<epam>

# Data Quality

Introduction to Quality Assurance

TRAINING

CENTER

<epam>

# Agenda

- Software Testing
- JIRA introduction

# SOFTWARE TESTING

# Why is Testing Necessary

UNCOVERS DEFECTS BEFORE IT IS DELIVERED TO CUSTOMER

UNTESTED SOFTWARE COSTS THE CUSTOMER BUSINESS FAILURE TOO

PROVIDE FEEDBACK TO ALL STAKEHOLDERS

AS LATER THE STAGE IN WHICH BUG IS IDENTIFIED, MORE IS THE COST TO RECTIFY IT

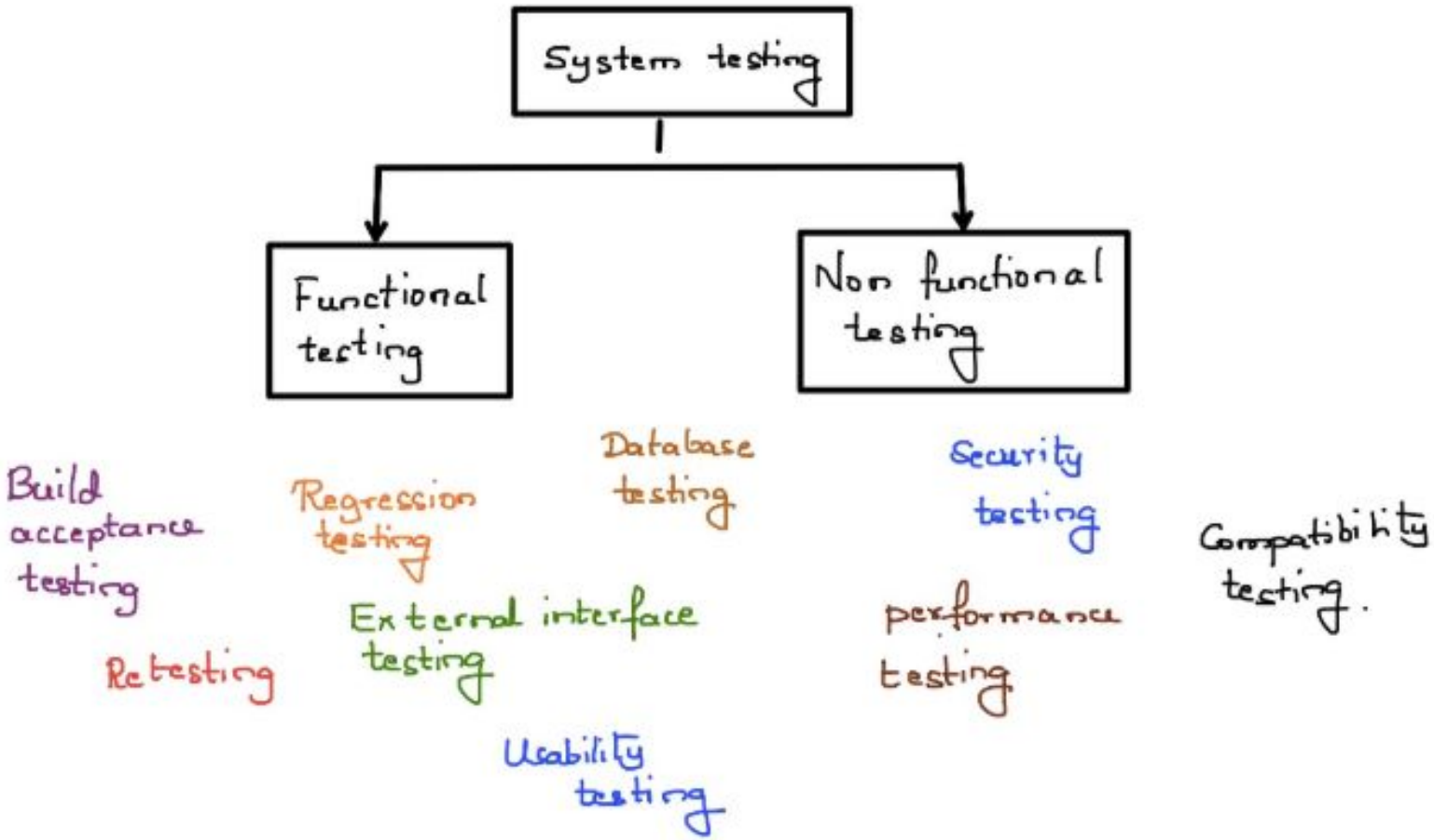
PROVIDE FEEDBACK

REDUCE IMPLEMENTATION COST

GAINING END USERS CONFIDENCE

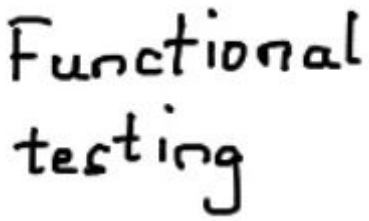
STAY IN THE BUSINESS

# Main Software Testing Types



The functions are “**what**” the system should do.

Functional tests tend to answer the question of "**does this particular feature work**“.




## Non-Functional testing

Non-functional tests tend to answer the question of “**how well**” the system behaves.

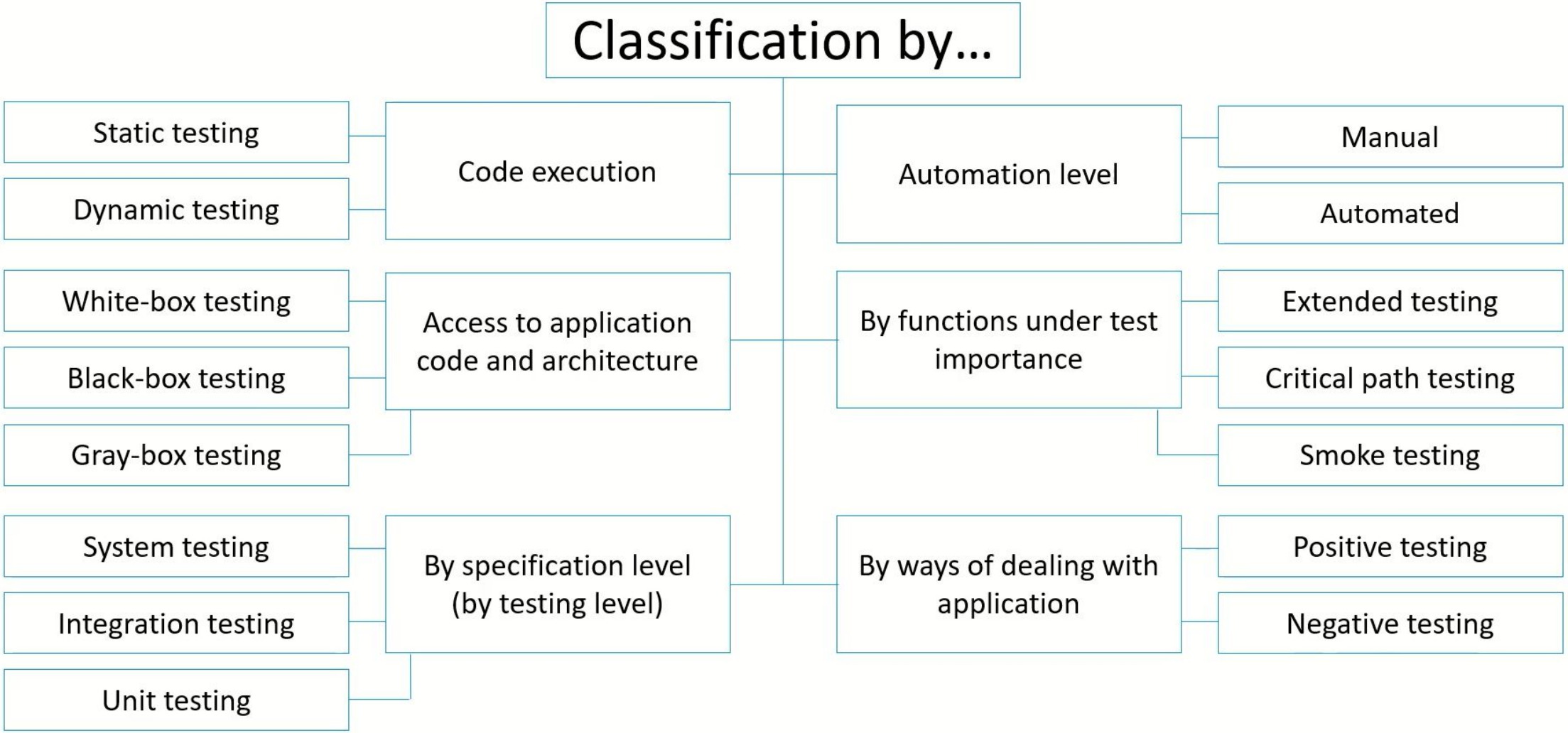
Security testing

Usability testing

performance testing

Compatibility testing.

# Functional Testing



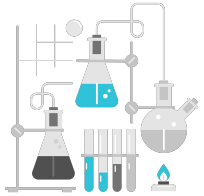
# Test types by testing depth

**Smoke**

a subset of all defined test cases that cover the main functionality of a component or system, the most crucial functions.

**Critical path**

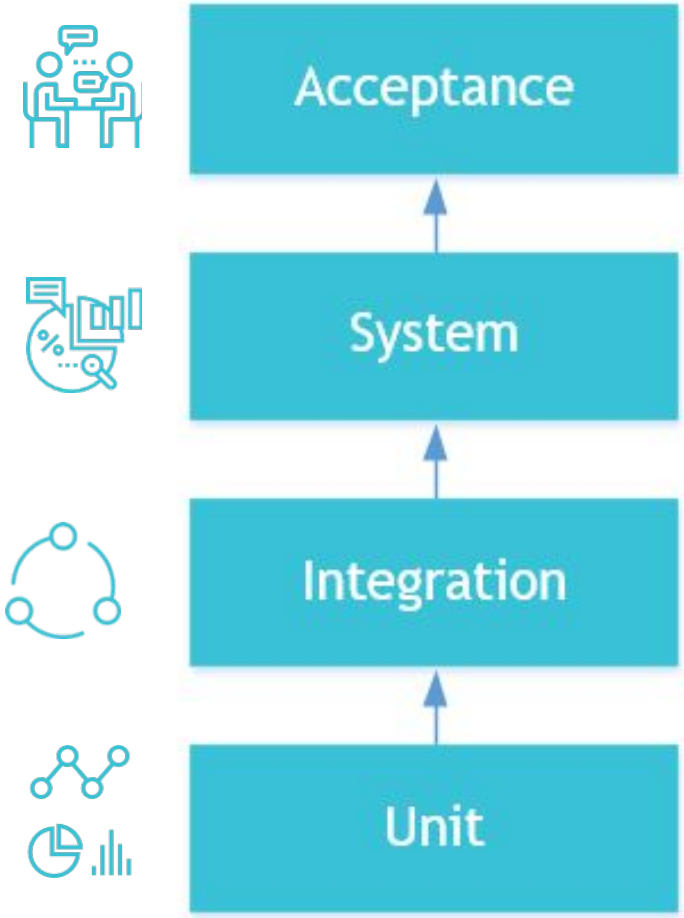
test cases that cover the functionality used most of the time by the majority of users.



**Extended path**

test cases that cover the “nice-to-have” functionality (not used most of the time by the majority of

# Software Testing Levels




focuses on the behavior and capabilities of a whole system to evaluate system's compliance with the business requirements

focuses on the behavior and capabilities of a whole system to evaluate system's compliance with the requirements

focuses on interactions between components to expose faults in the interaction between integrated units

focuses on components that are separately testable

# Software Testing Levels

## UNIT

- Who: DEV, QA
- When: Component is developed
- Why: To validate that each unit of the software performs as designed
- How: White-box testing

## INTEGRATION

- Who: DEV, QA
- When: Units to be integrated are developed
- Why: To expose faults in the interaction between integrated units
- How: White, Black or Grey-box testing

## SYSTEM

- Who: QA
- When: Separate units are integrated into System
- Why: To evaluate system's compliance with the specified requirements
- Black-box testing

## ACCEPTANCE

- Who: Business users
- When: Component is developed or units are integrated into System
- Why: To evaluate system's compliance with the business requirements; assess whether it is acceptable for the delivery
- Black-box testing

# JIRA INTRODUCTION

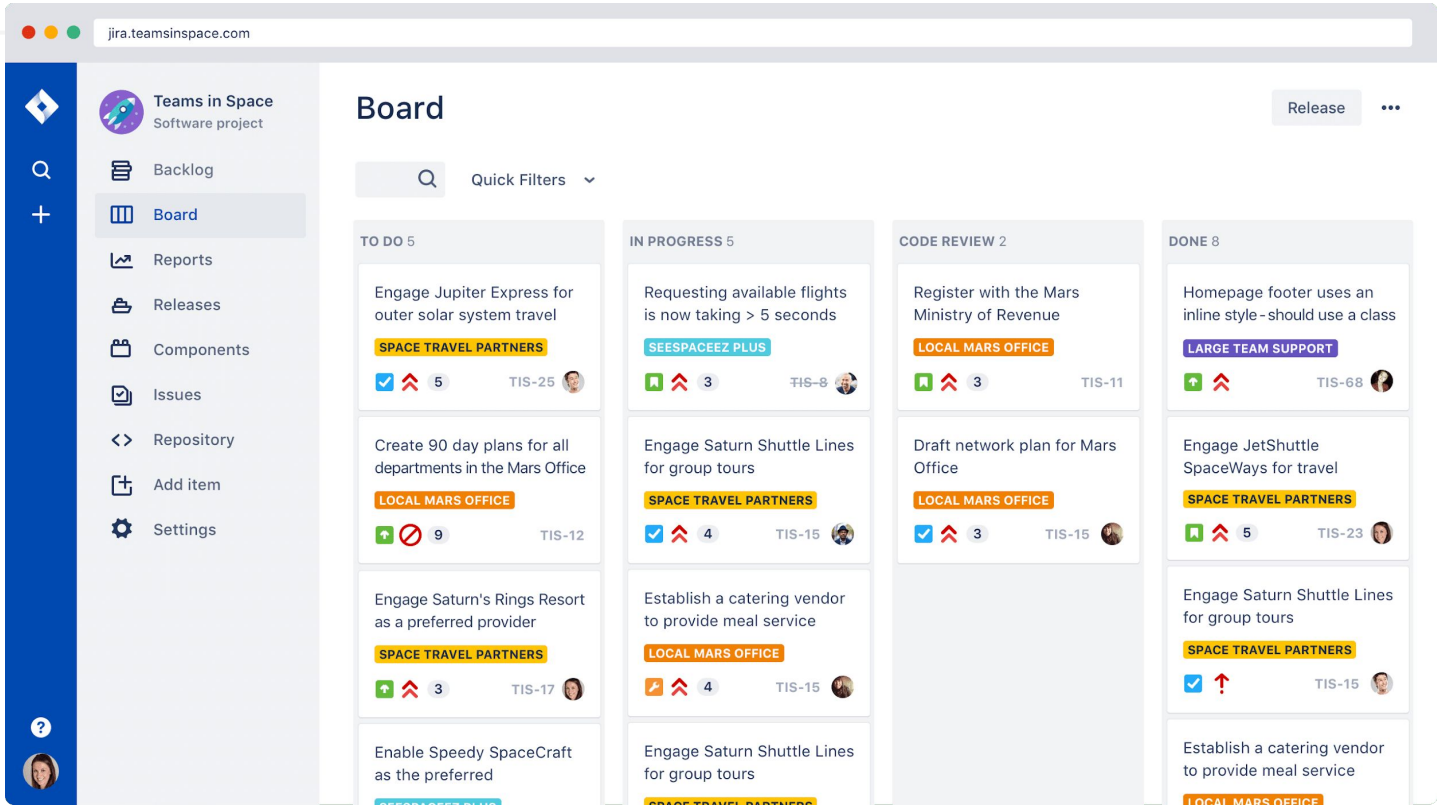
# What is JIRA?

## JIRA is …

- ✔ a web Atlassian tool
- ✔ the tracker for team planning
- ✔ configurable and extendable tool

## JIRA …

- ✔ adapts to business processes
- ✔ maintains your historic data
- ✔ used to prioritize, assign, track, discuss, report and watch issues\tasks

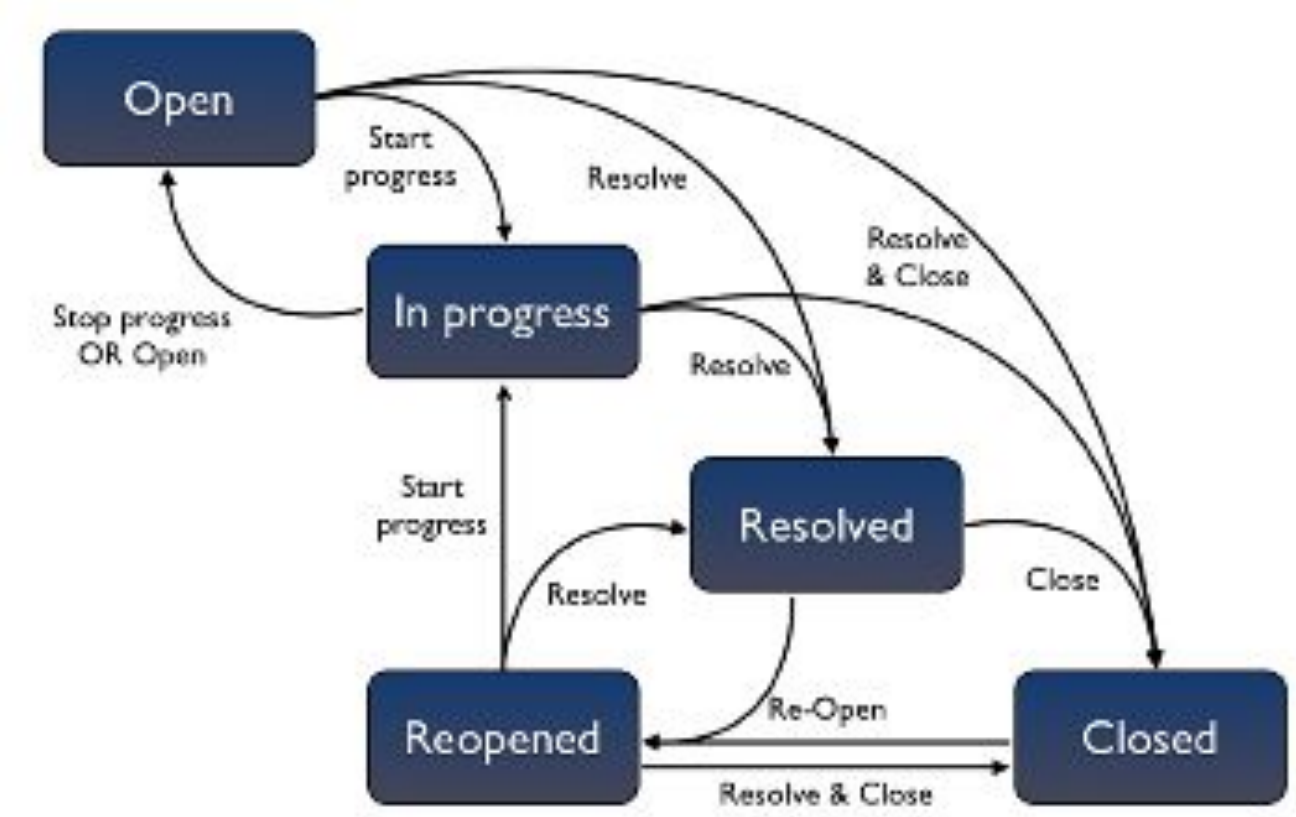


# JIRA advantages and disadvantages

| ADVANTAGES | DISADVANTAGES |
| --- | --- |
| • Teamwork facilitation<br>• Full visibility of the supported processes for the team and process/unit stakeholders<br>• Planning<br>• Time-reporting<br>• Tasks prioritizing<br>• Clear KPIs for People Management | • JIRA Performance<br>• Lack of training on project management<br>• Time and efforts spent on «Getting acquainted stage» |

## Workflow in JIRA?

A JIRA workflow is the set of statuses and transitions that an issue goes through during its lifecycle.



## What is an issue?

**Issue – any task that requires an action from a person.**

### Issue types:

**Bug** — a problem which impairs or prevents the functions of the product.

**Task** — a task that needs to be done.

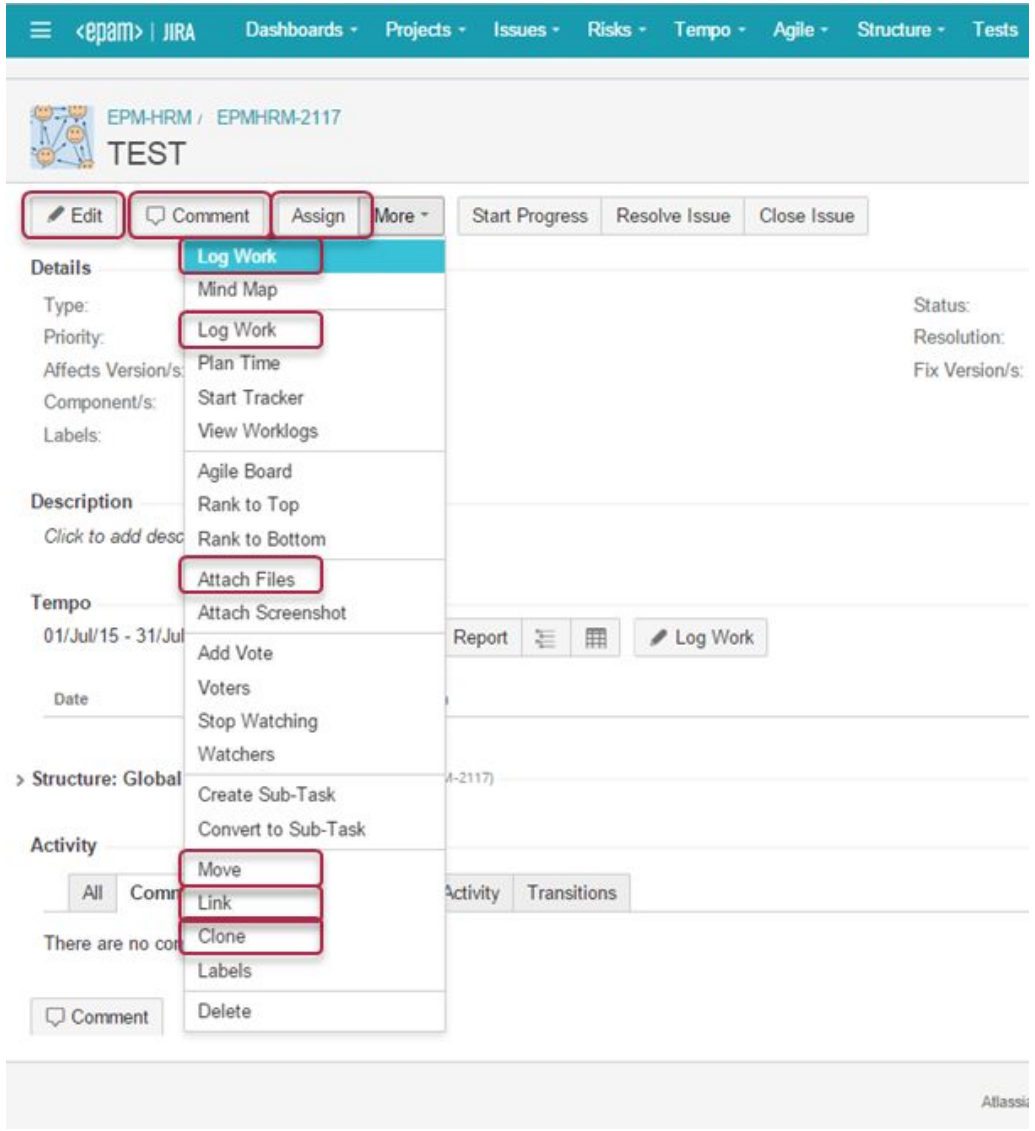
**Sub-task** - are useful for splitting up a parent issue into a number of smaller tasks that can be assigned and tracked separately.

**Story** – in JIRA Agile, a story is represented as an issue, and individual tasks within the story are represented as sub-task.

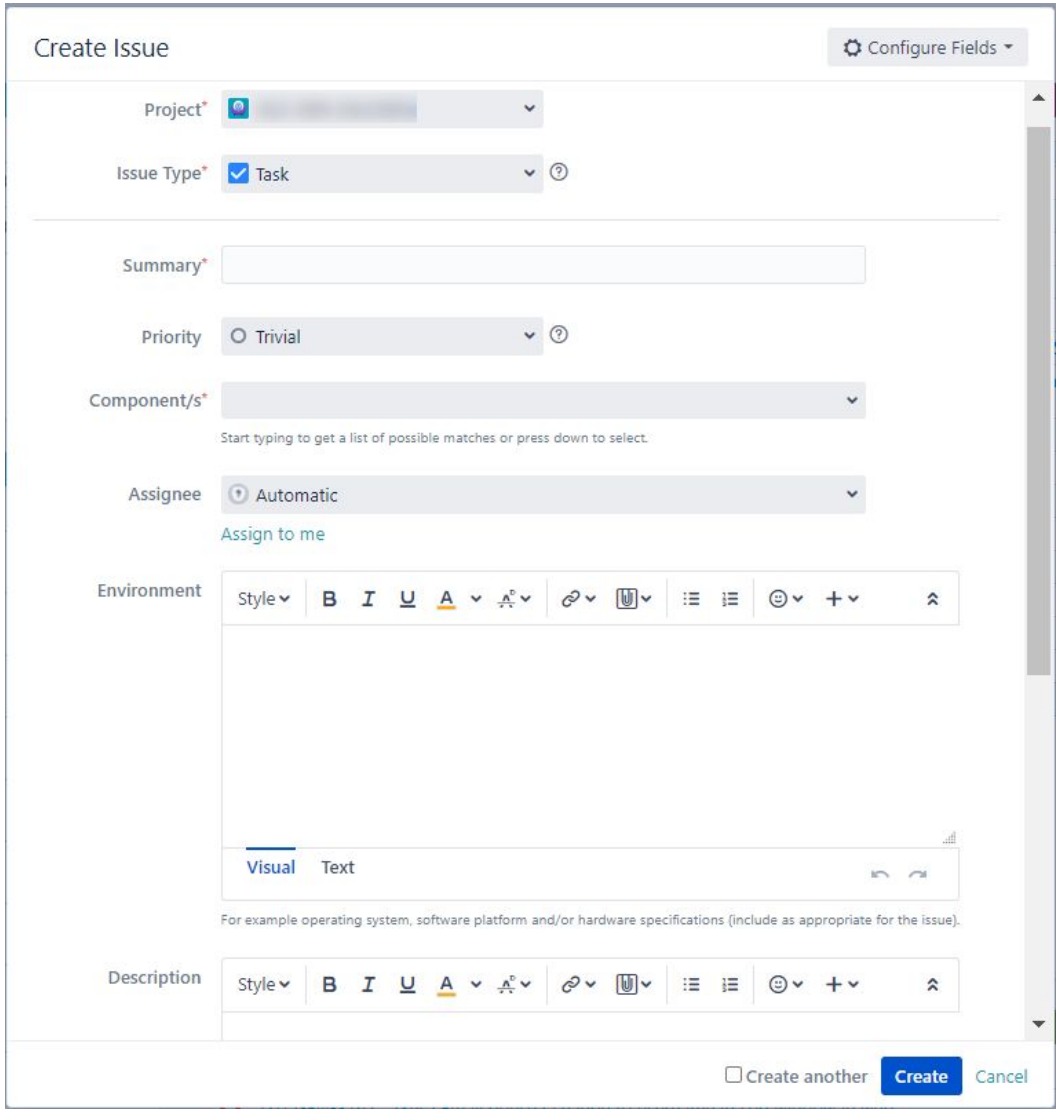
**Epic** - An epic captures a large body of work. It is essentially a large user story that can be broken down into a number of smaller stories. It may take several sprints to complete an epic.

# Standard actions on issue

- Create
- Edit
- Assign
- Comment
- Log work
- Attach
- Link
- Clone
- Delete

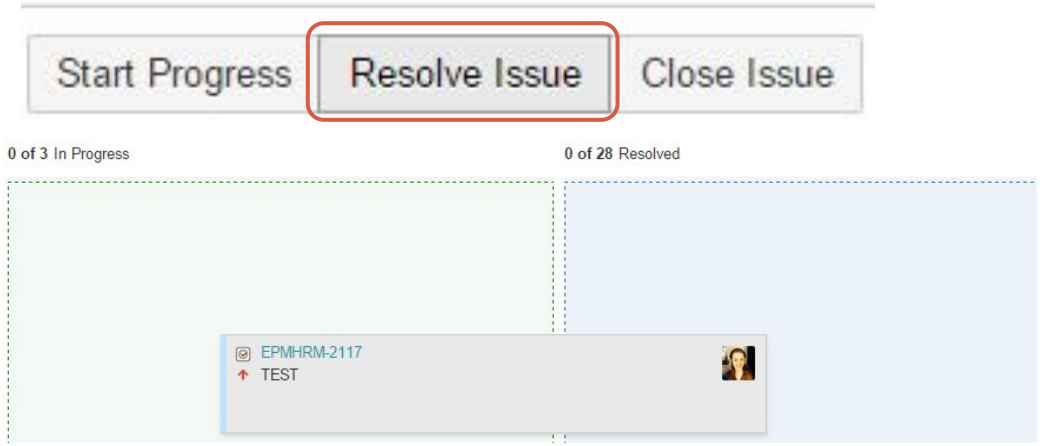


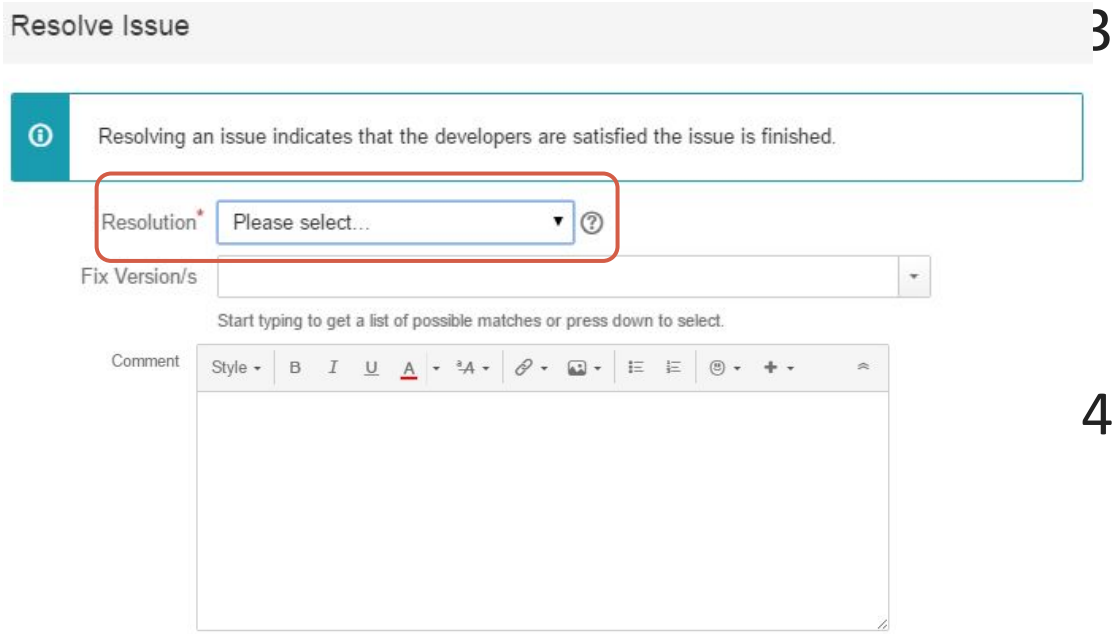
# Creating an issue




1. **Project.** Every issue is associated with a project.
2. **Issue Type.** This field allows you to select the kind of issue you want to create, such as a bug or task, etc.
3. **Summary.** A summary should be a very brief summation of the issue. The summary acts as the title of the issue.
4. **Priority.** You can set the priority of this issue. Setting the priority can help your team during the triage process.
5. **Assignee.** Here you can choose whom to assign the issue to.
6. **Environment.** OS/browser/product configuration.
7. **Description.** Here's where you can describe the issue in as much detail as you'd like.
8. **Create.** Once you click 'Create', your new issue will be added to JIRA.

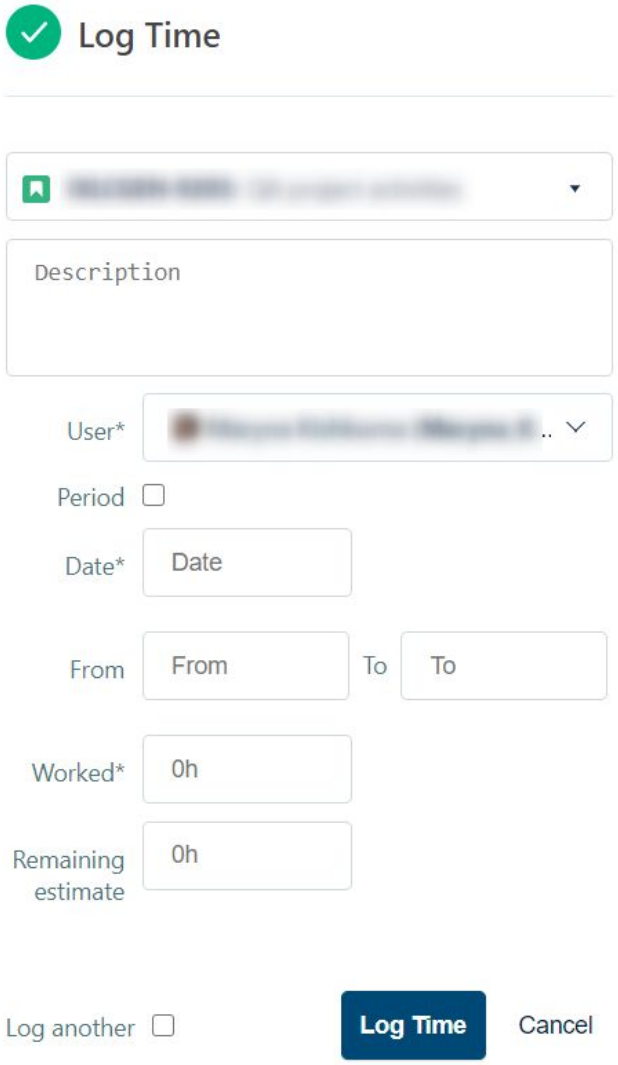
# Resolving an issue




1. Press **Resolve** button in the upper panel bar
2. Move the necessary Task from one column to another (Tracking board)
3. And don’t forget about Issue Resolution. JIRA by default will mark an issue to be Unresolved when the resolution field is not having any value on it.
4. Adding comments to an issue is a useful way to record additional detail about an issue and collaborate with team members.

# Log work on issue




## LOG WORK ON ISSUE

To log work, use the following format:

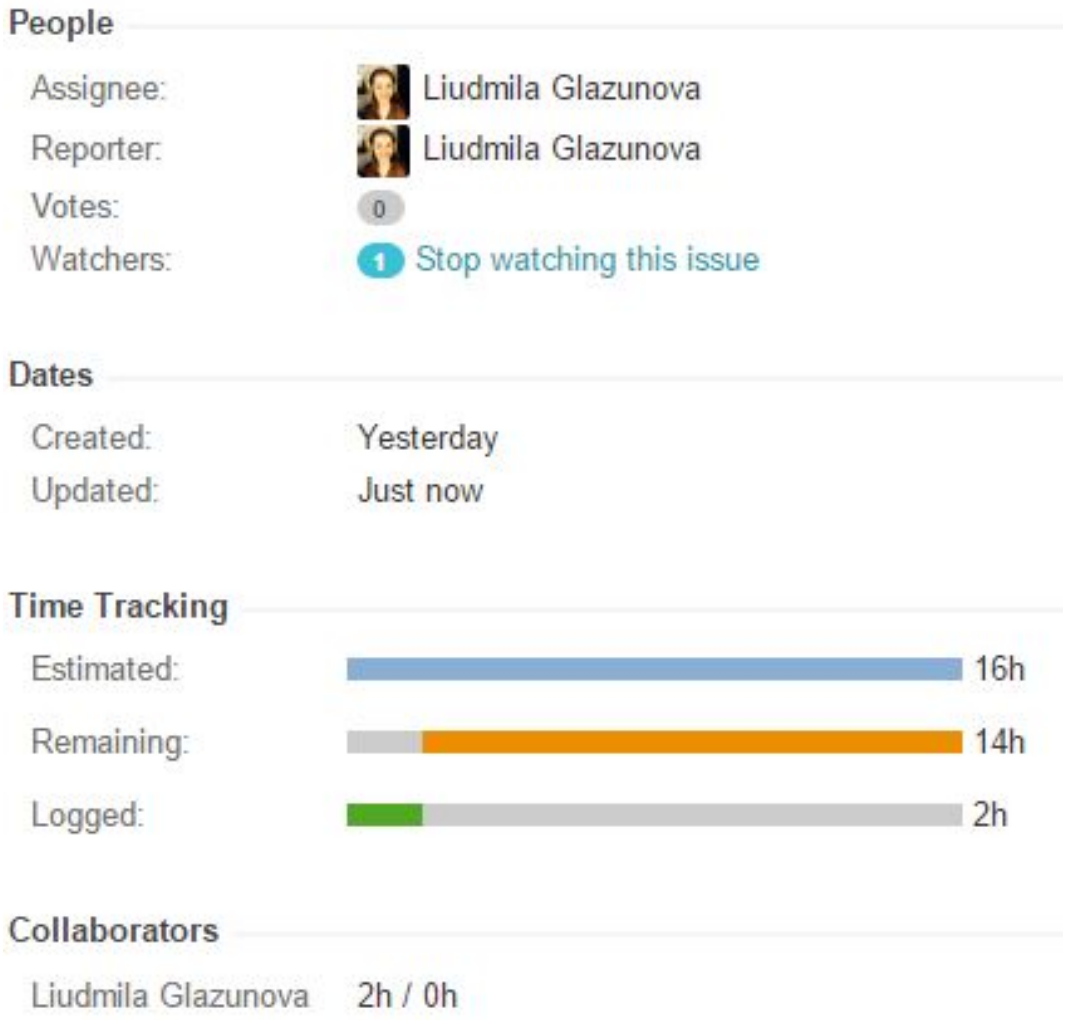
- **w** - weeks
- **d** - days
- **h** - hours
- **m** – minutes

## Time tracking

**Original Estimate** (blue) — the amount of time originally anticipated to resolve the issue.

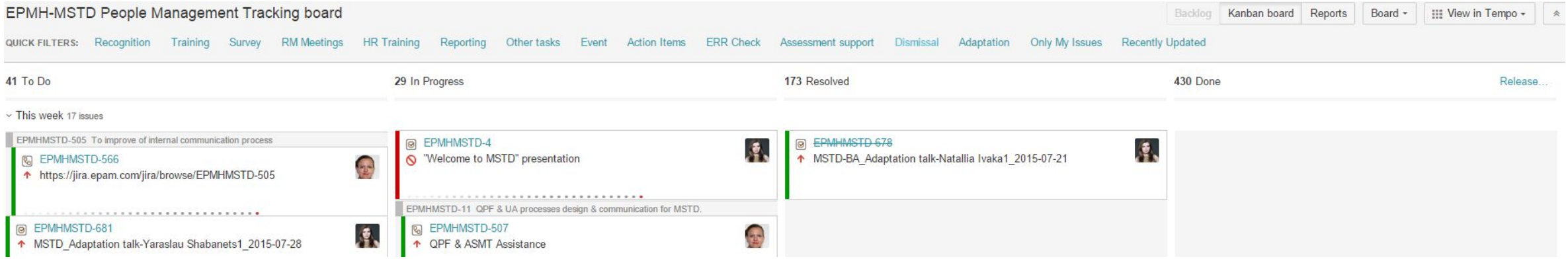
**Remaining Estimate** (orange) — the remaining amount of time currently anticipated to resolve the issue.

**Time Spent** (green) — the amount of time logged working on the issue so far.



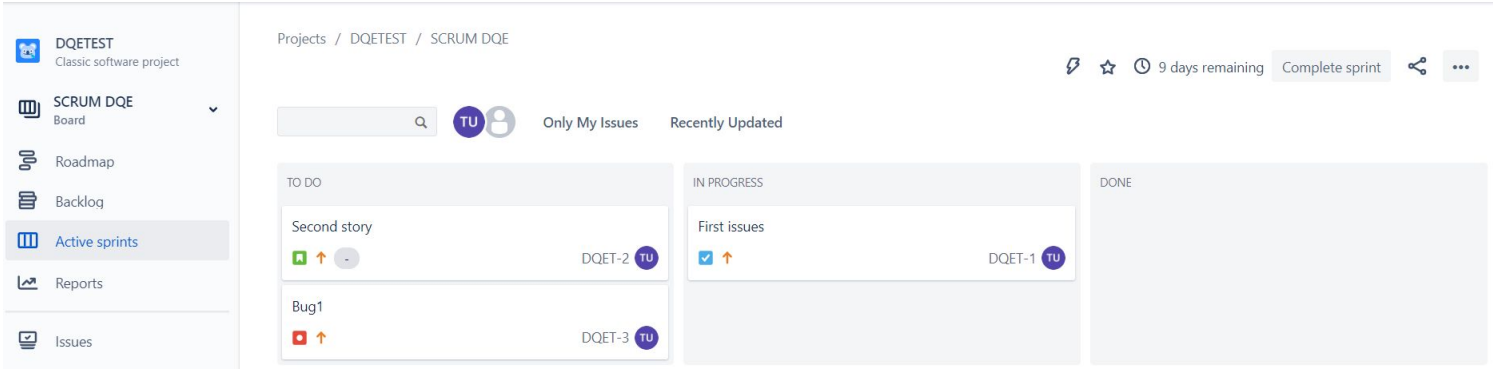
# KANBAN board

Kanban is a catalyst for change through small, incremental improvements to your existing process – be it scrum or otherwise. Rooted in lean manufacturing, Kanban is a signaling system that can be effectively applied to software development, DevOps, IT operations, HR processes and many other processes.

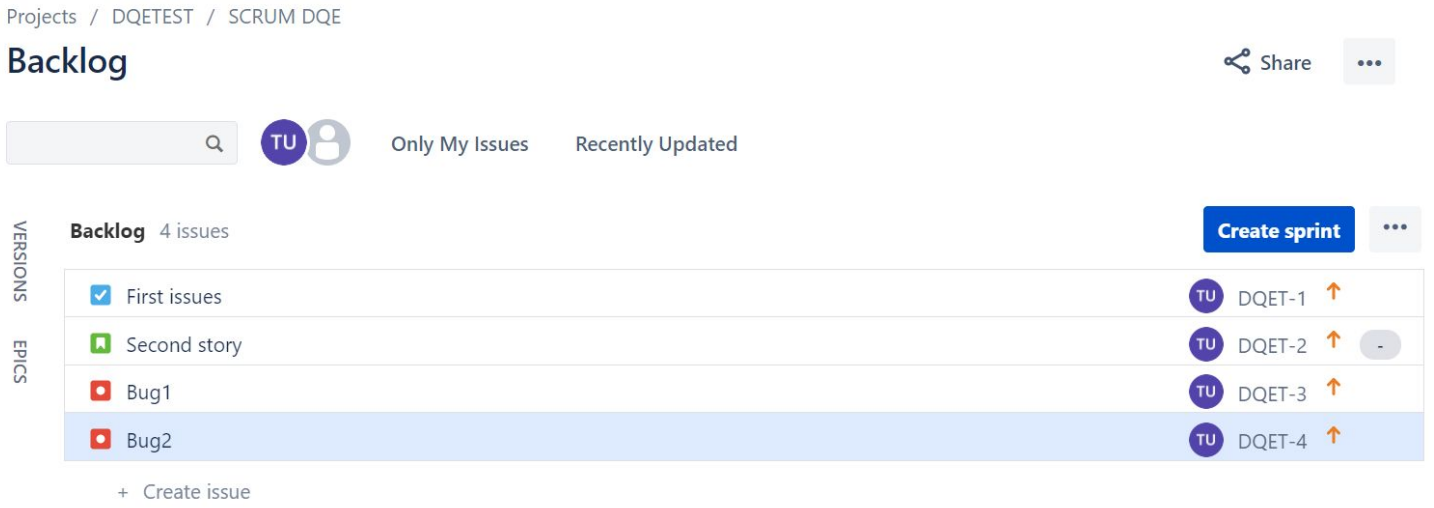


# SCRUM board

Teams who use scrum plan work in *sprints*, short blocks of time in which the team can estimate and return value back to the customer.
Software development teams use scrum to deliver complex solutions in a structured, predictable way.




The product owner manages the *backlog*, ensuring the team is working on the highest-priority items first. With JIRA Agile, the team can organize stories in sprints, in larger epics, and across release versions.



<epam>

# Q & A