**SDLC:-**

Software Development Life Cycle (SDLC) is a process used by the software industry to design, develop and test high quality softwares. The SDLC aims to produce a high-quality software that meets or exceeds customer expectations, reaches completion within times and cost estimates.

The following figure is a graphical representation of the various stages of a typical SDLC.



There are various software development life cycle models defined and designed which are followed during the software development process. These models are also referred as Software Development Process Models". Each process model follows a Series of steps unique to its type to ensure success in the process of software development.

Following are the most important and popular SDLC models followed in the industry.

* Waterfall Model
* Iterative Model
* Spiral Model
* V-Model
* Big Bang Model

Other related models are Agile Model, RAD Model, Rapid Application Development and Prototyping Models.

**Waterfall Model :-**

The Waterfall Model was the first Process Model to be introduced. It is also referred to as a **linear-sequential life cycle model**.

It is very simple to understand and use. In a waterfall model, each phase must be completed before the next phase can begin and there is no overlapping in the phases.

The Waterfall model is the earliest SDLC approach that was used for software development.

The waterfall Model illustrates the software development process in a linear sequential flow. This means that any phase in the development process begins only if the previous phase is complete. In this waterfall model, the phases do not overlap.

Waterfall Model - Design

Waterfall approach was first SDLC Model to be used widely in Software Engineering to ensure success of the project. In "The Waterfall" approach, the whole process of software development is divided into separate phases. In this Waterfall model, typically, the outcome of one phase acts as the input for the next phase sequentially.

The following illustration is a representation of the different phases of the Waterfall Model.



The sequential phases in Waterfall model are −

* **Requirement Gathering and analysis** − All possible requirements of the system to be developed are captured in this phase and documented in a requirement specification document.
* **System Design** − The requirement specifications from first phase are studied in this phase and the system design is prepared. This system design helps in specifying hardware and system requirements and helps in defining the overall system architecture.
* **Implementation** − With inputs from the system design, the system is first developed in small programs called units, which are integrated in the next phase. Each unit is developed and tested for its functionality, which is referred to as Unit Testing.
* **Integration and Testing** − All the units developed in the implementation phase are integrated into a system after testing of each unit. Post integration the entire system is tested for any faults and failures.
* **Deployment of system** − Once the functional and non-functional testing is done; the product is deployed in the customer environment or released into the market.
* **Maintenance** − There are some issues which come up in the client environment. To fix those issues, patches are released. Also to enhance the product some better versions are released. Maintenance is done to deliver these changes in the customer environment.

## Waterfall Model - Disadvantages

The disadvantage of waterfall development is that it does not allow much reflection or revision. Once an application is in the testing stage, it is very difficult to go back and change something that was not well-documented or thought upon in the concept stage.

The major disadvantages of the Waterfall Model are as follows −

* No working software is produced until late during the life cycle.
* High amounts of risk and uncertainty.
* Not a good model for complex and object-oriented projects.
* Poor model for long and ongoing projects.
* Not suitable for the projects where requirements are at a moderate to high risk of changing. So, risk and uncertainty is high with this process model.
* It is difficult to measure progress within stages.
* Cannot accommodate changing requirements.
* Adjusting scope during the life cycle can end a project.
* Integration is done as a "big-bang. at the very end, which doesn't allow identifying any technological or business bottleneck or challenges early.

### What is Agile Model?

AGILE Model is a practice that promotes **continuous iteration** of development and testing throughout the software development lifecycle of the project. Both development and testing activities are concurrent unlike the Waterfall model.

The agile software development emphasizes on four core values.

1. Individual and team interactions over processes and tools
2. Working software over comprehensive documentation
3. Customer collaboration over contract negotiation
4. Responding to change over following a plan

**Note:-**

For Example In a waterfall model during performance testing of the application, If it is fails due to an architecture / design issue then will not have been having enough time to accommodate changes.

But in agile model this kind of problems are not coming.

Because in Agile Parallel development and testing will be happens.

Agile is nothing but small waterfall models.

### Agile Vs Waterfall Method

|  |  |
| --- | --- |
| **Agile Model** | **Waterfall Model** |
| * Agile method proposes incremental and iterative approach to software design | * Development of the software flows sequentially from start point to end point. |
| * The **agile process** is broken into individual models that designers work on | * The design process is not broken into an individual models |
| * The customer has early and frequent opportunities to look at the product and make decision and changes to the project | * The customer can only see the product at the end of the project |
| * Agile model is considered unstructured compared to the waterfall model | * Waterfall model are more secure because they are so plan oriented |
| * Small projects can be implemented very quickly. For large projects, it is difficult to estimate the development time. | * All sorts of project can be estimated and completed. |
| * Error can be fixed in the middle of the project. | * Only at the end, the whole product is tested. If the requirement error is found or any changes have to be made, the project has to start from the beginning |
| * Development process is iterative, and the project is executed in short (2-4) weeks iterations. Planning is very less. | * The development process is phased, and the phase is much bigger than iteration. Every phase ends with the detailed description of the next phase. |
| * Documentation attends less priority than software development | * Documentation is a top priority and can even use for training staff and upgrade the software with another team |
| * Every iteration has its own testing phase. It allows implementing regression testing every time new functions or logic are released. | * Only after the development phase, the testing phase is executed because separate parts are not fully functional. |
| * In agile testing when an iteration end, shippable features of the product is delivered to the customer. New features are usable right after shipment. It is useful when you have good contact with customers. | * All features developed are delivered at once after the long implementation phase. |
| * Testers and developers work together | * Testers work separately from developers |
| * At the end of every sprint, user acceptance is performed | * User acceptance is **performed** at the end of the project. |
| * It requires close communication with developers and together analyze requirements and planning | * Developer does not involve in requirement and planning process. Usually, time delays between tests and coding |

Agile and Waterfall model are two different methods for software development process. Though they are different in their approach, both methods are useful at times, depending on the requirement and the type of the project.

Imp Points for Agile Model:-

Agile Model break the product into small incremental builds. These builds are provided in iterations. Each iteration typically lasts from about one to three weeks. Every iteration involves cross functional teams working simultaneously on various areas like −

* Planning
* Requirements Analysis
* Design
* Coding
* Unit Testing and
* Acceptance Testing.

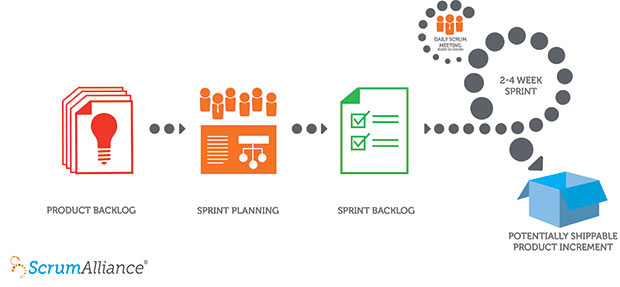
At the end of the iteration, a working product is displayed to the customer and important stakeholders.

Here is a graphical illustration of the Agile Model −



**Imp points of Agile:**

Many companies today are considering or using iterative methodologies like Agile and Scrum to deliver their software projects. This stemmed from challenges these businesses have faced with a waterfall delivery approach, where code is constantly in a state of flux, and development deadlines were missed routinely despite the best efforts of smart developers. Iterative methods tend to outperform traditional,



**Agile Project Management Tools**

* Easybacklog
* IceScrum
* XPWeb
* Jira
* Rally
* Bugzilla

**JIRA:-**

JIRA is a tool developed by Australian Company Atlassian. It is used for **bug tracking, issue tracking,** and **project management**. The name "JIRA" is actually inherited from the Japanese word "Gojira" which means "Godzilla".

The basic use of this tool is to track issues, and bugs related to your software and[Mobile](http://www.guru99.com/mobile-testing.html)apps. It is also used for project management. The JIRA dashboard consists of many useful functions and features which make handling of issues easy.

The JIRA current version is 7.0.x.

The Jira's main competitor is  [Bugzilla](https://en.wikipedia.org/wiki/Bugzilla).

**What is Scrum?**

## Scrum is an iterative and incremental agile software development framework for managing product development.

## It defines "a flexible, holistic product development strategy where a development team works as a unit to reach a common goal",challenges assumptions of the "traditional, sequential approach.to product development, and enables teams to self-organize by encouraging physical co-location or close online collaboration of all team members, as well as daily face-to-face communication among all team members and disciplines involved.

## A key principle of Scrum is the dual recognition that customers will change their minds about what they want or need (often called requirements volatility) and that there will be unpredictable challenges—for which a predictive or planned approach is not suited. As such, Scrum adopts an evidence-based empirical approach—accepting that the problem cannot be fully understood or defined up front, and instead focusing on how to maximize the team's ability to deliver quickly, to respond to emerging requirements, and to adapt to evolving technologies and changes in market conditions.

## Users/Roles

There are three core roles in the Scrum framework. They represent the scrum team. Although other roles involved with product development may be encountered, Scrum does not define any team roles other than those described below.

**1)Product Owner**

**2)Scrum Master**

**3)Development Team**

### Product owner:

The product owner represents the product's stakeholders and the voice of the customer; and is accountable for ensuring that the team delivers value to the business.

The product owner writes customer-centric items (typically user stories), prioritizes them based on importance and dependencies, and adds them to the product backlog in order of delivery.

**Scrum teams should have one product owner. This role should not be combined with that of the scrum master.**

The product owner should focus on the business side of product development and spend the majority of their time liaising with stakeholders and should not dictate how the team reaches a technical solution.

This role is equivalent to the customer representative role in some other agile frameworks.

**scrum master:-**

He is a person coordinating the backlog meetings, sprint planning meetings, Daily scrum meeting and sprint showcase meeting

**Sprint :**

A sprint (or iteration) is the basic unit of development in Scrum. The sprint is a timeboxed effort; that is, it is restricted to a specific duration.

The duration is fixed in advance for each sprint and is normally between one week and one month, with two weeks being the most common.

Each sprint starts with a sprint planning event that aims to define a sprint backlog, identify the work for the sprint, and make an estimated forecast for the sprint goal.

Each sprint ends with a sprint review and sprint retrospective,that reviews progress to show to product owner and identify lessons and improvements for the next sprints.

Scrum emphasizes working product at the end of the sprint that is really done. In the case of software, this likely includes that the software has been fully integrated, tested and documented, and is potentially shippable.

**Different Type of Issues In JIRA :**

* EPIC
* STORY
* TASK
* BUG

**Epic**: Epic is nothing high level grouping of components,(high level story).

(Epic contain logical related stories with respect to functionality).

Epic is a Big Story.

**Story :**

A story is a very high-level definition of a requirement, containing just enough information so that the developers can produce a reasonable estimate of the effort to implement it.

--> In a story we can create story points(Estimation).

-->In a story we can create task and sub task also.

**Project Backlog meeting:**

In project Backlog Meeting The Product Owner, change Manager ,Scrum Master ,Business Analysist and Subject Matter Experts will Involve.

In Project Backlog Meeting They Will give the Priority for Issues.

(That means what are the stories are required, to complete the project they will plan).

Note : The change manager generally concentrate the budget and integration of all teams with all technologies

### Sprint planning:-

At the beginning of a sprint, the scrum team holds a sprint planning event to:

* Communicate the scope of work that is intended to be done during that sprint
* Select product backlog items that can be completed in one sprint.
* Prepare a sprint backlog that includes the work needed to complete the selected product backlog items
* Once the development team has prepared their sprint backlog, they forecast (usually by voting) which tasks will be delivered within the sprint.

**Note:-**

Scrum master will conduct sprint planning meeting ( It mostly 1.30 to 2hrs meeting, for every 15 days they will conduct meeting).

--->In general Every Sprint Duration is 15 days.

In Spring Planning meeting Developer Team Will involve along with Subject Matter Experts.

That Means the Product Owner not involving in the Sprint Planning Meeting.

The subject master experts will explain about the requirements to all developers

After explanation the scrum master will asking about estimation for stories to all the developers.

The estimation is nothing but story points

Note:- in Agile Every Developer has an independency of taking stories , scrum master will act as just a co-oridinator . He is not assigning any task/Story to developers.

### Daily scrum: (maximum 15 min the scrum master will conduct Daily Scrum meeting)

Each day during a sprint, the team holds a daily scrum (or [stand-up](https://en.wikipedia.org/wiki/Stand-up_meeting)) with specific guidelines:

* All members of the development team come prepared. The daily scrum:
  + starts precisely on time even if some development team members are missing
  + should happen at the same time and place every day
  + is limited ([timeboxed](https://en.wikipedia.org/wiki/Timeboxing)) to fifteen minutes.
* Anyone is welcome, though only development team members should contribute.
* During the daily scrum, each team member typically answers three questions:
  + **What did I complete yesterday that contributed to the team meeting our sprint goal?**
  + **What do I plan to complete today to contribute to the team meeting our sprint goal?**
  + **Do I see any impediment that could prevent me or the team from meeting our sprint goal?**

Any impediment (e.g., stumbling block, risk, issue, delayed dependency, assumption proved unfounded). identified in the daily scrum should be captured by the scrum master and displayed on the team's scrum board or on a shared risk board, with an agreed person designated to working toward a resolution (outside of the daily scrum). No detailed discussions should happen during the daily scrum.

**Note:-** When Developer is Implementing the coding The Developer will add work log information is as a comment.

And the Log information showing to all Members .

### JIRA Scheme

Inside JIRA scheme, everything can be configured, and it consists of

* **Workflows**
* **Issue Types**
* **Custom Fields**
* **Screens**
* **Field Configuration**
* **Notification**
* **Permissions**

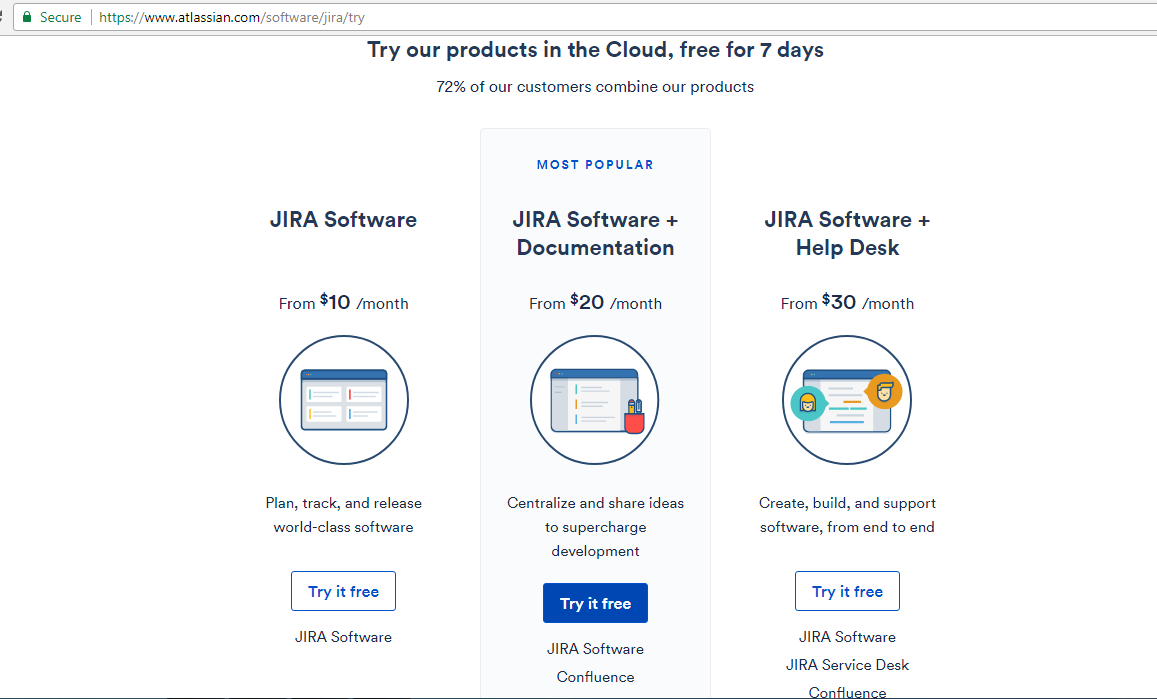
**JIRA Software :**

JIRA Software Is There in Colud , In Colud It is Free for 7 days.

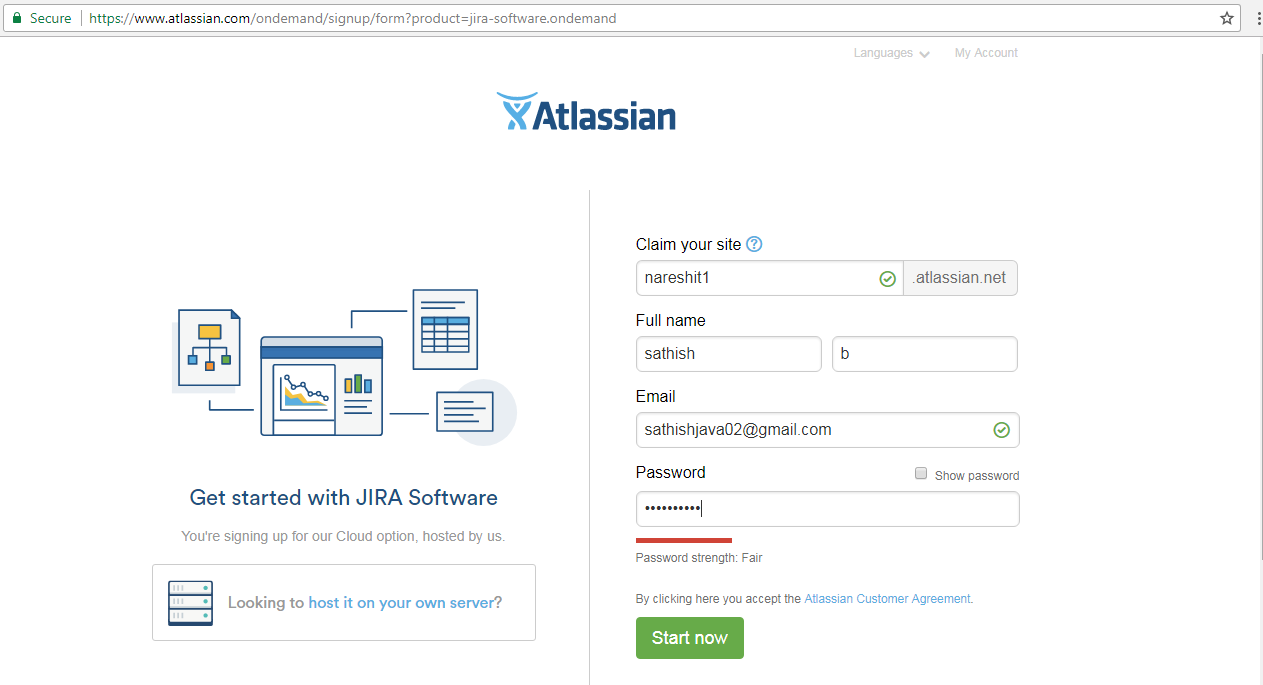
The JIRA URL for Registration : <https://www.atlassian.com/software/jira/try>.

The JIRA software also possible to install in our local System . And It is Free for 30 days.

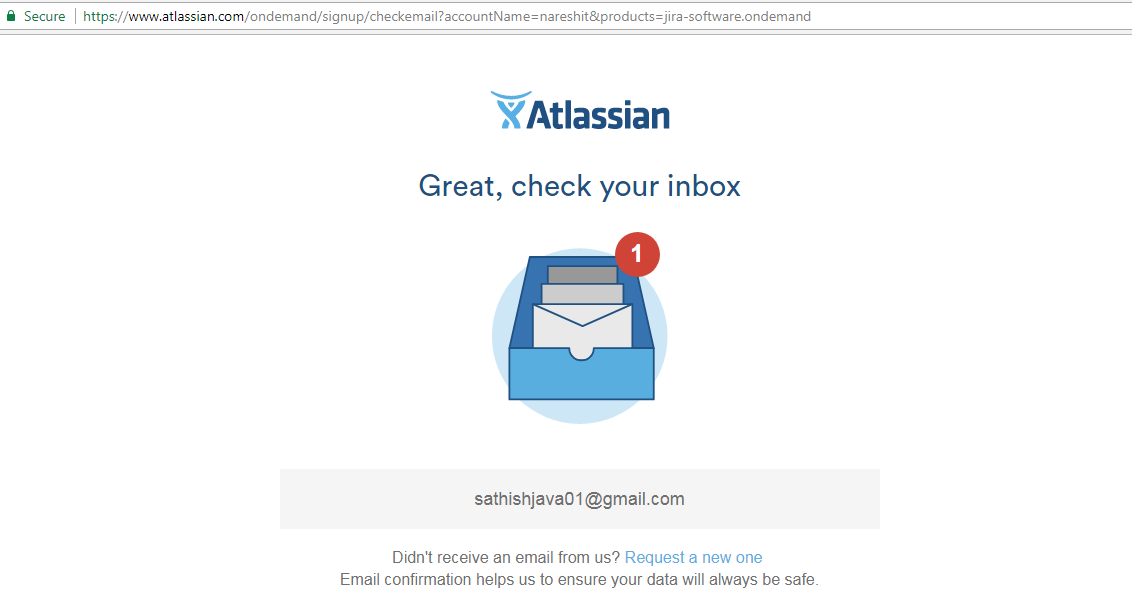
The JIRA Server Software Download URL : <https://www.atlassian.com/software/jira/download>



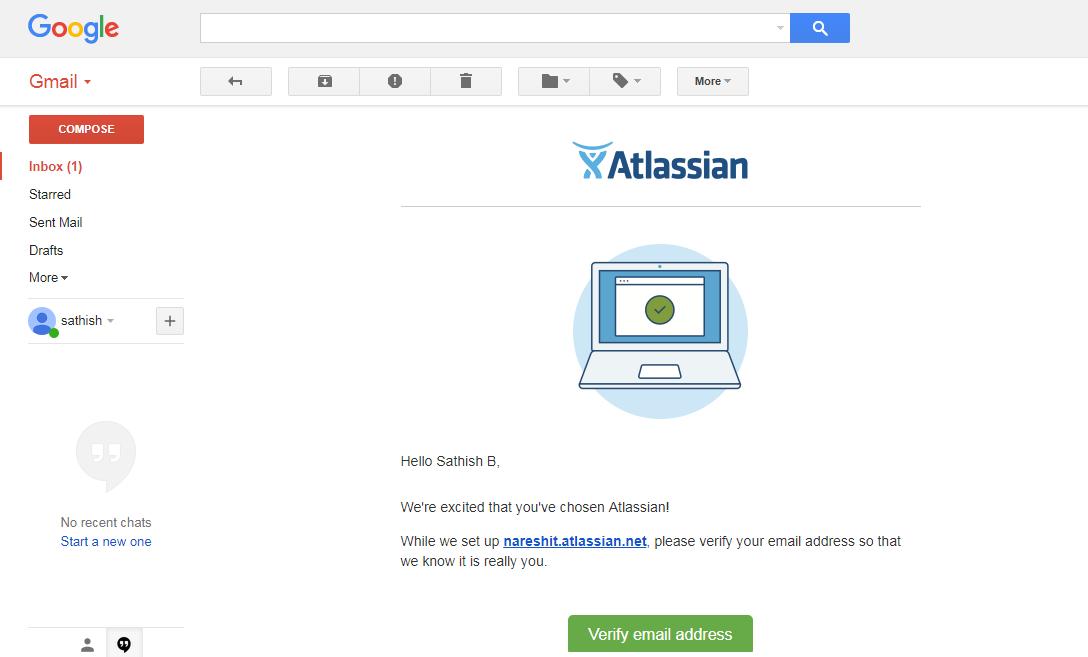
Here iam showing How to Register in JIRA Colud :



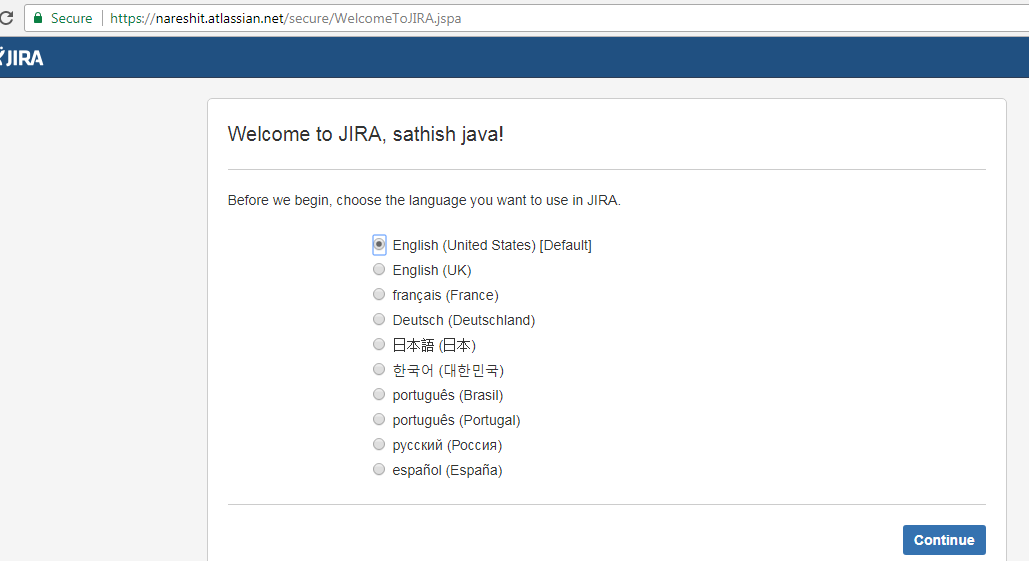
After Entering All Details Click on Start Now Button Then the following message is displaying .



Open Your Mail and Verify Email Address to activate Your JIRA Acocunt.

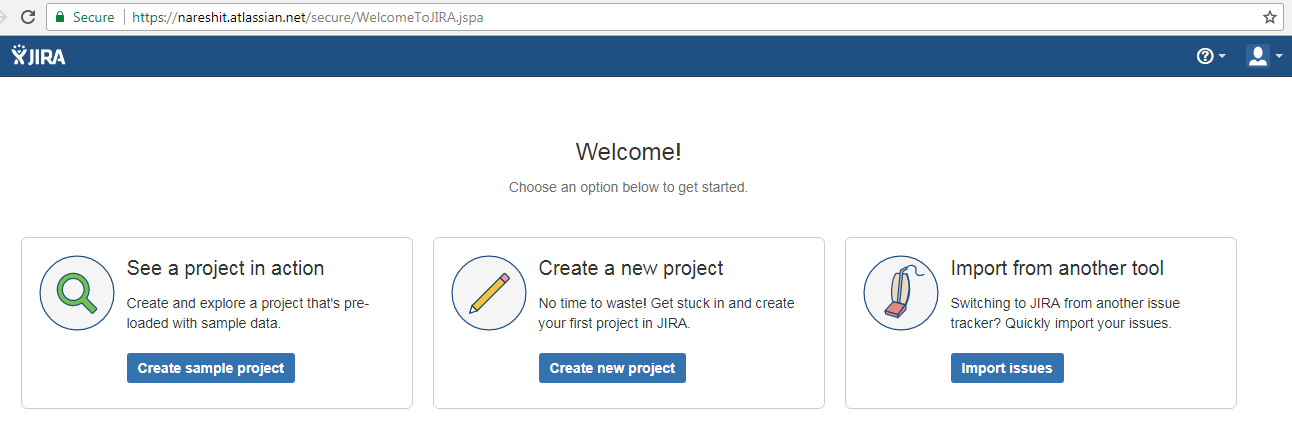


When we are clicking Verify Email Address the following Window is showing and select the Language and Contune.



After Clicking The Contine Button We will Login Into Our JIRA Account. In First Time Login

The following window is showing.

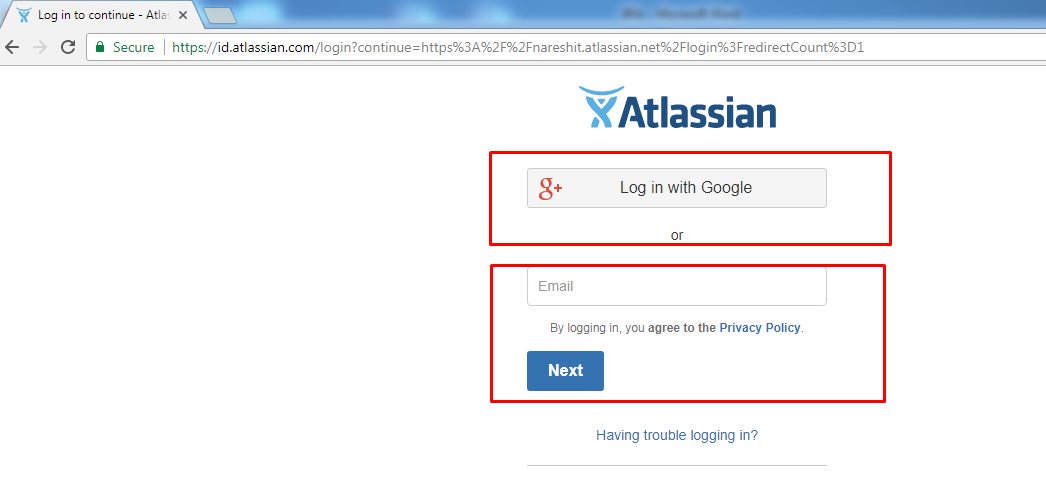


Second time Onwords if we want to login then in the brower Enter your Cliam Site Domain like

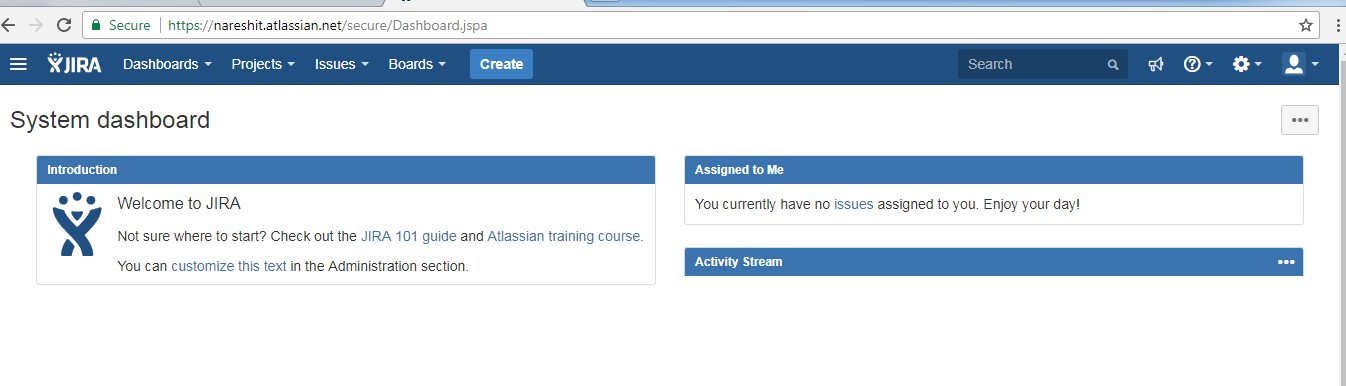
<https://nareshit.atlassian.net>.

When we are entering the above URL it is redirecting Login Site with the following URL:

<https://id.atlassian.com/login?continue=https%3A%2F%2Fnareshit.atlassian.net%2Flogin%3FredirectCount%3D1>



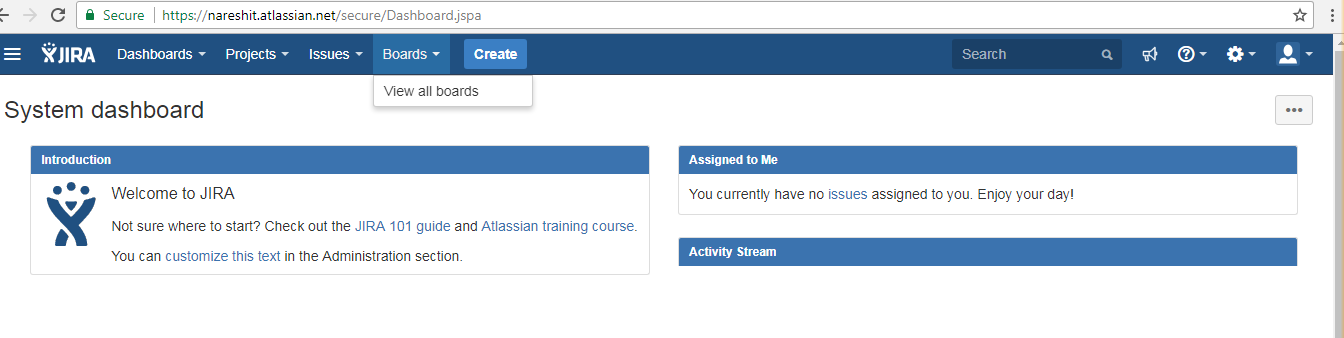
After Login We will get the Following Dashboard.

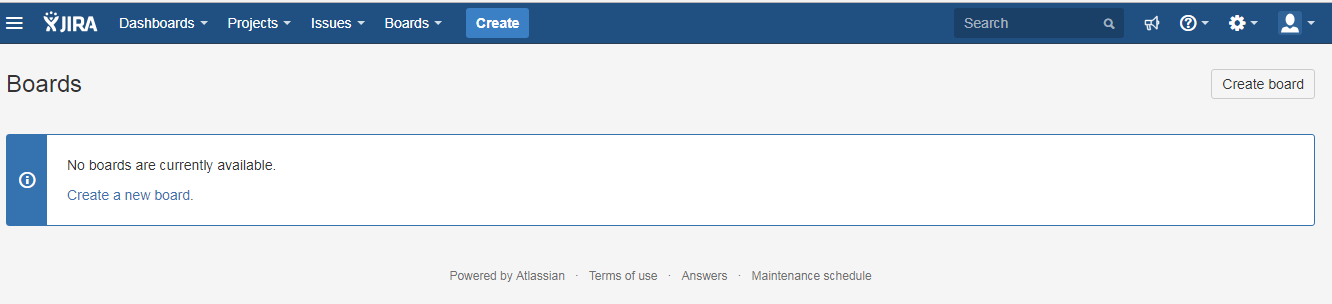


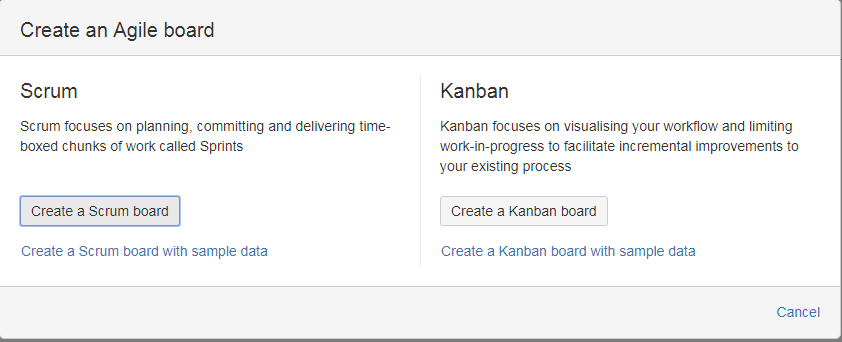
After Login It is mainly showing Dashboards, Projects, Issues ,Boards ,Create Options .

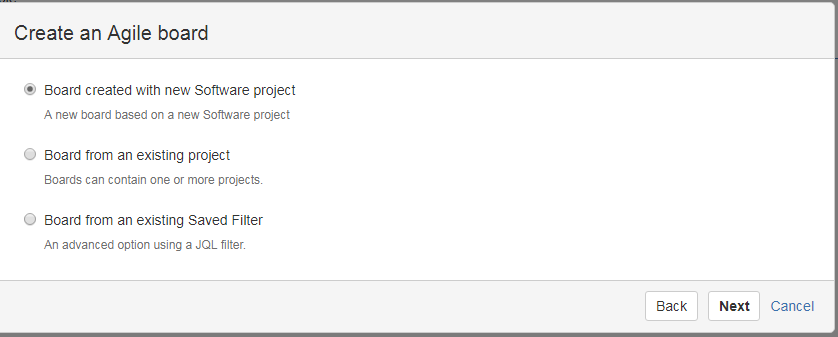
Then To Start with JIRA First Create Scrum Board By Clicking Board Menu .

When we are creating ScrumBoard , The Board will create Software Project.

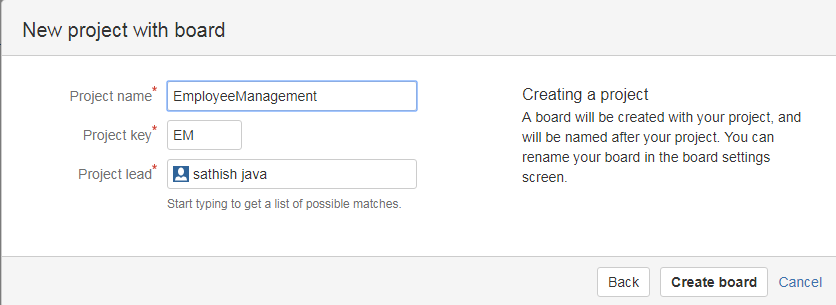




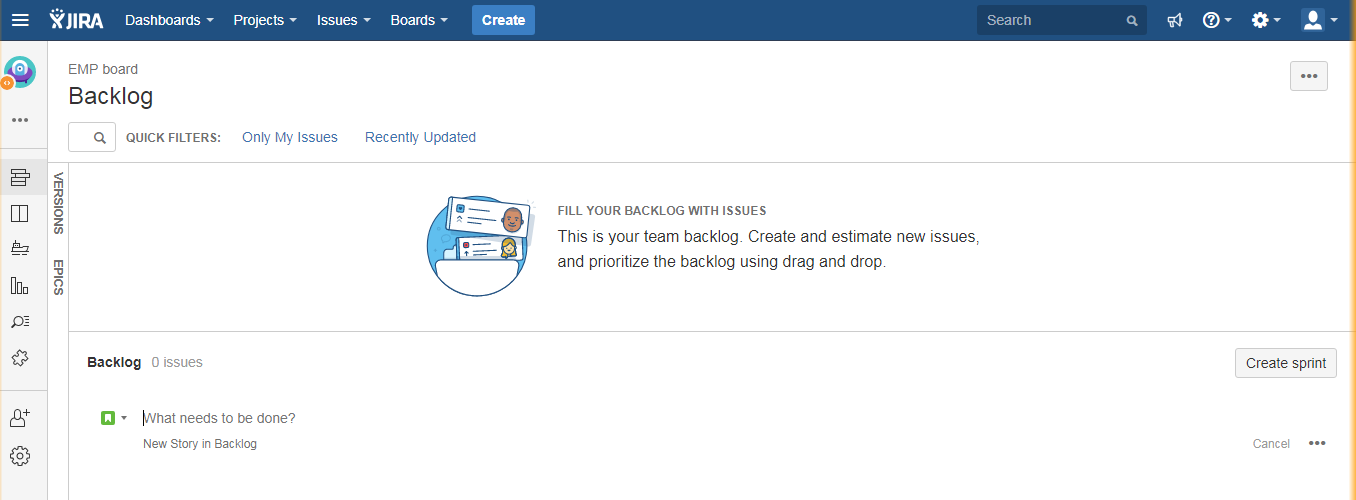




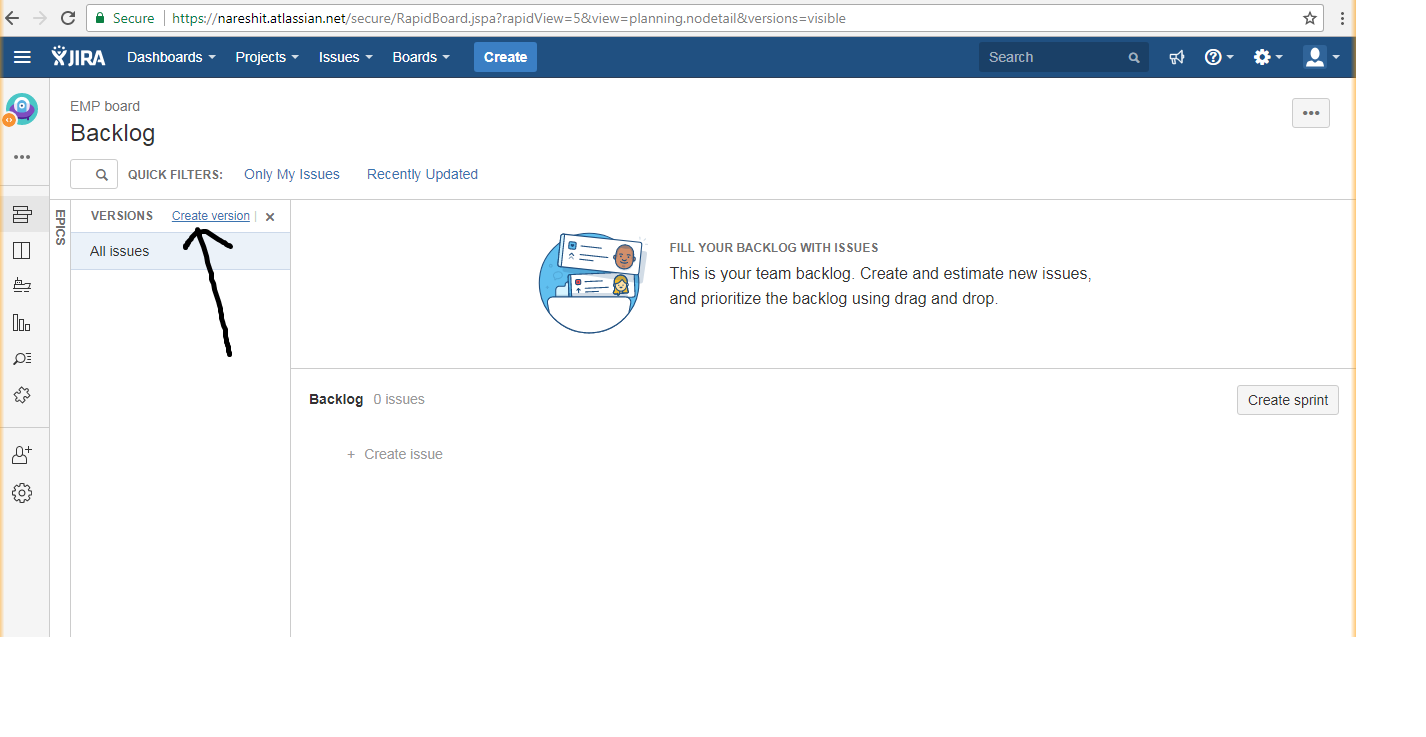
Enter Project Name And Project Key as follows

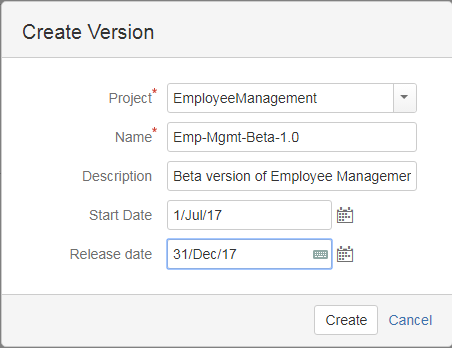


After Creating The Board A backlog window is showing to create Versions, Issues ,Sprint etc….

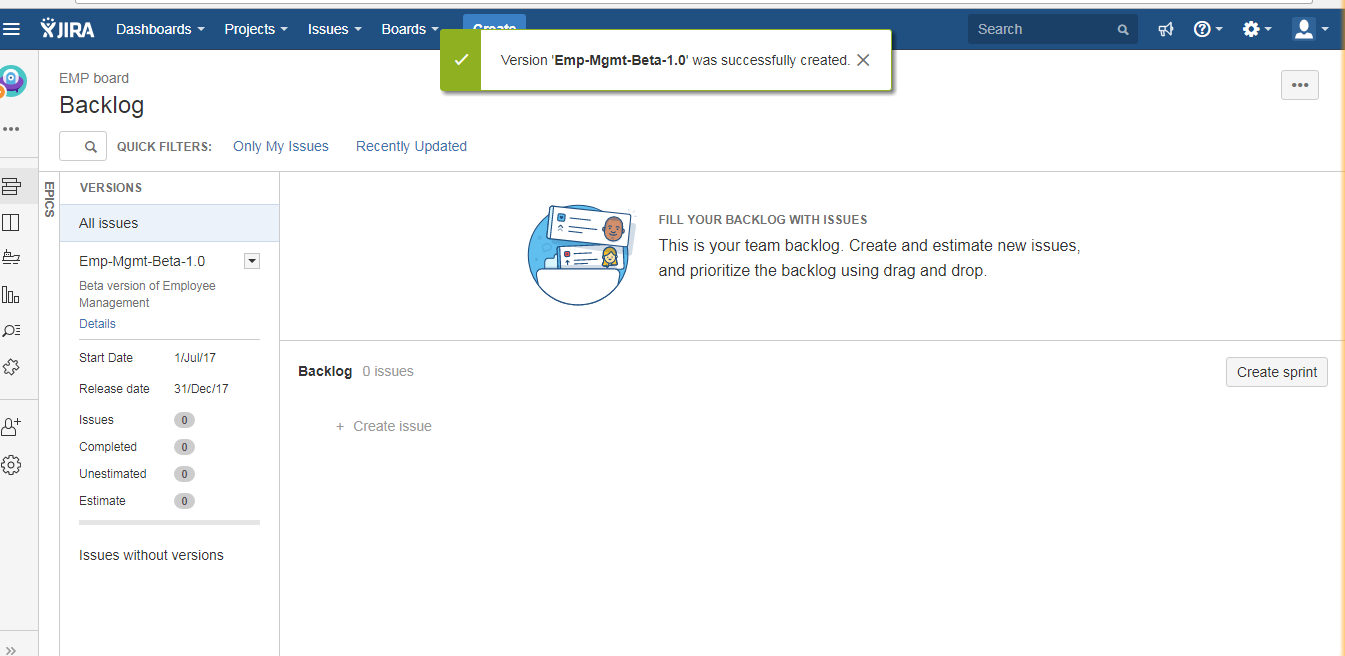


First Create The Version As for the Requirement as follows





When we are clicking the Create Button We will get the Following Message.



After creating The Versions Then start the Creating of Issues .

The Issues are Mainly the following Types

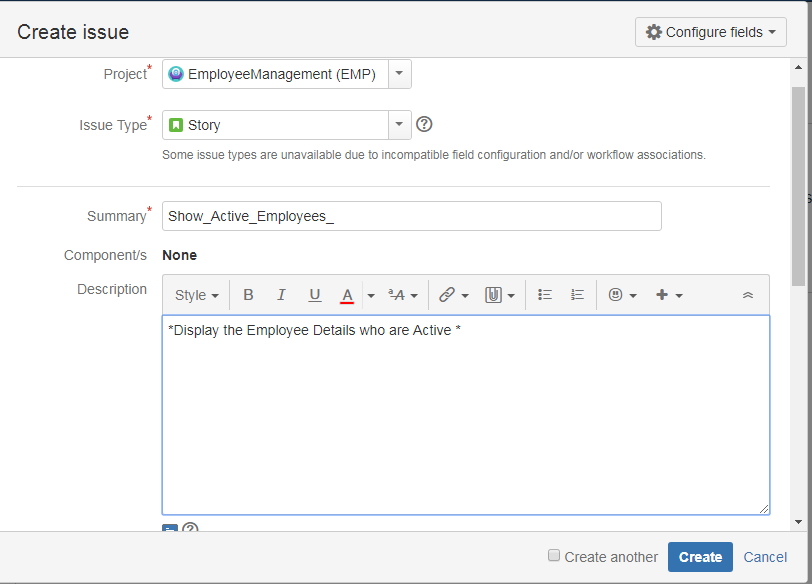
1)EPIC

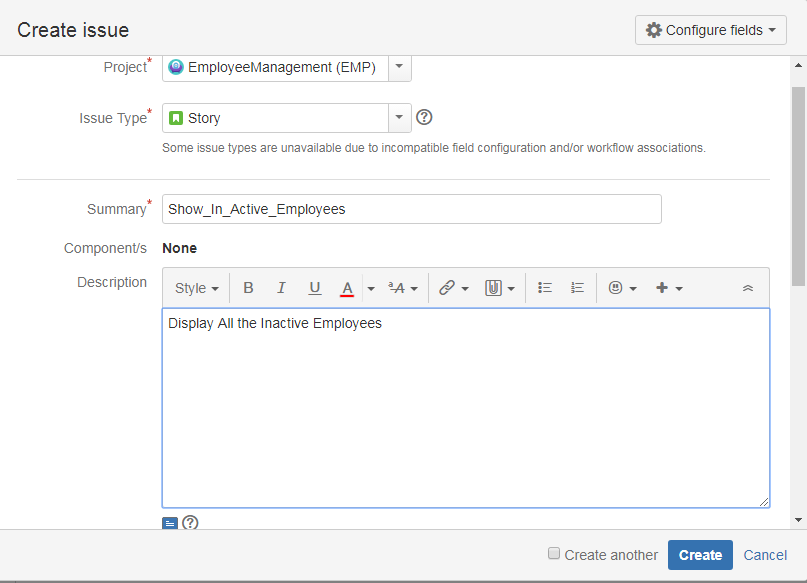
2)STORY

3)TASK

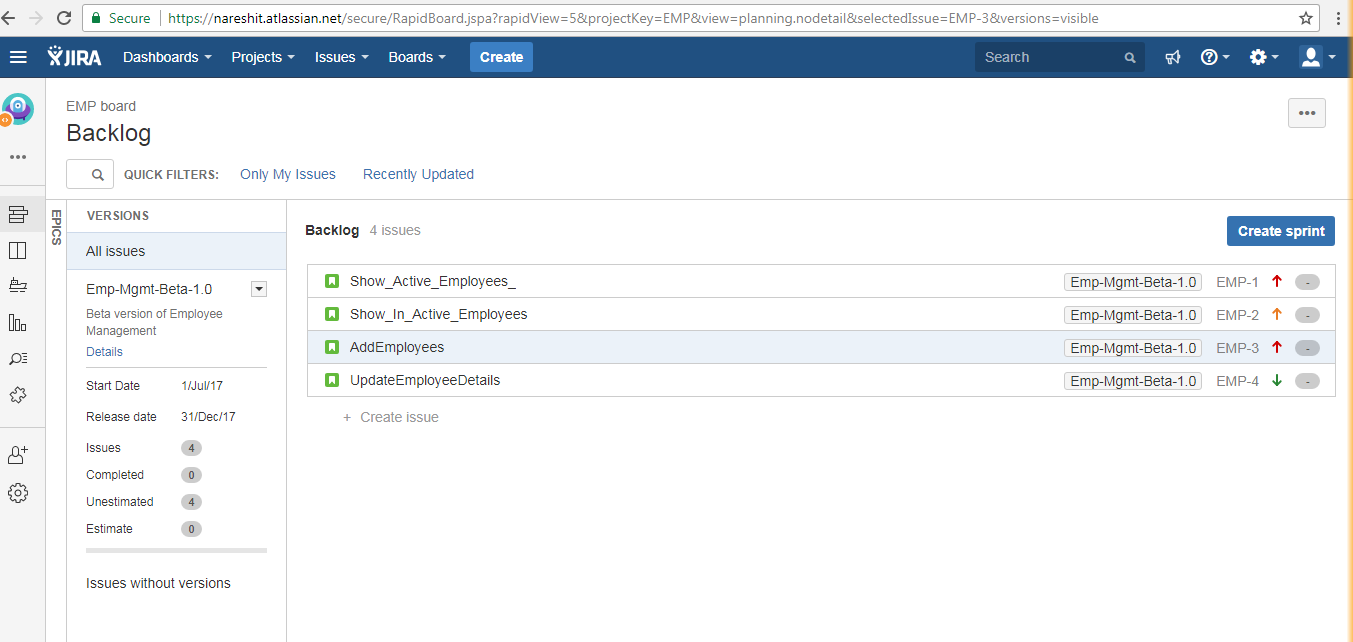
4)BUG

Here Iam showing how to create a First Story .



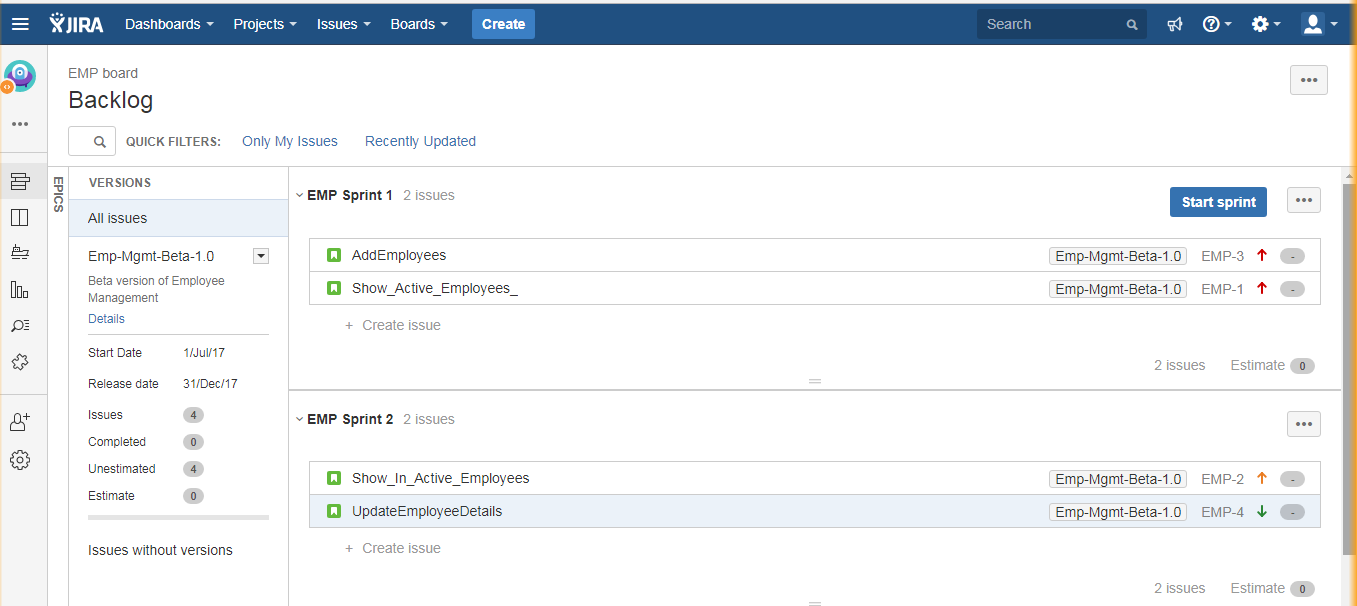


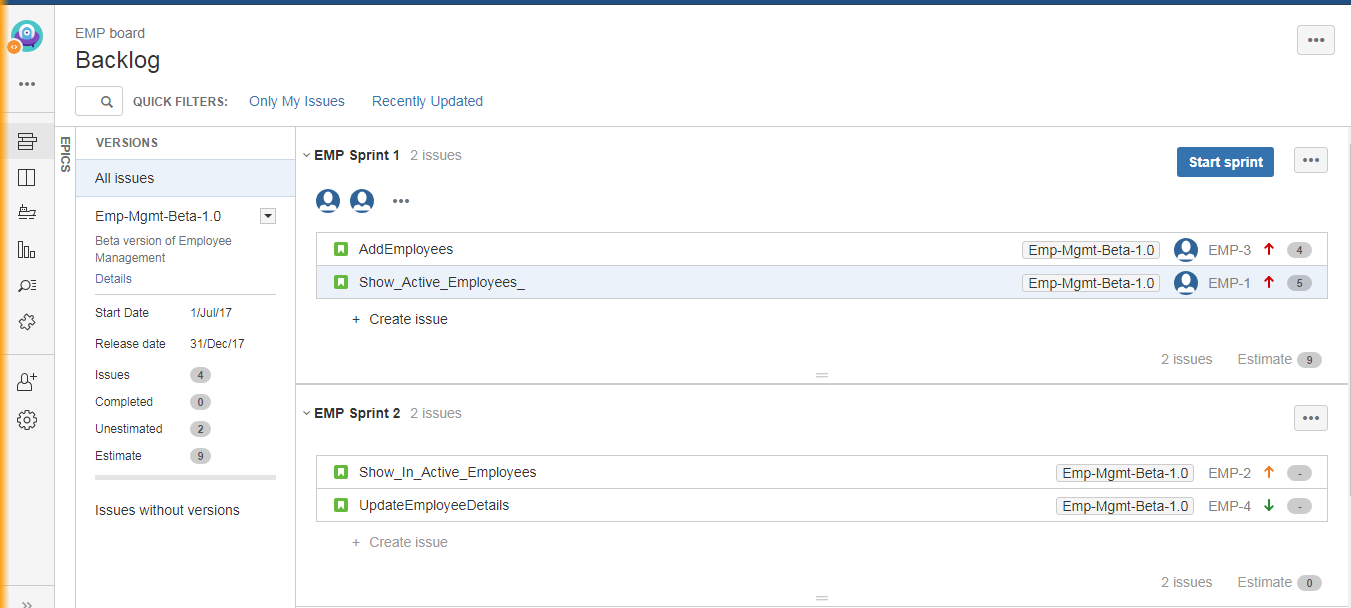
Here Iam showing how to create a Seond Story .



After creating all the Issues ,Then Create Sprint .

After Creating the Sprint drag the Issues into Sprint as follows .





**Note:-**

Scrum master will conduct sprint planning meeting ( It mostly 1.30 to 2hrs meeting, for every 15 days they will conduct meeting).

--->In general Every Sprint Duration is 15 days.

In Spring Planning meeting Developer Team Will involve along with Subject Matter Experts.

That Means the Product Owner not involving in the Sprint Planning Meeting.

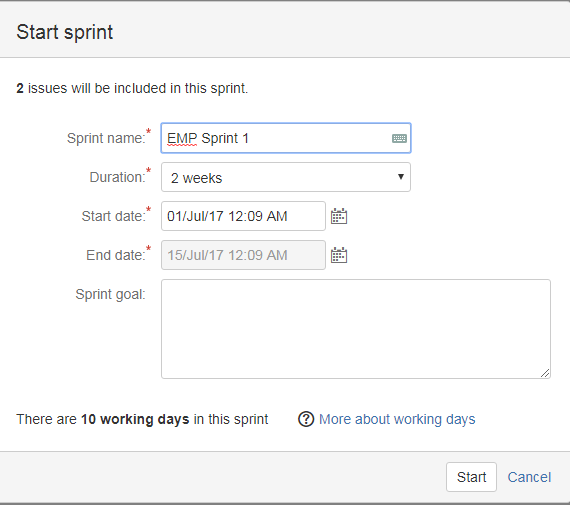
The subject master experts will explain about the requirements to all developers

After explanation the scrum master will asking about estimation for stories to all the developers.

The estimation is nothing but story points

Note:- in Agile Every Developer has an independency of taking stories , scrum master will act as just a co-oridinator . He is not assigning any task/Story to developers.

**After Competing The Spring Paining Meeting The Scrum Master will start The Sprint :**



--->In general Every Sprint Duration is 15 days. That is Two Weaks .

Just After Starting The Sprint ,The Scurm Board Is showing with The following Options .

1)TO DO

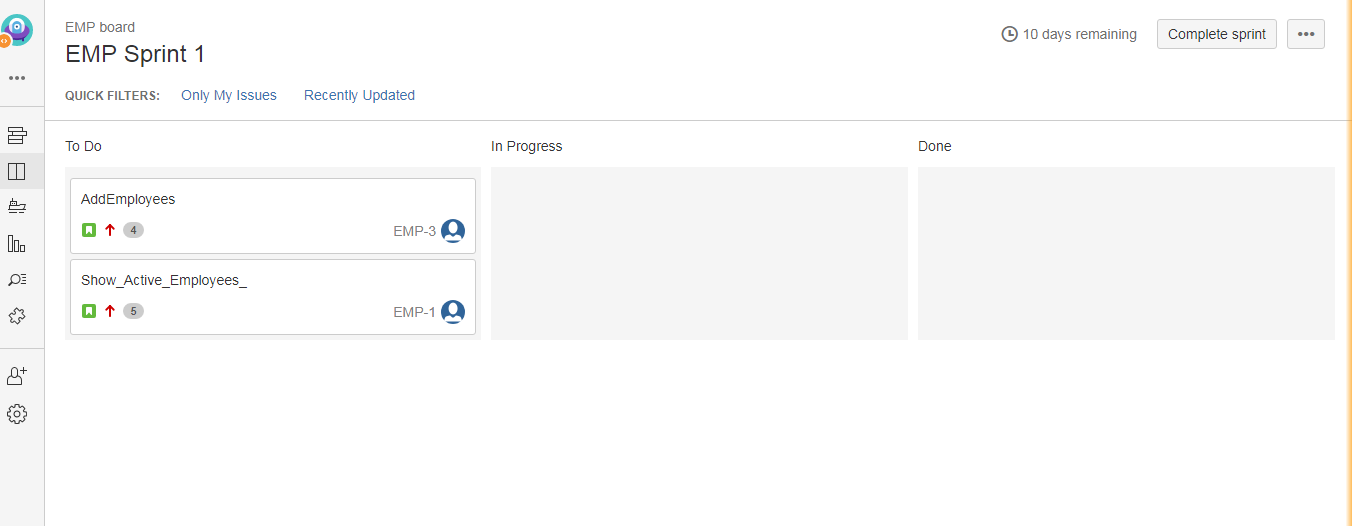
2)In Progress

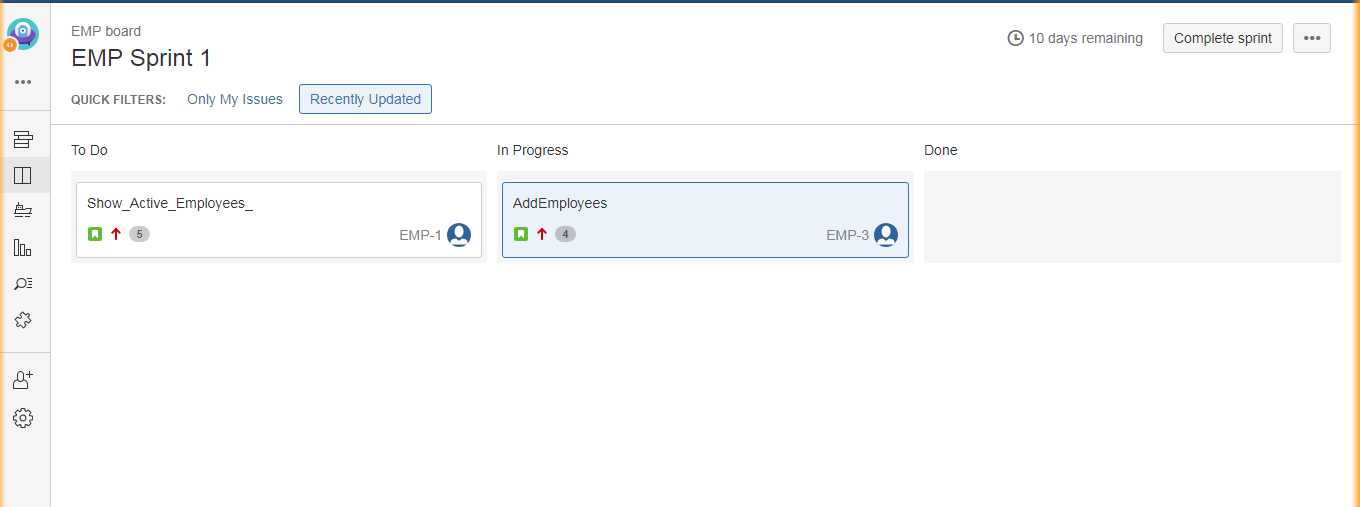
3)Done

All The User Stories By default showing in TODO.

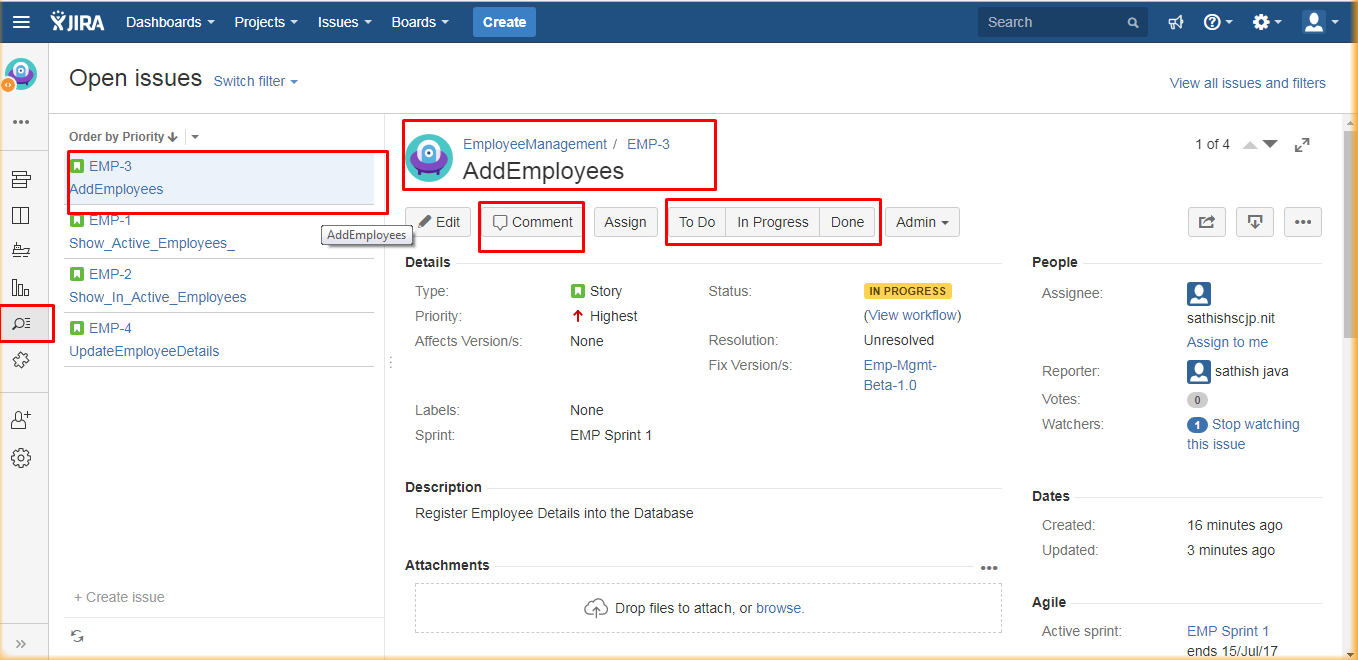
When developer will start the work he will move to InProgress.

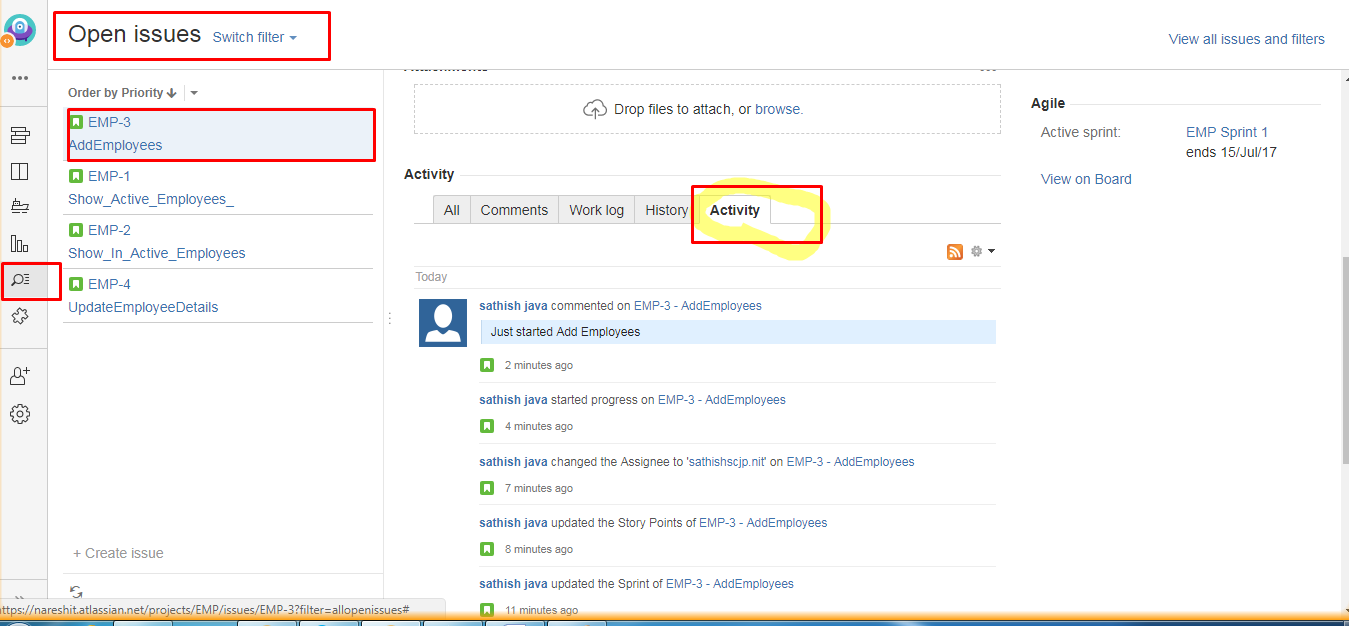
After Completing the Story he will move to DONE.

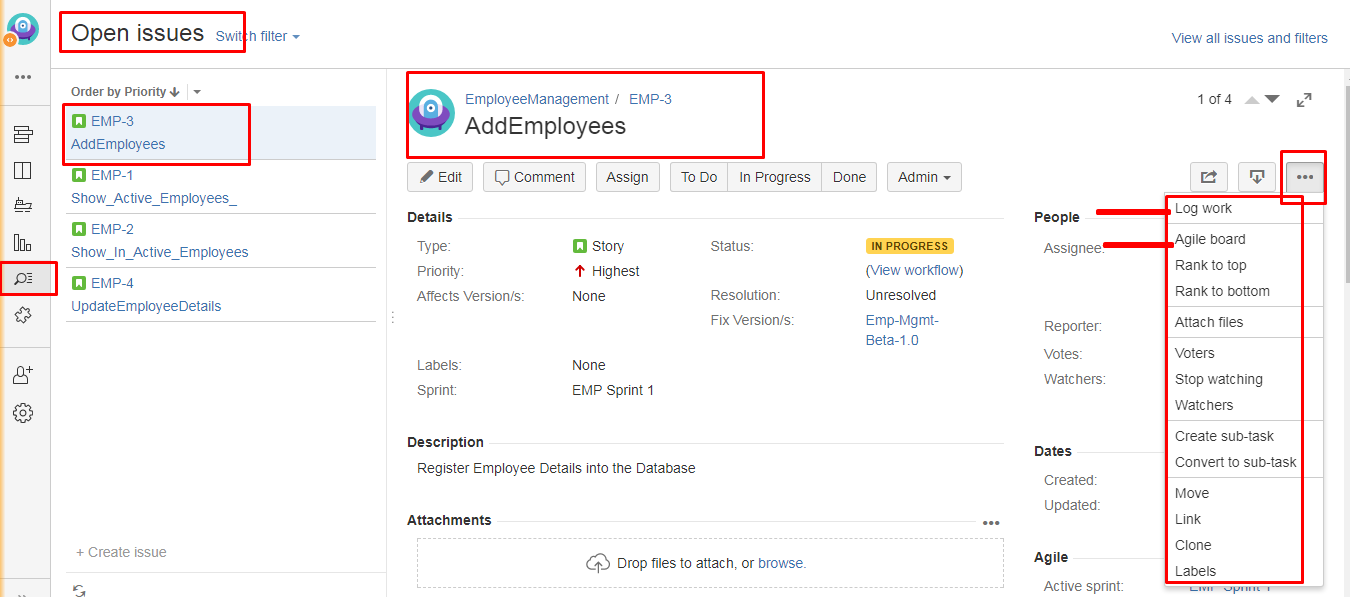




While Working Developer will add the Comments ,The Comments are showing as worklog.



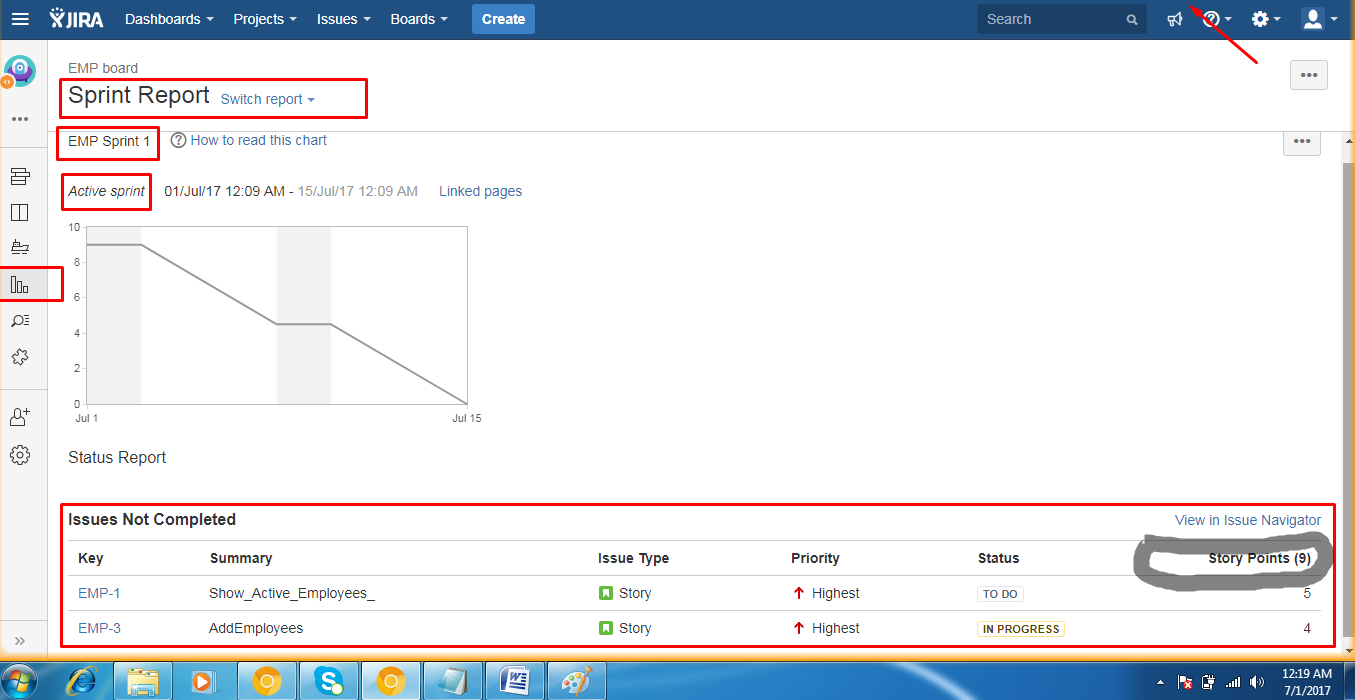




**Reports:-**

to calculate velocity we can use reports.

velocity -->how many no of story points can be develiered by team in a sprint is called velocity.



### Agile Scrum: What is it, and why should I care?

Agile is a system of methods to deliver iteratively. Agile Scrum adds a few rules to the structure to make processes more collaborative and team driven. Below is a high-level overview of Agile structural elements:

**1.** **A sprint (iteration)** that typically runs between one and three weeks, delivering a releasable bit of the product that may consist of bug fixes, new features, UI improvements, or some combination of these and other projects.

* Each sprint has a sprint planning session, where the team commits to the user stories, or subprojects, they can complete during that sprint and break the stories down into tasks.
* Short daily huddles known as the “daily scrum” or “daily standup” where the team provides brief status reports to determine where it is and what can be accomplished that day.
* A retrospective at the end of the sprint that enables the team to determine where they can improve and how they can build more velocity going forward.
* A demo that allows the team to showcase its delivery to stakeholders or users.

2. A release which includes a definitive set of features/stories that will be released over time. A release is usually a predetermined number of sprints. Typically, a team doesn't perform a release planning activity until it has demonstrated a sustained velocity that allows for effective planning.

3. Roles

* A Product Owner who owns the entire list, or backlog, of stories, prioritizes them, and makes sure the customer/stakeholder input is incorporated.
* A Scrum Master (project manager) who facilitates and guides the team towards success on a day-to-day basis.
* The team that is delivering the product.

While the description above is not comprehensive , there is also a change in mindset when embarking on the Agile journey. From my own experience, here are some aspects to consider that are usually quite useful in attaining better velocity and predictable delivery from your sprints. I recommend reading the article “A brief introduction to Scrum” linked above if you can; it is the basis for these thoughts.

1. **Think continuous integration**. Without automated tools to build, test, and release, your velocity will be limited to the speed of humans. Automation can greatly accelerate your delivery; update your tools alongside your features at all times

Think minimal documentation and written content. Yes, documentation is required; however, every bit of documentation should have purpose: improving the team’s capability/velocity.

* API documents are useful for the team, enabling them to rapidly distribute work without excessive handholding.
* A 20-page requirements document is not as useful as a product owner sitting with the team and working through the requirements in person. In past implementations, I have asked my team not to bother documenting the issues found within a sprint since they need to be collaboratively fixed immediately.
  + Take shortcuts...that’s what Agile allows you to do. But do not take shortcuts that land you in trouble down the road! Basic discipline is still a pre-requisite to building good products.

1. Accept and celebrate change. Agile mastery requires complete acceptance to change and an ability to adapt quickly. Guess what? If your requirements and business expect your code to change all the time, you are in a state of flux with lots of unknowns. But this is good news, since the unknown can drive the greatest growth machines.
2. Commitment. No Agile team ever delivered well without committing to the outcome of the sprint and going after it. Over time, long hours will subside and predictability will increase as you begin settling into a good, repeatable velocity and are also able to size your stories consistently.
3. Generalized capability within Scrum teams: Teams with a generalized capability tend to accept more stories and deliver more velocity over time since their backlog will almost never be low. While it is acceptable for an individual developer to have a specialization, it is recommended to build teams where any member could take on all stories in the backlog. This will require cross-training and learning and a focus on enabling and empowering each individual on the team to take on any task.

.