**Q. JIRA History?**  
A. JIRA is developed by Atlassian company in 2002,  
 Atlassian company also developed git, bitbucket and confluence.  
 Today, JIRA is one of the most widely used project management tools in the world.

**1. Origins and Early Development (2002)**

* **Launch**: JIRA was first released in 2002 by Atlassian, an Australian software company founded by Mike Cannon-Brookes and Scott Farquhar. The name "JIRA" was derived from "Gojira," the Japanese word for "Godzilla," which was a reference to the tool being a bug-tracking system designed to be more powerful than the popular tool at the time, Bugzilla.
* **Initial Purpose**: Initially, JIRA was developed as a simple issue-tracking tool for software development teams, focusing on tracking bugs and managing tasks within development projects.

**2. Growth and Expansion (2004-2010)**

* **Agile Adoption**: As agile methodologies, particularly Scrum and Kanban, gained popularity in the mid-2000s, JIRA adapted to these trends by incorporating features that supported agile practices. This allowed teams to use JIRA for more than just bug tracking, expanding into full project management.
* **JIRA Agile**: In 2010, Atlassian introduced JIRA Agile (formerly known as Green Hopper), an add-on that allowed teams to manage agile projects using Scrum and Kanban boards. This feature significantly increased JIRA's popularity among agile teams.

**3. Becoming an Industry Standard (2010-2015)**

* **Enterprise Adoption**: Throughout the early 2010s, JIRA became the go-to tool for software development teams worldwide. Its flexibility, scalability, and extensive customization options made it suitable for small startups and large enterprises alike.
* **Marketplace and Integrations**: Atlassian launched the Atlassian Marketplace, which allowed third-party developers to create plugins and extensions for JIRA. This expanded JIRA's capabilities, making it a highly customizable and integrated tool within software ecosystems.

**4. JIRA Software, JIRA Core, and JIRA Service Desk (2015)**

* **Rebranding**: In 2015, Atlassian rebranded JIRA into three distinct products to cater to different use cases:
  + **JIRA Software**: Focused on software development teams with features tailored for agile project management.
  + **JIRA Core**: Designed for business teams to manage projects, tasks, and processes without the technical focus of JIRA Software.
  + **JIRA Service Desk**: A service management tool designed for IT teams to manage and resolve customer issues.
* **Cloud** Instance **and Server** Instance: JIRA also offered deployment options, including cloud-based and server-based, allowing companies to choose between a hosted solution or managing the software on their own infrastructure.

**5. Modern Developments (2016-Present)**

* **Continued Evolution**: JIRA continues to evolve with regular updates, introducing new features, improved user interfaces, and more robust integrations with other tools like Confluence, Bitbucket, and Trello.
* **Atlassian Cloud**: The focus on cloud offerings has increased, with Atlassian investing heavily in its cloud platform to offer more scalable and globally accessible solutions.
* **JIRA Align**: In 2019, Atlassian acquired Agile Craft, a platform for scaling agile across enterprises, and rebranded it as JIRA Align. This product helps organizations implement and manage agile practices at scale.

**6. Current Status**

* **Widespread Adoption**: Today, JIRA is one of the most widely used project management tools in the world, particularly in the software development industry. It is known for its robust features, flexibility, and extensive ecosystem of integrations and plugins.
* **Focus on Cloud**: Atlassian has been shifting its focus toward cloud solutions, encouraging customers to migrate from server-based versions to Atlassian Cloud.

**Q. What Is JIRA?**  
A. JIRA is a popular project management and issue-tracking tool developed by Atlassian in 2002.

**Q. Need of JIRA?**  
A. One-stop Solution for project management.  
 Multiple projects can handle,  
 Team can communicate over the project.  
 Track Project Work Status,  
 Track Bug and Issues,  
 Team Work and Time Tracking, ETC…

**Q. Who Should Learn JIRA?**  
A. Software Developer, QA, Business Developers, Team Manager and Management.

**Q. What are the different Products of JIRA?**  
A. JIRA, developed by Atlassian, has evolved into a suite of products tailored to different aspects of project management and service delivery. Here are the main products under the JIRA brand:



**1. JIRA Software**

Purpose: JIRA Software is specifically designed for software development teams. It provides tools for planning, tracking, and releasing software.

**2. JIRA Service Management (formerly JIRA Service Desk)**

Purpose: JIRA Service Management is designed for IT and customer service teams to manage and resolve service requests, incidents, and changes.

**3. # JIRA Core (Not in Use)**

Purpose: JIRA Core is a simplified version of JIRA aimed at business teams for managing tasks, projects, and workflows without the complexity of software development features.

**4. JIRA Work Management**

Purpose: JIRA Work Management is a newer addition, designed to be a general-purpose project management tool for business teams.

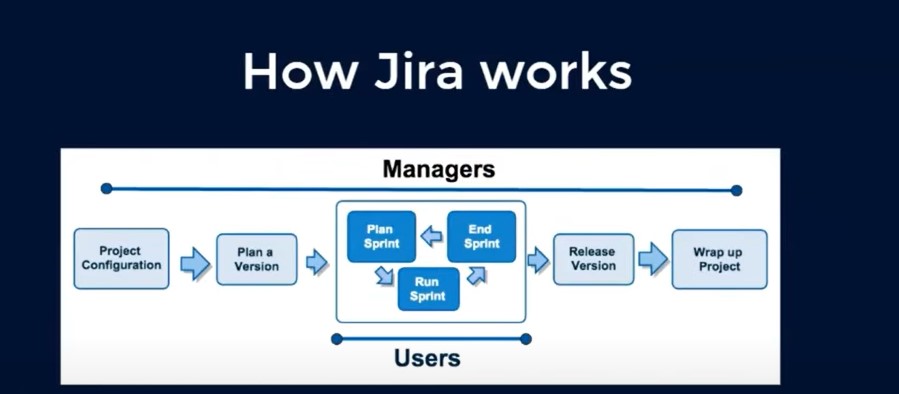
**Q. What we will learn in JIRA As A QA or Software Developer?**  
A.   
How to Install JIRA  
How to make A project in Jira,  
Practical use of Jira,  
How to make (Epic, Story, Bug, Task)  
Project Setting  
Project Dashboard  
Sprint  
Agile/ Scrum  
Search Items (Bug, story etc.)

**Q. What is Sprint Grooming?**  
A. sprint grooming is a continuous process that ensures the backlog is up-to-date, well-prioritized, and clear, setting the stage for effective sprint planning and successful project execution.

**Q. How to install JIRA?**  
A. **Steps to Install JIRA Cloud:**

1. **Sign Up for an Atlassian Account:**
   * Visit the [Atlassian website](https://www.atlassian.com/).
   * Click on "Try Cloud" under the JIRA Software section or go directly to JIRA Cloud Sign Up.
   * Sign up with your email or use an existing Atlassian account.
2. **Choose a JIRA Product:**
   * After signing up, you’ll be prompted to choose which JIRA product you want to try (e.g., JIRA Software, JIRA Service Management, or JIRA Work Management).
   * Select the product that suits your needs.
3. **Set Up Your Site:**
   * Choose a site name (this will be the subdomain for your JIRA instance, e.g., yourcompany.atlassian.net).
   * Atlassian will set up your JIRA Cloud instance automatically.
4. **Configure Your Project:**
   * Once your site is ready, you can create your first project by choosing a template (e.g., Scrum, Kanban, Bug tracking).
   * Configure your project settings such as permissions, workflows, and issue types.
5. **Invite Team Members:**
   * You can invite team members to your JIRA Cloud instance by entering their email addresses.
   * Assign roles and permissions as necessary.
6. **Start Using JIRA:**
   * You can now start using JIRA for project management, issue tracking, and other tasks.
   * Explore other Atlassian tools like Confluence, Bitbucket, and Trello to integrate with JIRA.

The image you provided outlines a high-level workflow of how JIRA is used in a typical project management process, specifically within an agile framework. Below is a detailed explanation of each component in the image:



**1. Project Configuration:**

Who: Managed by “Managers”.

What: This step involves setting up the project in JIRA. This includes configuring the project parameters, setting up the necessary fields, permissions, and workflows, and defining issue types (e.g., stories, tasks, bugs).

Why: Proper configuration ensures that the project is aligned with the team's workflow and organizational requirements.

**2. Plan a Version**

Who: Managed by \*\*Managers\*\*.

What: This step involves planning the project in terms of versions or releases. A version represents a significant milestone or deliverable within the project.

Why: Planning versions helps in organizing the backlog and sets clear goals for what needs to be achieved in each release.

**3. Plan Sprint**

Who: Involves both \*\*Managers\*\* and \*\*Users\*\*.

What: Sprint planning is the process where the team selects the backlog items (user stories, tasks) to be worked on during the upcoming sprint. This step is typically done during the sprint planning meeting.

Why: This ensures that the team has a clear, manageable workload for the sprint and that they are aligned on the sprint goals.

**4. Run Sprint**

Who: Involves \*\*Users\*\*.

What: During the sprint, the team works on the tasks and user stories selected during sprint planning. They track their progress using JIRA’s tools, such as boards, to manage and visualize their work.

Why: Running the sprint is where the actual development work happens, with the goal of completing the tasks planned for the sprint.

**5. End Sprint**

Who: Involves both \*\*Managers\*\* and \*\*Users\*\*.

What: At the end of the sprint, the team reviews what was completed. Unfinished items may be moved to the next sprint or reprioritized. This is also when sprint review and retrospective meetings are held.

Why: Ending the sprint allows the team to assess their progress, identify areas for improvement, and plan the next steps.

**6. Release Version**

Who: Managed by \*\*Managers\*\*.

What: Once a version (or milestone) is completed, it is released. This involves deploying the completed work to production or delivering it to the client.

Why: Releasing a version marks the delivery of a set of features or improvements, providing value to users or customers.

**7. Wrap up Project**

Who: Managed by \*\*Managers\*\*.

What: The final step is wrapping up the project, which includes closing out all tasks, ensuring all documentation is complete, and reflecting on the project’s outcomes.

Why: Wrapping up the project ensures that everything is properly documented, lessons learned are captured, and the project is formally closed.

**Roles:**

Managers: Typically, responsible for overseeing the entire project, including setup, planning, and release management. They ensure that the project aligns with business goals and manages resources effectively.

Users: Generally, refers to the development team or those directly involved in executing the tasks within each sprint. They focus on delivering the work planned during the sprint.

**Q. What is (Epic, Story, Bug, Task)  
1. Epic**

* **Definition**: An Epic is a large body of work that can be broken down into smaller tasks (stories, bugs, or tasks). It typically represents a significant feature or goal that requires multiple sprints to complete.
* **Use Case**: Epics are used to group related stories and tasks that contribute to a broader objective. For example, an Epic could be "User Authentication System," which would include stories like "Login Page," "Forgot Password," and "Social Media Integration."
* **Duration**: An Epic often spans multiple sprints or even releases.
* **Purpose**: It helps in organizing and tracking progress on larger goals within a project.

**2. Story**

* **Definition**: A Story (often called a User Story) is a small, concise requirement or functionality written from the perspective of an end-user. It describes a feature or function that the user needs.
* **Use Case**: Stories are used to capture specific requirements for a feature. For example, a story might be "As a user, I want to be able to reset my password so that I can access my account if I forget it."
* **Duration**: Typically completed within a single sprint.
* **Purpose**: Stories help the team to focus on delivering small, valuable increments of functionality to the end-user.

**3. Bug**

* **Definition**: A Bug is an issue that represents a defect or problem in the software that needs to be fixed. Bugs usually arise when something does not work as expected or when a feature behaves in an unintended way.
* **Use Case**: Bugs are logged whenever a problem is identified during testing, by users, or during development. For example, "Login button is not working on the mobile version of the site" would be a bug.
* **Duration**: Fixing a bug can vary in time depending on the complexity and impact of the issue.
* **Purpose**: Bugs are crucial for maintaining software quality and ensuring that the application behaves as intended.

**4. Task**

* **Definition**: A Task is a general-purpose issue type that represents any kind of work that needs to be done but is not necessarily tied to a user story. Tasks can be technical, non-technical, or administrative.
* **Use Case**: Tasks are used for things like setting up a development environment, performing code reviews, writing documentation, or conducting a training session. For example, "Set up the new server environment" could be a task.
* **Duration**: Tasks can usually be completed within a sprint.
* **Purpose**: Tasks provide a way to track and manage work that does not directly contribute to user-facing features but is still necessary for the project's success.

**Relationship Between These Issue Types:**

* **Epics**: Are overarching goals that contain multiple stories and sometimes tasks.
* **Stories**: Are user-centric pieces of work that usually contribute to the completion of an epic.
* **Bugs**: Are issues that need fixing to ensure the quality and functionality of the stories or tasks.
* **Tasks**: Are general pieces of work that need to be done, not necessarily tied to a specific story or epic.

**Visual Example:**

Imagine you're working on an e-commerce website:

* **Epic**: "Shopping Cart System"
  + **Story**: "As a user, I want to add items to my shopping cart."
  + **Story**: "As a user, I want to view items in my shopping cart."
  + **Bug**: "Shopping cart does not update when an item is removed."
  + **Task**: "Set up the database schema for the shopping cart."