# Issue Tracking System

### Overview

An issue tracking system is an application in which customers can submit issues against a project in a well-defined format in order to qualify the project, e.g. for bug fixing, improvement an existing feature or creating a new one.

The issue tracking systems are very popular in the modern software development world. They are used for a natural project management when a certain flow of submitting and assigning issues is defined and a process of people who should submit, review or update the issues at certain time.

For instance, people can see how many bugs the project has experienced so far. In which period of the history there were more bugs than ever? What is the average ratio of fixed bugs by developer at a time? How does the project evolve?

The well-defined process might say some rules:

* Don’t do any work outside the issue tracker
* Release only these tickets (issues) that are verified by a QA
* Only the Project Manager can assign tasks to developers
* And so forth

### Examples

In today’s market there are some issue tracking products enjoying big popularity:

* Atlassian JIRA - <https://www.atlassian.com/software/jira>
* JetBrains YouTrack - <https://www.jetbrains.com/youtrack/>

Take a look at them. Play with their cloud instances. It’s very important to understand the big picture of the Issue Tracking System in order to develop one.

### Task Overview

Design and implement with full technology stack an issue tacking system. The development phases of features described in this document will be separated in stages.

The basic overview of the project is:

When the user accesses the Issue Tracker for first time, it is a guest to the system. The guests can only register new users or login with existing ones. Thus the login form with a link to the register form is shown. The user consists of the following information: Full name, Username, Email and a Password.

Once the user logs in into the system, it is led to the dashboard. The dashboard consists of project cards where the project name is said, its Team Lead and how many open issues there are.

Once the user chooses a project, it is led to the project issue board. On the left pane the user can find information regarding the project. On the rest of the screen there is the issue board, with all chosen statuses for this project as column names and the issues by this status distributed in the columns. There might be issues with no status which could stay in a separate column. The user can drag and drop an issue from one status to another.

Clicking on an issue card leads to the issue information page, where the user can change some of the significant information regarding an issue: its title, body, assignee or even project and status. Creating an issue from the left pane leads to a form which prompts the user to provide the same information.

Still take a look in JIRA or YouTrack to understand better the description above.

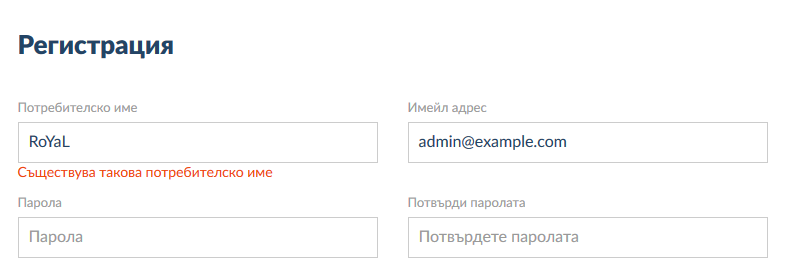
### Required Technologies

1. Java 8+ (at the time this document is written, the last official version is 8)
2. Spring Framework
   1. You may use Spring Boot in order to pack all the Spring Stack at once (Java EE, Application Server, Configurations and so forth)
3. MySQL
   1. Or equivalent database engine supporting relational tables and foreign key constraints
4. Spring Data
   1. Use the JPA Repositories from Spring Data
   2. Write custom interface methods when needed
   3. Do not query that data in memory (e.g. don’t do findAll().stream().filter(u -> u.getUserName().equals(“….”));
5. HTML5 & CSS3
   1. You may use [Bootstrap](https://bootswatch.com/) and apply a theme
6. JavaScript
   1. It’s highly recommended to use [jQuery](https://jquery.com/)

### Stage 1

Read carefully this stage and all others, because some functionality might change and you’d need to make the architecture extensible!

Setup a new project. Import Bootstrap and jQuery. Use header and footer separation. Implement users and roles, login form and registration form. When there is an error in the user data in the login or registration form, the relevant error message should appear under the field:



The users should login by username and password, but should register by the full information from the overview.

### Stage 2

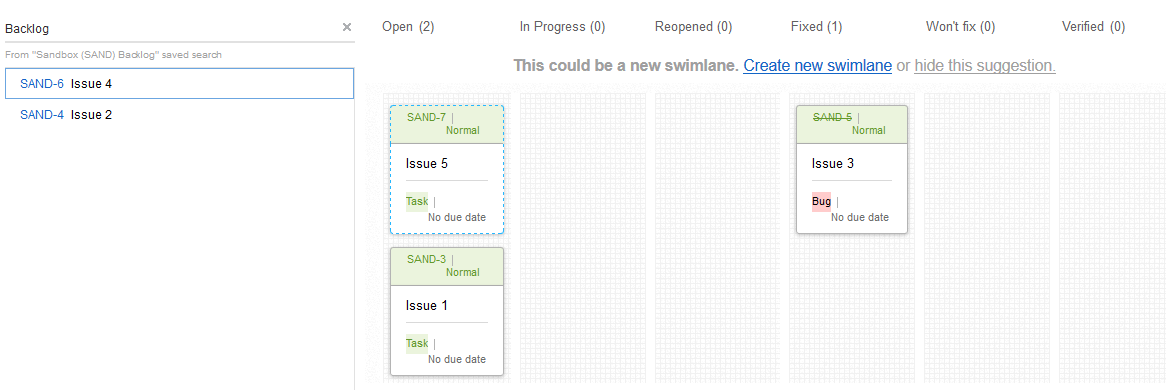
Design projects entities and core logic. Projects should be created by the administrator only. They have a team lead, which together with the administrator can edit the project at whole. No other key figure can edit the project settings.

Projects have also Name (which is unique per application), issues and allowed issue statuses. By default the application distributes with statuses “Open”, “In Progress” and “Fixed”. The administrator can add new statuses globally for the application and the team leads can choose whether they want to apply these statuses from the application to their projects, or they can create local statuses which are invisible for the other projects.

### Stage 3

Make projects work – the dashboard. On the dashboard from the left pane the user can see the Project Name, the Team Leads name and a button for creating new issue.

On the other side of the view, the user sees a column for issues with no status. And all other columns are columns for each status that the team lead has applied to the project. Under these columns there are issues that has this status



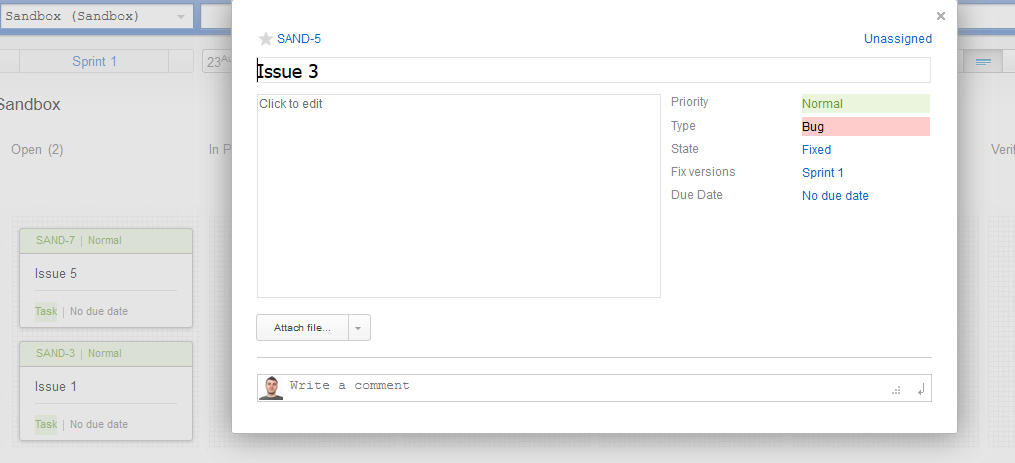
### Stage 4

Clicking on adding an issue button leads to a dialog (or new page, but better a dialog) which prompts the user to provide the Issue title, description, its type (application starts with “Bug”, “Improvement” and “Task”, but the administrator can add new types) and its assignee (a list of users in the application). The issue then goes with status “Open” in the current opened project.



### Stage 5

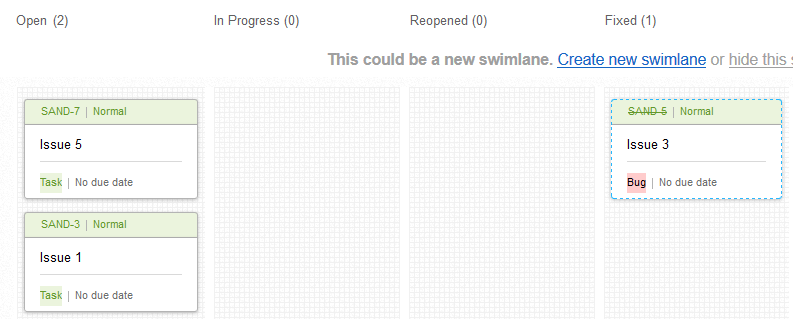
Clicking on existing issue from the board leads to the issue edit page. The user can change the status amongst the available in the project, the type, the assignee or even the project.



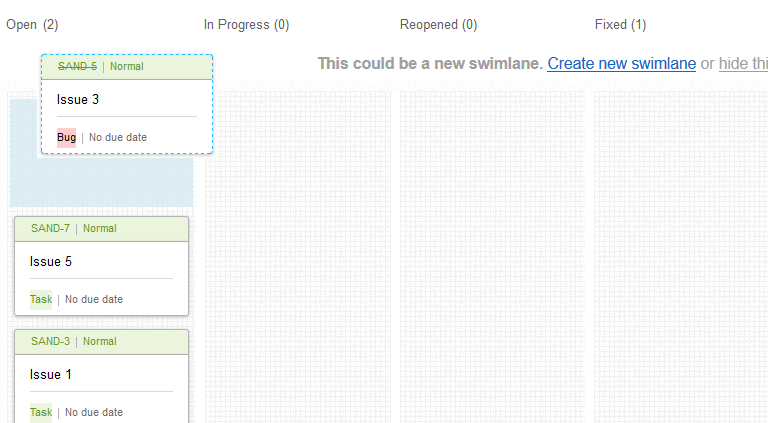
### Stage 6

Issue cards are useless if they cannot be dragged from one status to another. Implement drag-drop functionality, where the user can drag a certain card and drop it to any other status column, including the column No status (Backlog). If the user drops the card to a non-status area in the page, the card should automatically get back to its initial place. It’s possible the issue card to not fit in the current column height. Increase dynamically the column height when the user is about to drop it there:

For instance in the below screenshot there is no room for issues in the “Open” status column



But when the user is about to drop “Issue 3” there, the column increases its height:



### Stage 7

Implement status transitions. In the normal world there are required transitions in order to go from one place to another. For instance you can barely go from 3rd grade to 8th grade. You’d need to pass all the exams through 4th, 5th, 6th and 7th grade. That’s the way it is in the task processing too. You cannot directly “verify” an open task. It probably should stay some time in “In progress” while the developer is developing it. Then it should go to “Fixed” when the developer is done. Then you need to move it to “Testing” while you are testing it, and once you are done you can move it to “Verified”. That’s what transitions are. One issue can transit from one status only to defined statuses by the Project Team Lead.

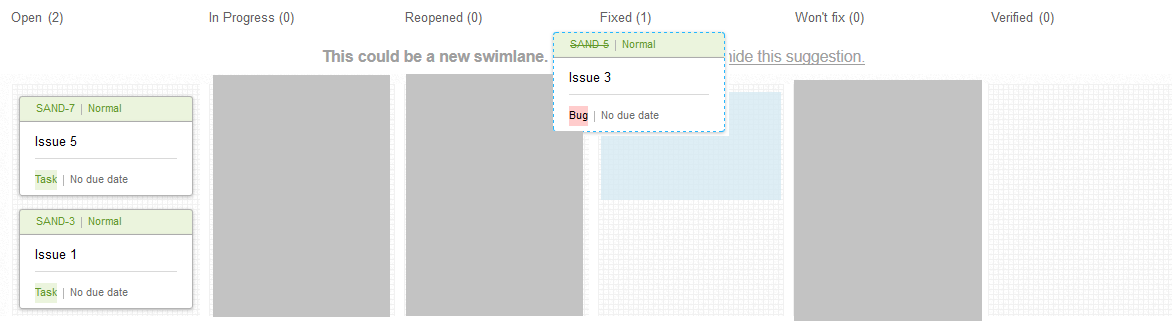
Make a menu where the Team Lead can define status transitions. One should pick a status and choose which statuses the issue could go from that status. For instance he might choose

From “Open” the issue only can go to “In Progress”.

From “In Progress” the issue can go either to “Open” back or to “Fixed” if it’s fixed.

From “Fixed” the issue cannot go to any other status. It’s its final destination.

This means that when you try to edit an issue, the dropdown shows only statuses available from the current one. When you try to drag an issue, all columns that it cannot go – goes blurry and its practically impossible to drop it to a status that cannot transit from this one.



In the above screenshot it’s shown that the user tried to drag “Issue 3” from status fixed and columns “In Progress”, “Reopened” and “Won’t Fix” came blurry. So he is allowed to drop it only in “Open”, “Verified” or stay in “Fixed”.

### Stage 8

Implement project teams. The administrator or project team lead can add or remove users that are in the project team group. This means that you only can assign issues to people which are in the project team group.

Once you try to remove an user from the project team, and it has non-closed issues assigned, you should be prompt either to click “close them all” and they automatically reach the terminal status for the project or “go to the list of issues” where the list of these issues is opened and you manually to something with them.

### Stage 9

Implement project permissions. The administrator or the project team can lead can add permissions for each page in the project and operation. Then give that permission to certain user from the project team group. For instance certain users might be able to add new Statuses in the project, despite they are not team leads, nor admins.

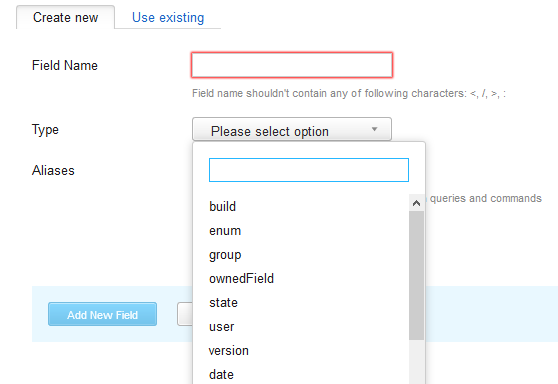
### Stage 10

Implement dynamic roles. Fetch all pages and operations upon pages dynamically (e.g. by reflection) and list them to the administrator. Then he chooses one or more of these permissions and creates a role with them. The roles and relative permissions are stored in the database and can be assigned to any user.

Stage 9 and 10 mean that there should be an user profile menu where the administrator can change all the things including roles and permissions.

### Stage 11

Implement dynamic project columns. The administrator or the project team lead can add columns to the project that later will be shown in the add/edit/show issue menu. He chooses the name of the column (field) and its type – a string type, an integer type, a date-time type or a dropdown with predefined values of one type. If its string type, users can add anything to that field. If it is integer – only integers can be input. If it is a date-time – a calendar is shown. And if it is a dropdown with predefined values – the dropdown with the values is shown.



### Stage 12

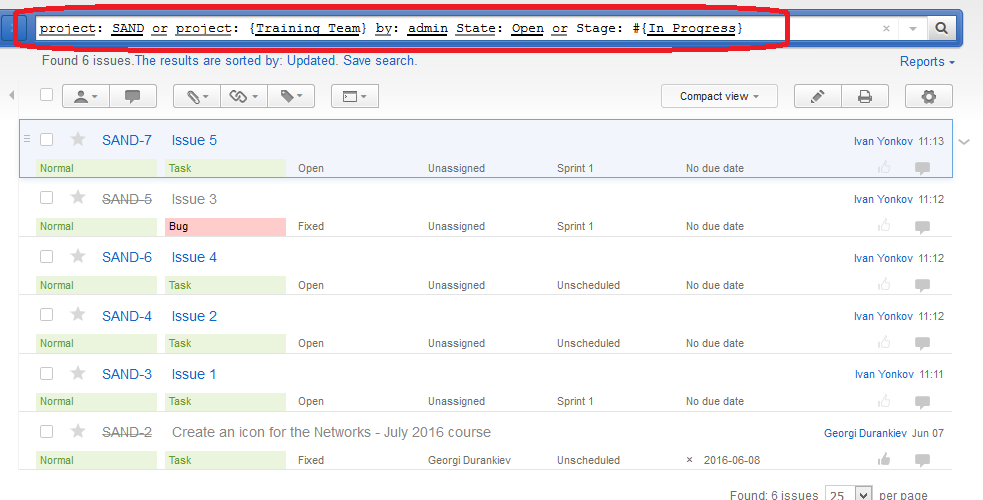
Implement searching for an issue by one or more criteria. You may fetch all predefined and dynamically added fields and give them to the user to populate them with values. After which you need to find all issues matching the criteria of the populated fields and list them to the user

### Stage 13

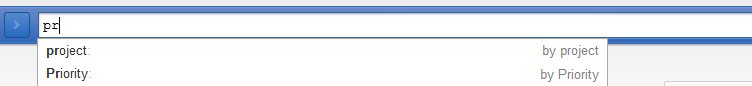
Implement searching filters. Filters are saved searches. For instance after the task from Stage 12, the user can click “Save search”, provide a name and then in a separate menu has a list of these names. Clicking on some of the names leads to performing the same search again.

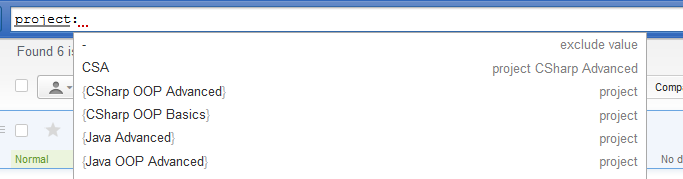
### Stage 14

Implement searching query language. The search from Stage 12 is powerful, but not so much. What if I want to search for assignee “Pesho” OR “Gosho”?



As you might see there is a custom query here. It also gives suggestions for either Keywords and Values:





### Stage 15

Implement mail sending. As any user has provided an email in its registration process, we can send emails to all of them when certain event happens.

Send emails with relevant titles and texts for these events:

1. Issue has been added in a project – to the one that is assigned to it. If it’s unassigned – to the whole project team group
2. Issue has been updated – to the old and new assignee. If issue was or goes unassigned – to the whole project team group
   1. If you have implemented a comments to each issue – the comments text should be send by email too
3. The team lead should receive emails for each event

### Stage 16

Make email sending events customizable. List each updatable operation to the administrator so he can choose which of them to be send by email and its text. Allow the administrator to use placeholders in text which will be automatically populated with dynamical data e.g. administrator says {text} and its populated with the comment text, if he has chosen the “Issue commented” event.

The project team lead can also do that, but on project level.

**Feel free to take any ideas for design or/and functionality from the existing issue trackers in the market (the given ones JIRA and YouTrack, but you can research for many others). The screenshots from this document are taken from JetBrains YouTrack.**