistent interpretation and implementation across different users and teams.
- **Clarification:** This helps eliminate ambiguities by providing additional context and specific details, ensuring that all stakeholders have a clear and consistent understanding of the requirement. This prevents misinterpretation and enhances the accuracy and effectiveness of implementation.
- **Notes:** This offers supplementary information, considerations, or context that may not be immediately apparent, aiding in better understanding and implementation. This helps ensure that all nuances and relevant factors are communicated, supporting more informed decision-making and development.

9.3 Requirement Levels

We have three standard requirement levels:

- **Requirement:** A requirement denotes a feature or functionality that is indispensable for the core functionality of the product. It must be implemented to ensure the product meets its fundamental objectives and serves its intended purpose. Failure to include a requirement may result in the product not functioning as intended or failing to meet essential user needs.
    - Requirements generally use the terms "must" have or "shall" have.
- **Recommended:** Recommended features are enhancements or additions that are deemed beneficial for the product's overall usability, performance, or user experience. While not mandatory for the initial release, recommended features provide valuable insights into the product's future direction and evolution. Integrating recommended features is encouraged to enhance the product's value proposition and competitiveness, but their absence does not compromise the core functionality.
    - Recommendations generally use the term "should" have.
- **Optional:** Optional features represent functionalities that are desirable but not critical for the product's core objectives. These features serve as "nice-to-haves" that can enhance user satisfaction or provide additional value, but their inclusion is not imperative for the product's basic functionality. Optional features may be implemented based on resource availability, project timeline, or specific user preferences, offering flexibility in tailoring the product to varying needs and preferences.
    - Optionals generally use the term "may" have.