Stakeholder Requirement Management

Introduction to Stakeholder Requirement Management

Managing stakeholder requirements ensures that the needs of users are correctly reflected in the system's design

1. Stakeholder Requirements as the Foundation

- Stakeholder requirements form the basis for system design, influencing project goals and objectives.
- They help define what success looks like for both users and stakeholders involved in the project.

2. Importance of Clear Communication and Documentation

- Stakeholder requirements form the basis for system design, influencing project goals and objectives.
- They help define what success looks like for both users and stakeholders involved in the project.

3. Meeting Business and User Expectations

- Properly managed stakeholder requirements lead to a system that meets both business goals and user satisfaction.
- This alignment is crucial for achieving successful project outcomes and ensuring stakeholder buy-in.

Converting Elicitation Results into Stakeholder Requirements

Elicitation results are converted into actionable stakeholder requirements,
which provide clear direction for system development

1. Gathering Raw Data from Elicitation Techniques

- Elicitation techniques like interviews, workshops, surveys, and observation yield valuable but unstructured information.
- This raw data includes insights into stakeholder needs, pain points, and goals.

2. Organizing Data into User Stories and Job Stories

- Stakeholder requirements form the basis for system design, influencing project goals and objectives.
- They help define what success looks like for both users and stakeholders involved in the project.

3. Prioritizing and Refining

- Use techniques like MoSCoW prioritization (Must have, Should have, Could have, Won't have) to determine which stakeholder requirements are most critical.
- Stakeholders and teams can collaboratively refine these stories to ensure they align with business goals and user needs.

What are User Stories and Job Stories?

User stories and job stories serve different but complementary purposes in capturing stakeholder requirements

1. User Stories

- Focus on what the user wants to achieve, such as desired features or tasks.
- Example format: "As a [user], I want [action/feature], so that [benefit]."

2. Job Stories

- Provide richer context by capturing both the motivation and situation: "When [situation], I want to [motivation], so I can [outcome].“
- Job stories explain the "why" behind the user's actions and help developers understand the context.

3. Aligning Solutions with User Needs

- Both approaches ensure that technical solutions are built with user and stakeholder needs in mind.
- This helps teams deliver relevant and usable systems, increasing project success rates.

Key differences between User Stories and Job Stories

Aspect	User Stories	Job Stories
Focus	What the user wants to achieve.	The situation (when) and motivation (why) behind the user’s action.
Structure	"As a [user], I want [action/feature], so that [benefit]."	"When [situation], I want to [motivation], so I can [expected outcome]."
Goal	Describes a specific user need or feature, focusing on functionality.	Explores the user’s context and motivation, focusing on real-world situations and needs.
Context Provided	Limited context, primarily focused on the user’s goal or desired feature.	Provides rich context, including the user's environment and reasoning behind the action.
Use Case	Ideal for capturing functional requirements that describe what the user expects from the system.	Ideal for situations where understanding user behavior, context, or motivation is critical.
Example	"As a student, I want to view my grades online, so I can track my academic progress."	"When the semester ends, I want to view my grades online, so I can understand my academic performance."
Level of Detail	Typically provides less detail about why or when the feature is needed.	Offers more detail by explaining the situation and why the feature is important for the user.
Benefit to Development Teams	Helps teams focus on specific user needs and desired features.	Helps teams understand the broader context and motivation behind user actions, leading to better prioritization and design.
When to Use	Use when a clear goal or feature needs to be captured.	Use when more context or motivation is needed to fully understand the user’s requirement.
Prioritization	Easier to prioritize based on the desired feature or functionality.	Provides insights for prioritization based on the user’s situation, urgency, or motivation.

User Stories	Job Stories
"As a customer, I want to add items to my shopping cart, so I can purchase them later."	"When I'm browsing products, I want to add items to my shopping cart, so I can easily review them before purchasing."
"As a user, I want to receive order confirmation emails, so I know my order was successful."	"When I complete a purchase, I want to receive an order confirmation, so I know it was successful."
"As a customer, I want to track my order, so I know when it will arrive."	"When waiting for a delivery, I want to track my order in real-time, so I can know exactly when it will arrive."
"As a user, I want to reset my password, so I can regain access to my account."	"When I forget my password, I want to reset it easily, so I can quickly regain access to my account."
"As a shopper, I want to filter products by price, so I can find items within my budget."	"When I'm shopping for a specific budget, I want to filter items by price, so I can easily find affordable options."
"As a user, I want to view product reviews, so I can make informed purchase decisions."	"When deciding on a product, I want to view customer reviews, so I can make an informed purchase decision."
"As a customer, I want to save my payment information, so I can checkout faster in the future."	"When I'm frequently shopping, I want to save my payment details, so I can speed up the checkout process."
"As an admin, I want to generate monthly sales reports, so I can track our revenue growth."	"When it's the end of the month, I want to generate a sales report, so I can analyze the revenue growth for that period."
"As a user, I want to receive notifications about discounts, so I can take advantage of them."	"When there are new discounts available, I want to be notified immediately, so I can take advantage of the savings."
"As a user, I want to browse product categories, so I can find specific types of items more easily."	"When I'm shopping for specific items, I want to browse product categories, so I can quickly find what I'm looking for."

Examples of User Stories and Job Stories

Structuring Requirements in User Stories

A well-structured user story captures the user's needs in a simple, actionable format

1. Standard Structure of User Stories

- User stories follow a format: "As a [user], I want [action/feature], so that [benefit].“
- This ensures that each story is focused on user goals and outcomes.

2. Clarity and Simplicity

- Each user story should be written clearly and concisely, avoiding technical jargon.
- User stories should focus on the user's perspective and needs.

3. Breaking Down Complex Requirements

- User stories help decompose complex requirements into manageable, actionable tasks.
- This makes it easier for development teams to prioritize and deliver features in an agile environment.

Good vs Bad Examples of User Stories

Good user stories provide specific details and context, while bad user stories are vague and lack clarity

👍 **Good Example:**

- "As a customer, I want to receive an order confirmation email, so I know my order was placed successfully."

👎 **Bad Example:**

- "The system should send an email."

2. Good User Stories Are Specific and Actionable

- They clearly describe the user’s goal, making it easier for the development team to implement the right features.
- Good stories focus on user needs and how the system will deliver a benefit.

3. Vague Stories Lead to Ambiguity

- Poorly written stories leave too much room for interpretation, leading to potential misunderstandings and incorrect implementations.

Good vs Bad Examples of User Stories 2

Good user stories provide clear motivation and context, while bad user stories lack actionable details

👍 **Good Example:**

- "As a frequent traveler, I want to save my frequent destinations, so that I can quickly select them when booking."

👎 **Bad Example:**

- "The system should save destinations."

2. Good User Stories Provide Full Context

- These stories give developers the necessary detail to implement the feature effectively, focusing on how the user will benefit.
- They also make it easier to prioritize tasks based on user impact.

3. Vague User Stories Leave Gaps

- Bad stories do not provide enough detail or context, making it difficult for development teams to understand what's expected.
- These gaps can lead to features that don't meet the actual user needs or goals.

Good User Story	Bad User Story
"As a customer, I want to receive an order confirmation email, so I know my order was placed successfully."	"The system should send an email."
"As a user, I want to reset my password, so I can regain access to my account securely."	"The system should allow password reset."
"As a frequent traveler, I want to save my frequent destinations, so I can quickly select them when booking."	"The system should save destinations."
"As an admin, I want to generate weekly sales reports, so I can track revenue trends."	"The system should create reports."
"As a project manager, I want to assign tasks to my team, so I can track progress."	"The system should handle task assignment."
"As a shopper, I want to filter products by price, so I can find items within my budget."	"The system should provide a filter."
"As a customer, I want to track my order, so I know when my package will arrive."	"The system should show order status."
"As a user, I want to be notified of new messages, so I can respond quickly."	"The system should send notifications."
"As a job applicant, I want to upload my resume, so that recruiters can review my qualifications."	"The system should upload documents."
"As a user, I want to see my recent activities, so I can easily pick up where I left off."	"The system should show recent activities."

Examples of User Stories and Job Stories

Structuring Requirements in Job Stories

Job stories help teams understand the context behind the requirement and why it is important to the user

1. Job Story Structure

- Job stories add context to the user's situation: "When [situation], I want to [motivation], so I can [outcome].“
- This structure helps teams understand the conditions under which the user interacts with the system.

2. Revealing Motivation and Situational Context

- Job stories focus not only on the user's actions but also why and when the action occurs.
- This added context helps guide development teams in designing appropriate solutions.

3. Useful in Complex Scenarios

- Job stories are especially valuable in scenarios where environmental or situational factors significantly influence user behavior.
- They can lead to more user-centered design decisions.

Good vs Bad Examples of Job Stories

Good job stories capture the user's context and motivation, while bad job stories are vague and lack the 'why' behind the action

👍 **Good Example:**
- "When I'm tracking a package, I want to see real-time updates, so I can know when my delivery will arrive."

👎 **Bad Example:**
- "The system should show package tracking."

2. Good Job Stories Focus on the Situation
- They emphasize the user's motivation and provide insights into when and why the user will interact with the system.
- This extra context helps developers understand the broader picture.

3. Vague Job Stories Lead to Confusion
- Job stories without clear motivation leave developers guessing about the importance and priority of a feature.
- Missing context can result in misaligned implementations that don't fully address user needs..

Good vs Bad Examples of Job Stories 2

Good job stories provide context and motivation, helping teams understand the importance of the requirement

👍 **Good Example:**

- ""When I’m working on a tight deadline, I want the system to notify me of critical tasks, so I can focus on urgent issues."

👎 **Bad Example:**

- "The system should notify about tasks."

2. Good Job Stories Add Depth

- They help teams understand the urgency and specific needs that the user faces in certain situations.
- Job stories provide more realistic scenarios for developers to consider when designing solutions.

3. Vague Job Stories Cause Misalignment

- Without understanding the user’s situation or motivation, teams may design features that don’t fully support the user’s real-world needs.
- Lack of situational context can also affect prioritization.

Good Job Story	Bad Job Story
"When I’m placing an order, I want to receive a confirmation email, so I know my purchase was successful."	"The system should send confirmation emails."
"When I forget my password, I want to be able to reset it, so I can quickly regain access to my account."	"The system should reset passwords."
"When I’m traveling, I want to save my frequently visited destinations, so I can easily book future trips."	"The system should save destinations."
"When I’m analyzing sales data, I want to generate reports weekly, so I can track revenue trends."	"The system should generate reports."
"When I’m managing a project, I want to assign tasks to team members, so I can track progress and ensure completion."	"The system should assign tasks."
"When I’m shopping, I want to filter items by price, so I can find products within my budget."	"The system should provide filters."
"When I’m waiting for a delivery, I want to track my package in real-time, so I know when it will arrive."	"The system should show package tracking."
"When I’m receiving important messages, I want to be notified instantly, so I can respond promptly."	"The system should send notifications."
"When I’m applying for jobs, I want to upload my resume, so employers can easily review my qualifications."	"The system should allow document uploads."
"When I’m revisiting the app, I want to see my recent activities, so I can pick up where I left off without hassle."	"The system should display recent activities."

Examples of User Stories and Job Stories

Organizing User Stories and Job Stories into Chapters

Organizing stories into chapters provides structure and helps teams focus on specific areas of the system

1. Grouping Stories by Features or Goals

- Organizing user stories and job stories into thematic chapters (e.g., account management, notifications) helps create a clear structure.

2. Clarity and Focus for Development Teams

- Grouping stories into chapters ensures teams can focus on one feature or module at a time, preventing overlapping work and confusion.

3. Easier Tracking and Progress Monitoring

- A well-organized structure allows stakeholders and project managers to track progress on specific system modules or features more efficiently.

3. Grouping Example

- Group 1: "Order Management“
- Group 2: "User Profile Management“
- Group 3: "Notification System"

Organizing User Stories and Job Stories into Chapters

Organizing stories into chapters provides structure and helps teams focus on specific areas of the system

1. Grouping Stories by Features or Goals

- Organizing user stories and job stories into thematic chapters (e.g., account management, notifications) helps create a clear structure.

2. Clarity and Focus for Development Teams

- Grouping stories into chapters ensures teams can focus on one feature or module at a time, preventing overlapping work and confusion.

3. Easier Tracking and Progress Monitoring

- A well-organized structure allows stakeholders and project managers to track progress on specific system modules or features more efficiently.

3. Grouping Example

- Group 1: "Order Management“
- Group 2: "User Profile Management“
- Group 3: "Notification System"

Using ChatGPT for Requirement Extraction

AI tools can assist in extracting and organizing requirements from raw stakeholder feedback, helping streamline the elicitation process.

1. AI for Analyzing Transcripts and Documents

- ChatGPT can analyze transcripts or documents to extract key requirements, reducing manual effort.

2. Identifying Themes and Patterns

- AI tools can help identify patterns or common themes across multiple stakeholder inputs, ensuring major requirements are not missed.

3. Speed and Efficiency

- Using AI like ChatGPT for extraction speeds up the elicitation process, allowing teams to focus more on analysis and refinement.

Steps

- Step 1: "Upload stakeholder interview transcript“
- Step 2: "ChatGPT analyzes and extracts key requirements“
- Step 3: "ChatGPT refines and organizes requirements into user stories or job stories"

Process

- Input: "Raw Elicitation Data“
- Process: "ChatGPT Analysis and Refinement“
- Output: "User Stories and Job Stories"

Using AI Tools to Augment Requirements

Using AI tools to augment requirements can help improve clarity, completeness, and consistency across the project

1. Refining Vague Requirements

- AI tools can refine vague requirements by suggesting more specific, actionable alternatives.

2. Identifying Gaps and Inconsistencies

- AI can detect gaps or inconsistencies in requirements, helping teams ensure that the full scope is covered.

3. Generating Variations for Different Scenarios

- AI tools can suggest different variations of requirements to address edge cases or alternative user scenarios.

Example of Requirement Augmentation with AI tools

AI tool can help refine vague requirements into clear, actionable user stories or job stories

1. Initial Requirement:

- "The system should allow users to upload documents."

2. Refined by AI tool

- "As a user, I want to upload documents up to 10 MB in size, so that I can store important files securely in the system.

3. AI Augmentation Enhances Clarity

- AI tools helps teams refine vague or incomplete requirements, transforming them into clear, detailed stories.

Using AI tools for Requirement Validation

AI tools can help validate requirements, ensuring that they are clear, complete, and ready for implementation.

1. Reviewing for Clarity and Consistency ✅

- AI Tools can review stakeholder requirements to ensure they are clear, consistent, and actionable.

2. Ensuring Alignment with Stakeholder Needs 🔄

- AI tools can validate that requirements are aligned with stakeholder expectations and project goals.

3. Suggestions for Refinement 🧠

- AI tools can suggest refinements or improvements, ensuring that the requirements are detailed enough for implementation and validation

Process

- Step 1: "Input stakeholder requirements“
- Step 2: "ChatGPT reviews for clarity and completeness“
- Step 3: "ChatGPT suggests refinements"

Best Practices for Organizing and Managing Requirements

Using best practices for managing requirements helps ensure that stakeholder needs are addressed consistently throughout the project

1. Use of Tools like Jira or Confluence

- Leverage tools like Jira or Confluence to track and organize user stories, job stories, and overall stakeholder requirements.

2. Regular Reviews and Stakeholder Involvement

- Establish frequent review cycles with stakeholders to validate and refine requirements as the project evolves.

3. Version Control and Traceability

- AI tools can suggest different variations of requirements to address edge cases or alternative user scenarios.

Process

- Step 1: "Organize requirements in a tool (e.g., Jira)“
- Step 2: "Review regularly with stakeholders“
- Step 3: "Refine based on feedback"

Using Atlassian Tools for Requirement Management

Jira and Confluence work together to ensure that requirements are tracked, documented, and easily accessible throughout the project lifecycle

1. **Use of Tools like Jira or Confluence**

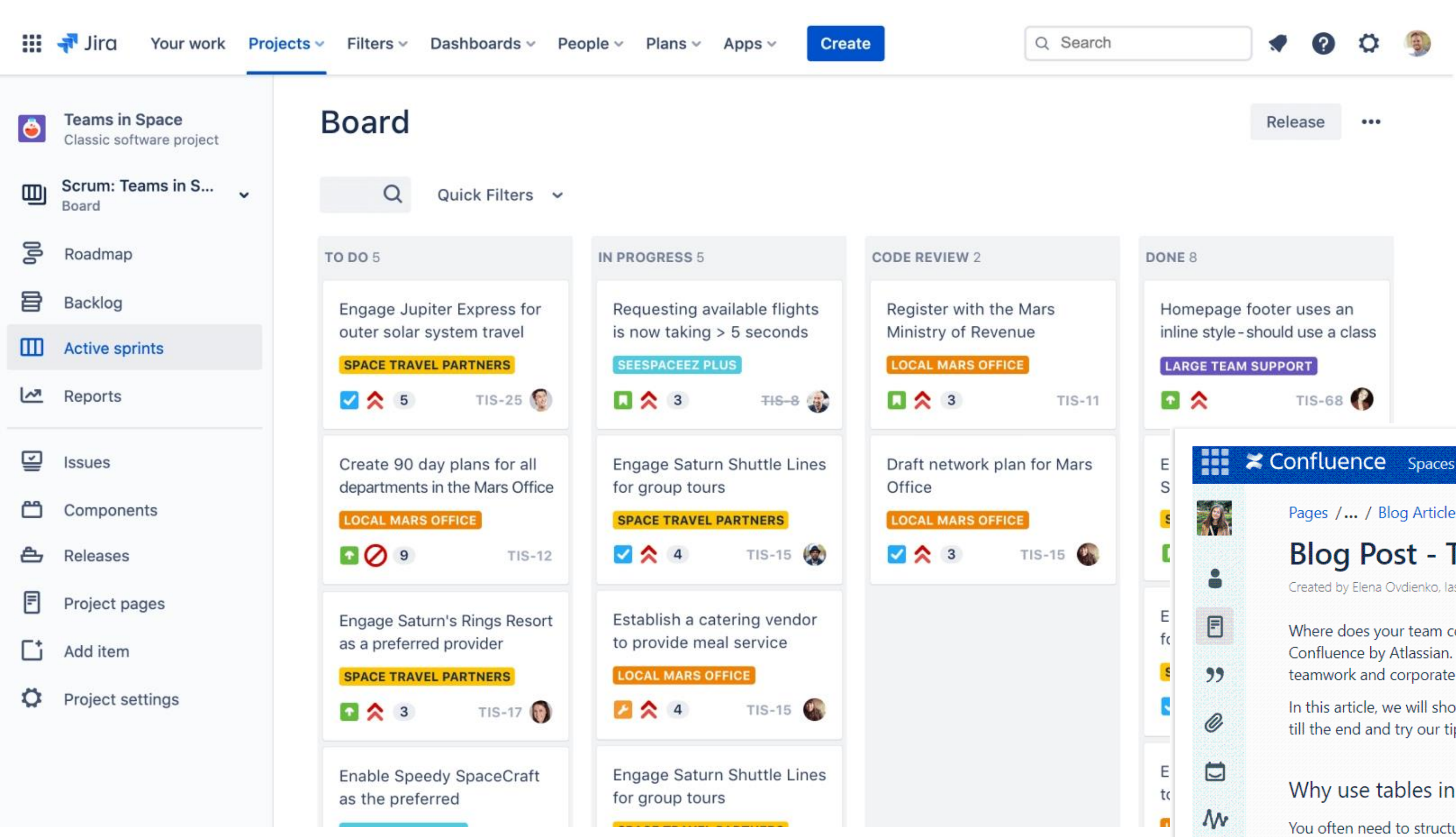
- Jira helps manage and track user stories, job stories, and tasks, ensuring they are organized and aligned with project goals.

2. **Collaborative Documentation in Confluence**

- Confluence provides a collaborative space for documenting requirements, capturing feedback, and discussing changes with stakeholders.

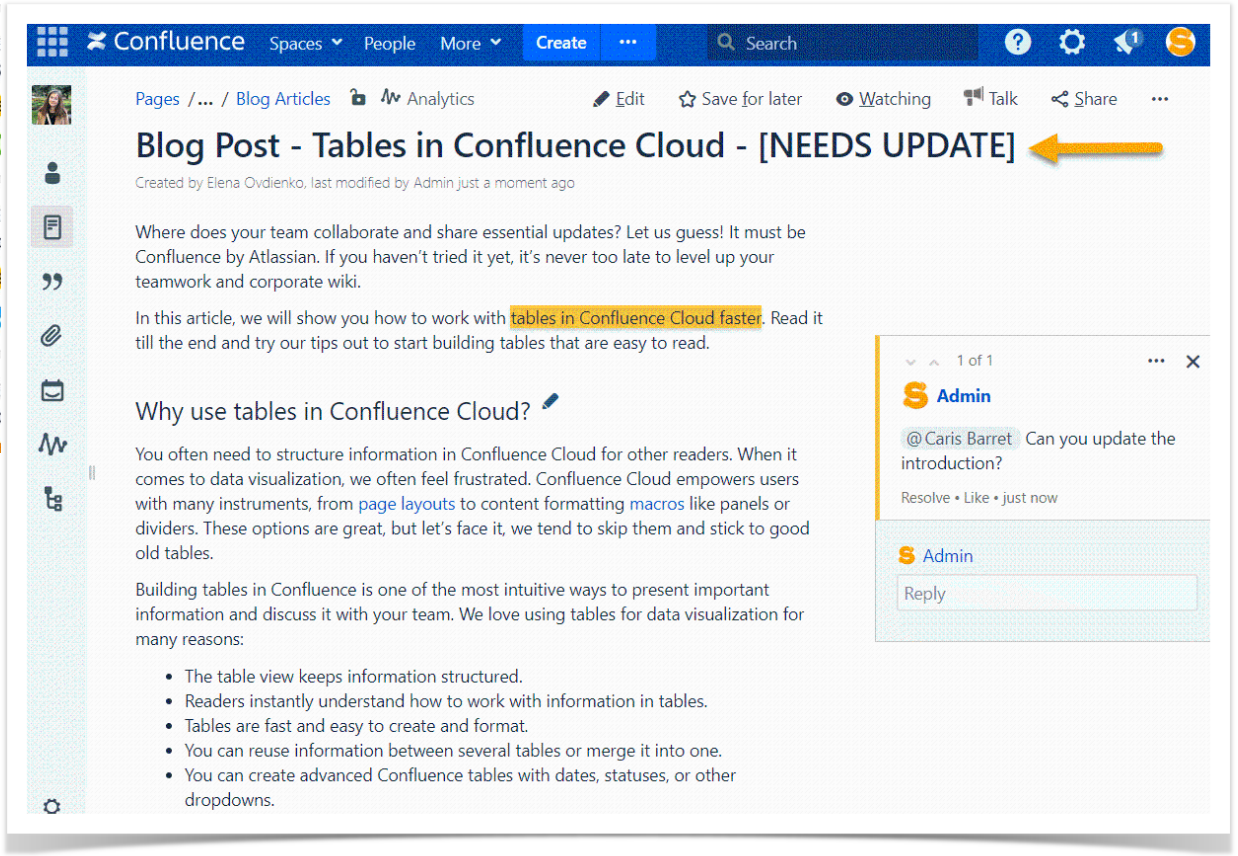
3. **Integrating Jira and Confluence**

- Integrating Jira and Confluence creates a seamless workflow, linking tasks with related documentation, providing full traceability from requirement to delivery.



Board

TO DO 5	IN PROGRESS 5	CODE REVIEW 2	DONE 8
Engage Jupiter Express for outer solar system travel — SPACE TRAVEL PARTNERS — 5 — TIS-25	Requesting available flights is now taking > 5 seconds — SEESPACEEZ PLUS — 3 — ~~TIS-8~~	Register with the Mars Ministry of Revenue — LOCAL MARS OFFICE — 3 — TIS-11	Homepage footer uses an inline style - should use a class — LARGE TEAM SUPPORT — TIS-68
Create 90 day plans for all departments in the Mars Office — LOCAL MARS OFFICE — 9 — TIS-12	Engage Saturn Shuttle Lines for group tours — SPACE TRAVEL PARTNERS — 4 — TIS-15	Draft network plan for Mars Office — LOCAL MARS OFFICE — 3 — TIS-15	
Engage Saturn's Rings Resort as a preferred provider — SPACE TRAVEL PARTNERS — 3 — TIS-17	Establish a catering vendor to provide meal service — LOCAL MARS OFFICE — 4 — TIS-15		
Enable Speedy SpaceCraft as the preferred	Engage Saturn Shuttle Lines for group tours		



Blog Post - Tables in Confluence Cloud - [NEEDS UPDATE]

Created by Elena Ovdienko, last modified by Admin just a moment ago

Where does your team collaborate and share essential updates? Let us guess! It must be Confluence by Atlassian. If you haven't tried it yet, it's never too late to level up your teamwork and corporate wiki.

In this article, we will show you how to work with tables in Confluence Cloud faster. Read it till the end and try our tips out to start building tables that are easy to read.

Why use tables in Confluence Cloud?

You often need to structure information in Confluence Cloud for other readers. When it comes to data visualization, we often feel frustrated. Confluence Cloud empowers users with many instruments, from page layouts to content formatting macros like panels or dividers. These options are great, but let's face it, we tend to skip them and stick to good old tables.

Building tables in Confluence is one of the most intuitive ways to present important information and discuss it with your team. We love using tables for data visualization for many reasons:

- The table view keeps information structured.
- Readers instantly understand how to work with information in tables.
- Tables are fast and easy to create and format.
- You can reuse information between several tables or merge it into one.
- You can create advanced Confluence tables with dates, statuses, or other dropdowns.

1 of 1

Admin

@Caris Barret Can you update the introduction?

Resolve • Like • just now

Admin

Reply

Jira and Confluence Tools

Summary and Q&A

Stakeholder requirement management is key to project success, and using the right tools and techniques ensures clarity, collaboration, and alignment with stakeholder needs.

1. Organizing Requirements into User Stories and Job Stories

- Structuring stakeholder requirements into clear user and job stories ensures development teams can align their work with user needs.

2. Good vs Bad Examples to Ensure Clarity

- Using good and bad examples helps highlight the importance of clarity in requirement writing, improving communication and reducing errors.

2. Using AI for Requirement Extraction and Augmentation

- AI tools assist in extracting and refining requirements, streamlining the process and improving accuracy.

2. Managing Requirements with Atlassian Tools (Jira & Confluence)

- Atlassian tools like Jira and Confluence facilitate effective requirement tracking, documentation, and collaboration across teams.

2. Best Practices for Organizing and Managing Stakeholder Requirements

- Following best practices, including regular reviews, traceability, and documentation, ensures successful requirement management throughout the project lifecycle.