Software Development Assignment 4

Project Management Tool Usage Report

*Weiran Yuan, Jianhui Lu, Li Chen*

*RIN: 661677444，661752675，661693723*

In this assignment we need to demonstrate our use of project management tools and version control system. Our group use both Redmine and Icescrum as the project management system to fulfill different functionalities and satisfy all the requirements in the assignment. And we use Github as our version control system. Actually, Github is evolving and now can do pretty much most of the jobs project management tools can do, such as work flow, tasks assign, etc. Using its extensions, more magical things are able to do.

# Version Control System -- Git

The version control system is Github, it’s a distributed control system. After we download the Git app on our computer, we are able to git push and pull our project files to or from the Git remote repository. First thing first, we created a project on git website, as you can see in the followed picture.

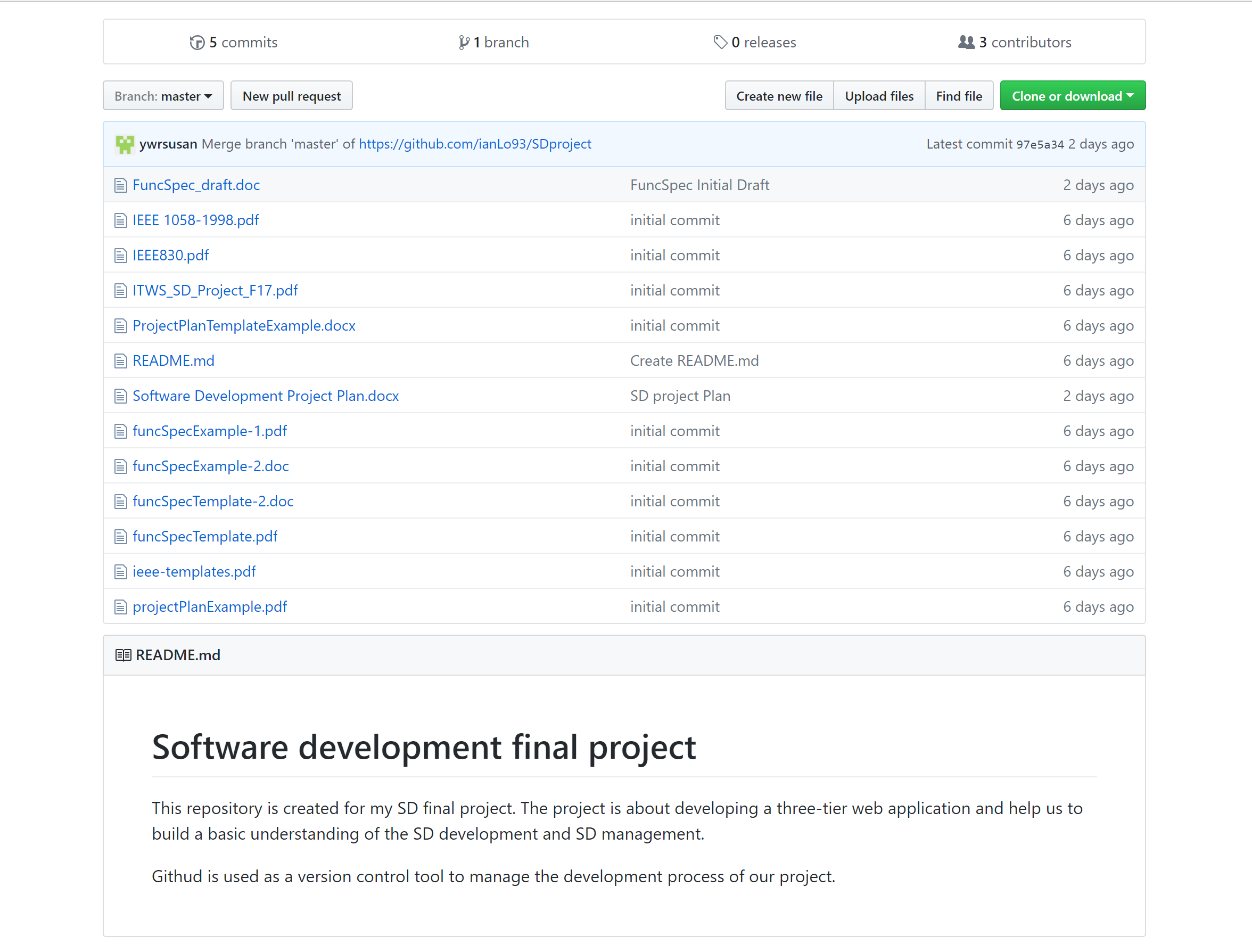


Fig. Github Main Page

The project did not have any files in it when it’s created, to push all these relevant files to git, we need to create a local repository. Use command window and locate on the project file on the laptop, we create, config username, user email and the core editor, use

* git init
* git add .

command to initialize and add all the content to the repository, then

* git commit -m ‘first commit’

We can see a ‘.git’ file lie right there in your local file, this is the local repository

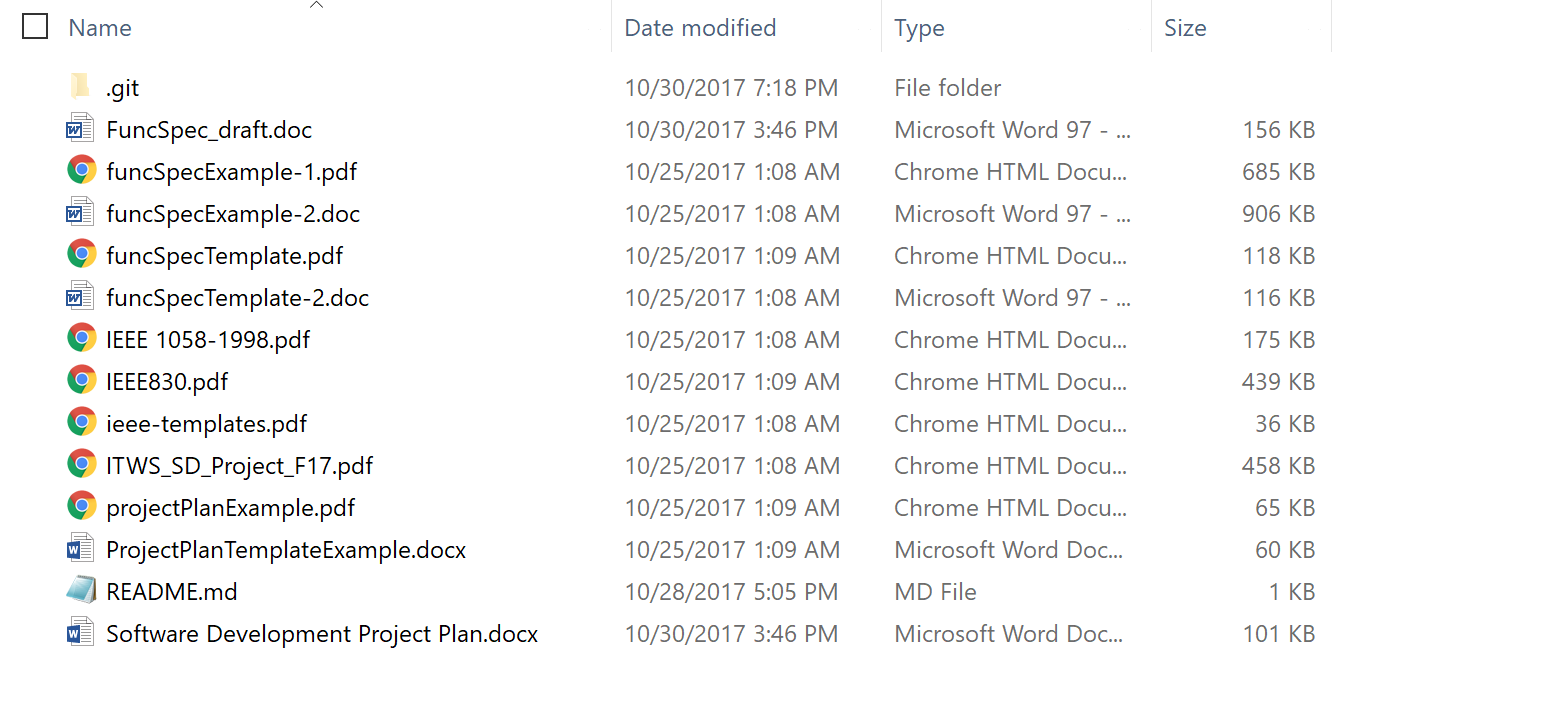


Fig. Local Repository

Using the command

* git remote add SDproject <https://github.com/ianLo93/SDproject>

We create a remote repository named “SDproject”. Now we can push and pull using the following command line

* git push -u SDproject master
* git pull SDproject master

Using these two commands we are able to update remote and local repository. And we also create a new branch using the command line

* git branch extra\_bran
* git checkout extra\_bran

The second command line change our branch to the new branch named extra\_bran, as you can see in the following picture, we now have two branches, one is master and the other is extra\_bran.

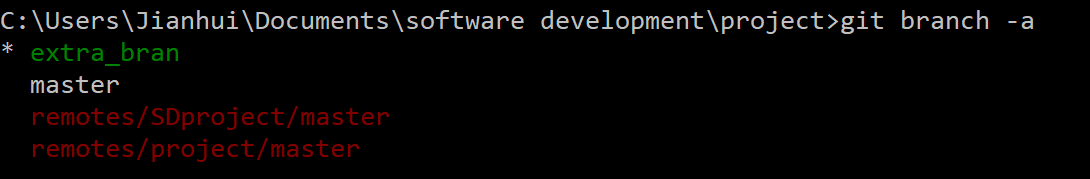


Fig. Git Branches

The green color with the star mark in front of the extra\_bran indicate that the branch status is extra\_bran, which means the I am now in extra\_bran branch.

# Project Management

The project management tools that we use are Redmine and Icescrum to fulfill different requirements such as Gantt chart and Burndown Chart. Combining these two management tools, we can have more convenience on managing our project.

We create a project named SDproject both on Icescrum and Redmine, we will post some examples. Basically our project main page is as the following capture.

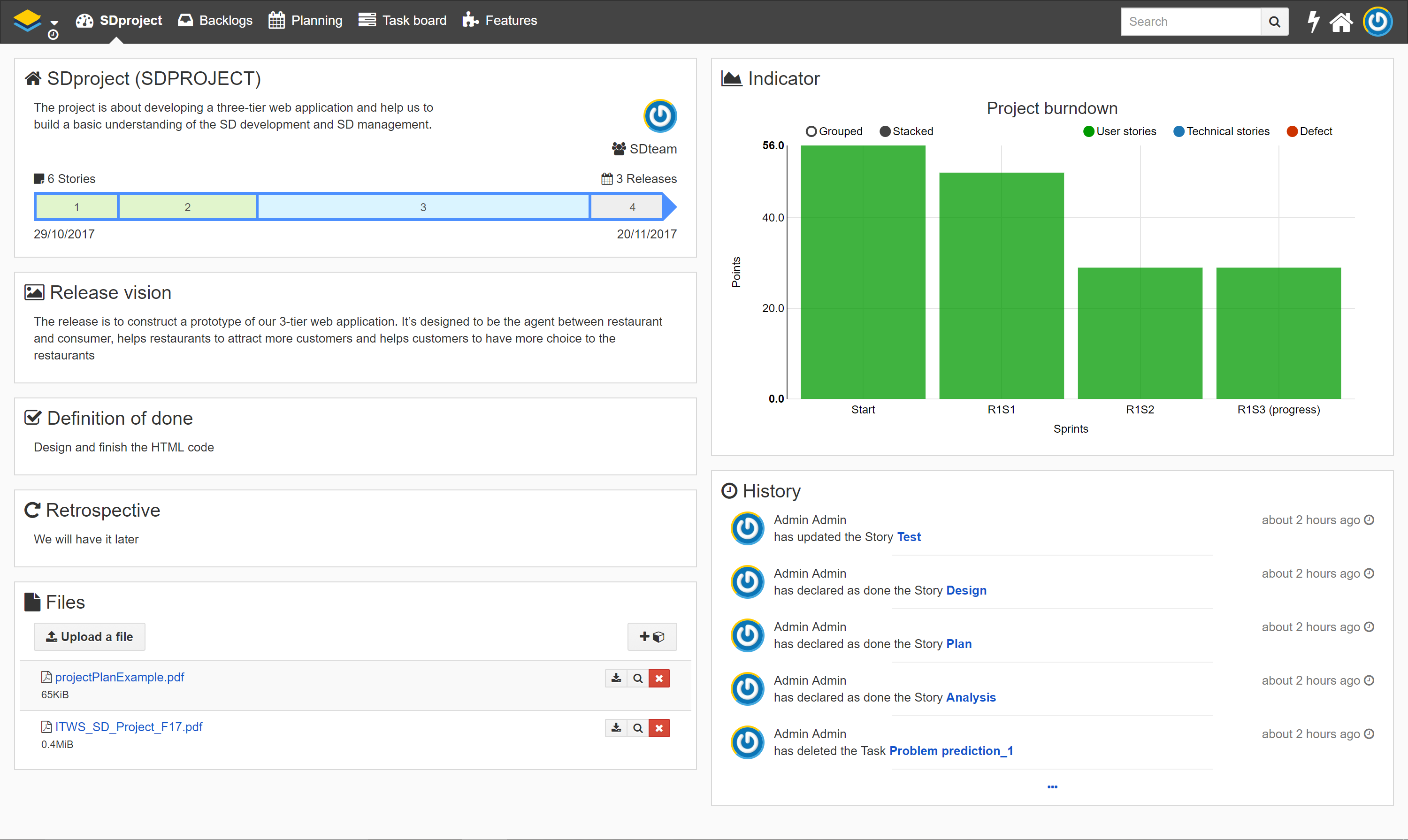


Fig. IceScrum Main Page

The layout is neat, most of the information are posted on the main page, such as history, the current release, current sprint and the introduction including files to the project.

When we try to create an issue on Redmine, we will have the following interface.

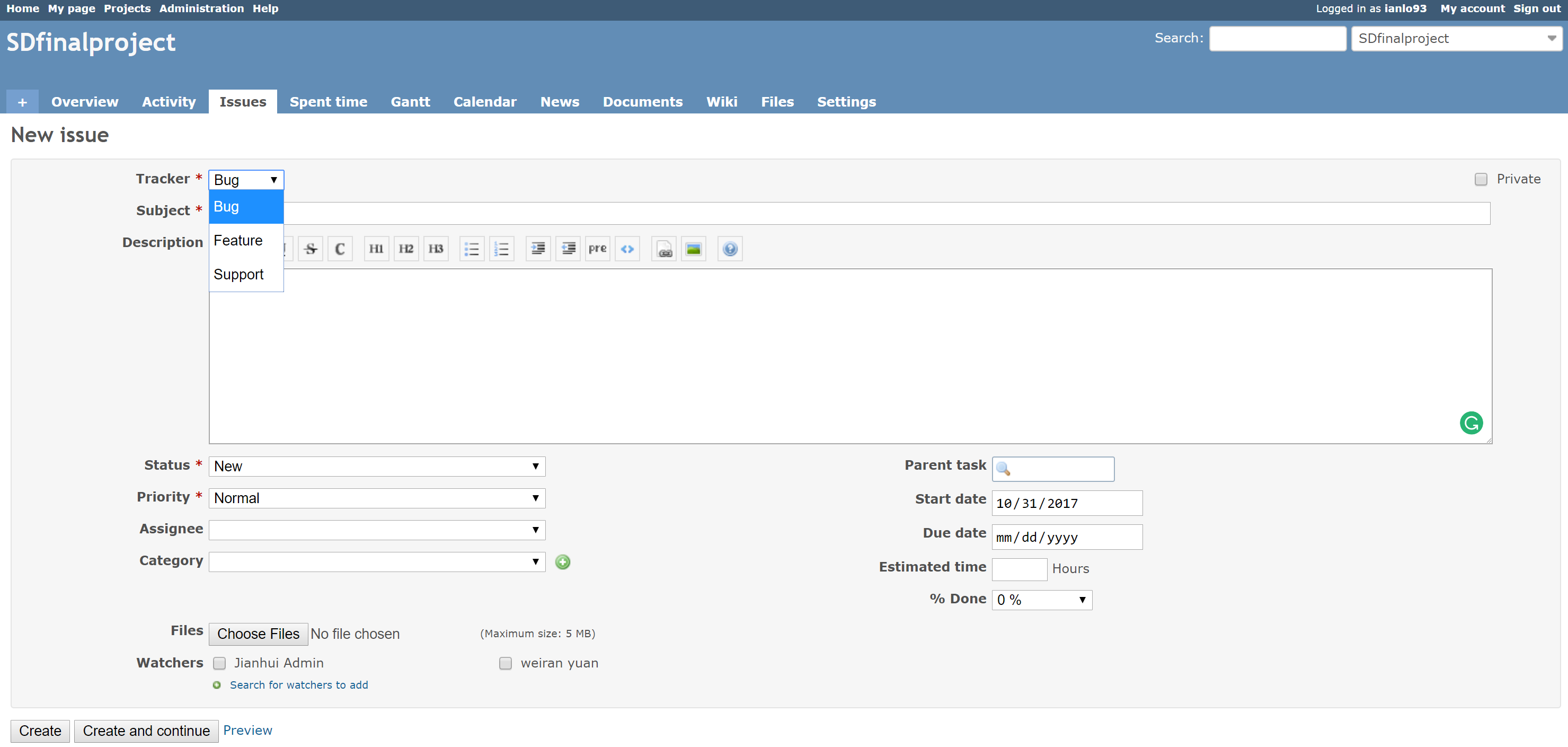


Fig. Issue Create Interface

This will allocate the issue to a **task category** that we created in the setting, hear I created 5 categories for our project. And we can also **assign our issue** to one of our members by pull down the assignee list.

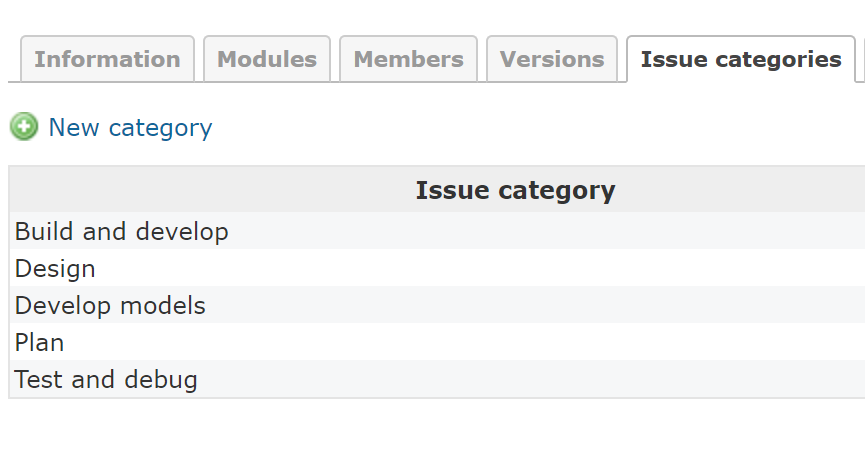


Fig. Issue Categories

And we can also give **task dependency** to the issue like setting its parent issues. The three members of our team are all added to the project both on the Redmine and Icescum.



Fig. Project Members

The milestone of our project can be defined on the Github, where there is a milestone setting explicitly. In our project management tool, we define the **milestone** in Icecrum as three releases:

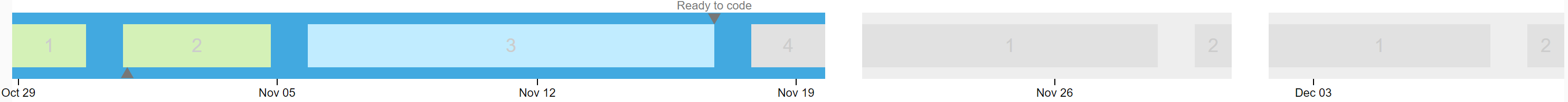


Fig. Project Milestones (Releases)

When we click the planning bottom on the navigator, we will see the three releases as the above picture, we defined three releases as milestone and assign some sprints, known as the process of a span of work in some extend, in between the milestone. In each sprint, we also created tasks and assign them as we described on the issues part.

Gantt chart can be seen on the Redmine, it’s base on the issues that we created before. In the Gantt chart, we can see the issues time span of our project and their current status. It’s a pretty convenient tool to manage our project. And as you can see beside the chart, some issue dependencies are also shown there.

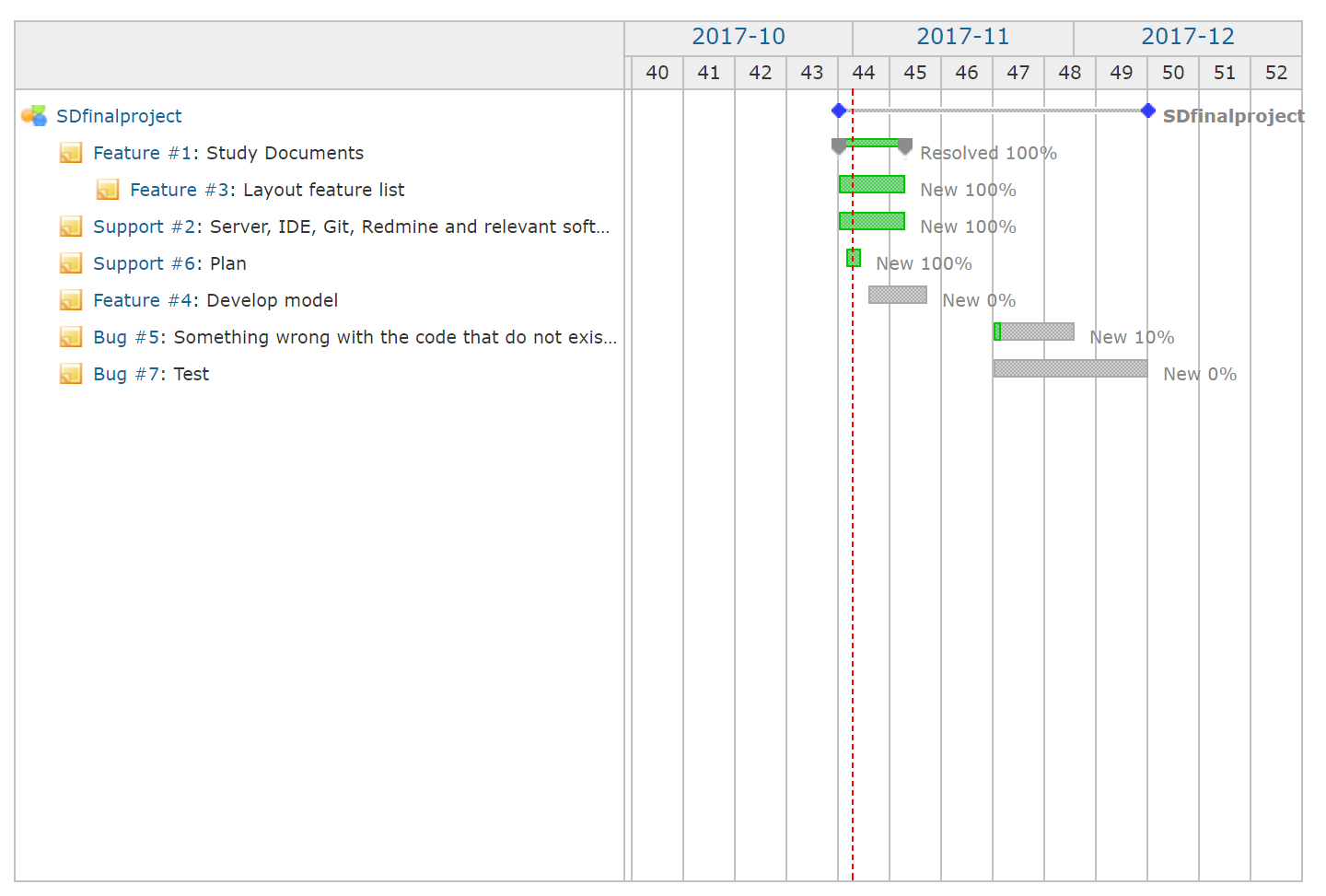


Fig. Gantt Chart

Burndown chart serves a similar functionality as the Gantt chart, we created it on the Icescrum as you can see below.

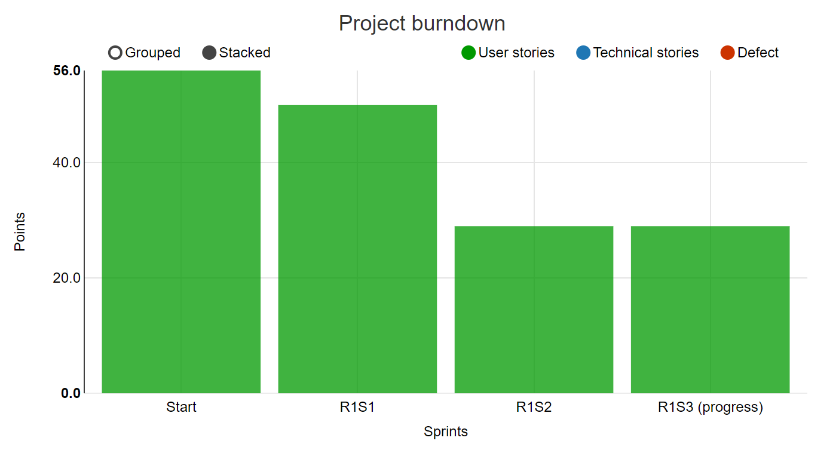


Fig. Burndown Chart

The burnup chart.

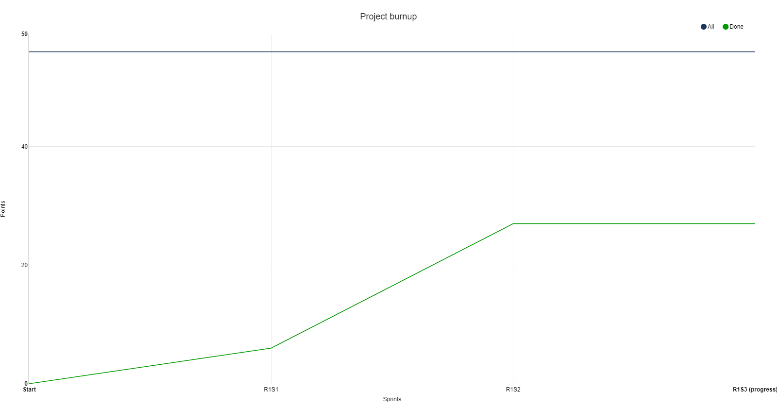
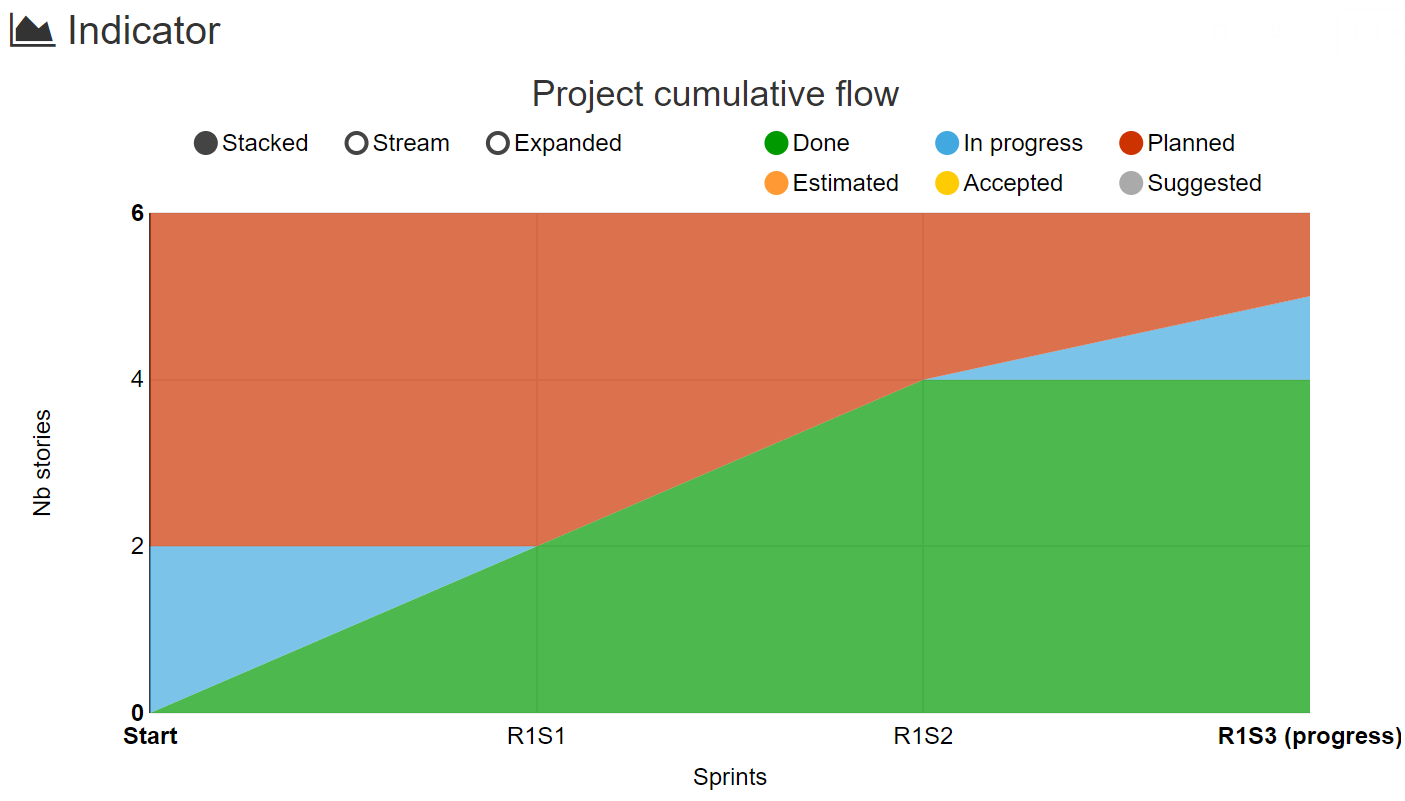


Fig. Burnup Chart

Project cumulative flow:



We created a control task flow in the IcsScrum, we designed the whole task flow by adding the tasks in backlog to each sprint, once we activate a sprint, the task flow of the current sprint shows up and tells you what tasks are urgent, what tasks are in progress and what we need to do in the sprint and how we can finish the sprint. Basically, we designed all the work flow for the whole project corresponds to the progress in our plan. Now we are in the third sprint of the first release, so we are now ready to code for our delivery web app.

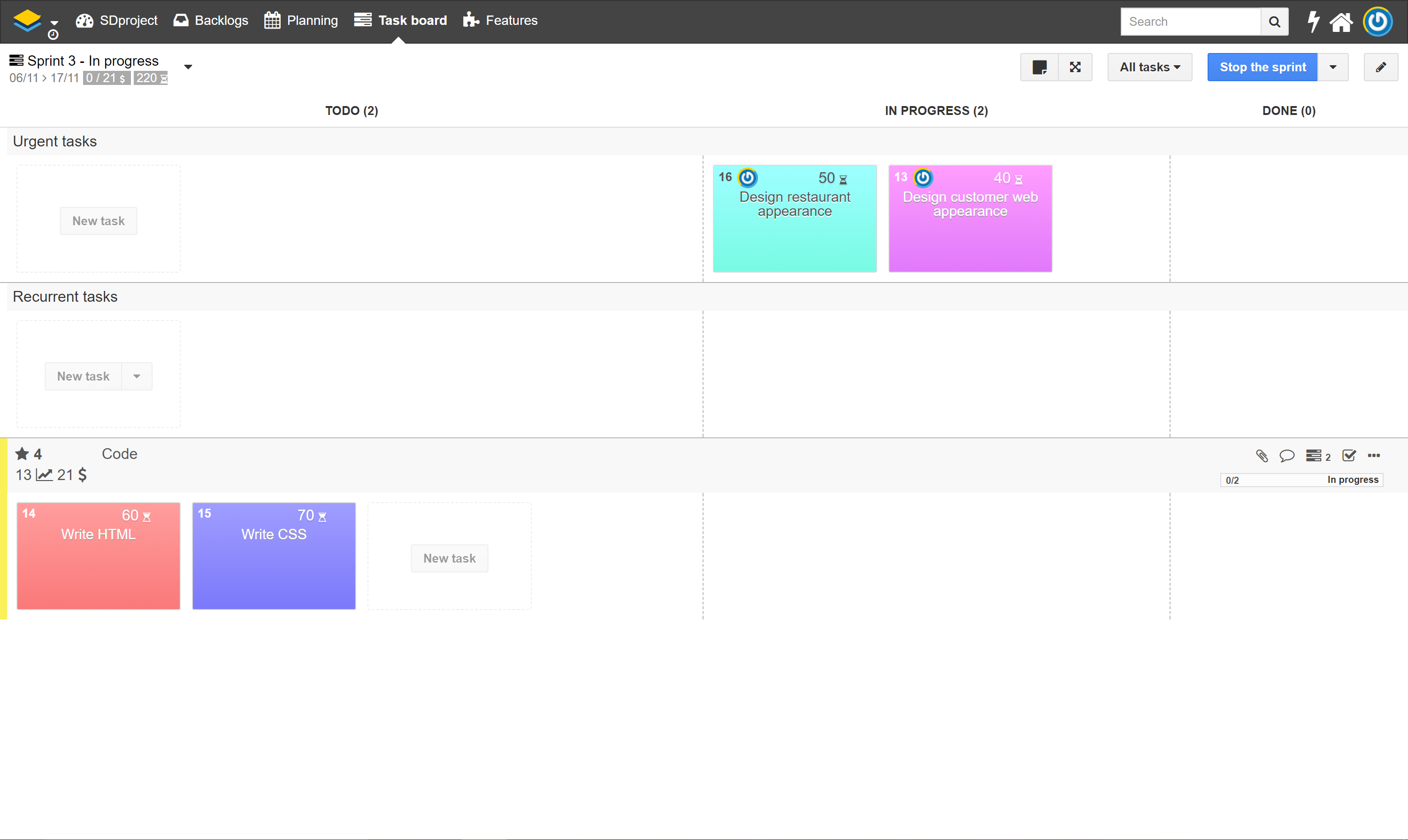
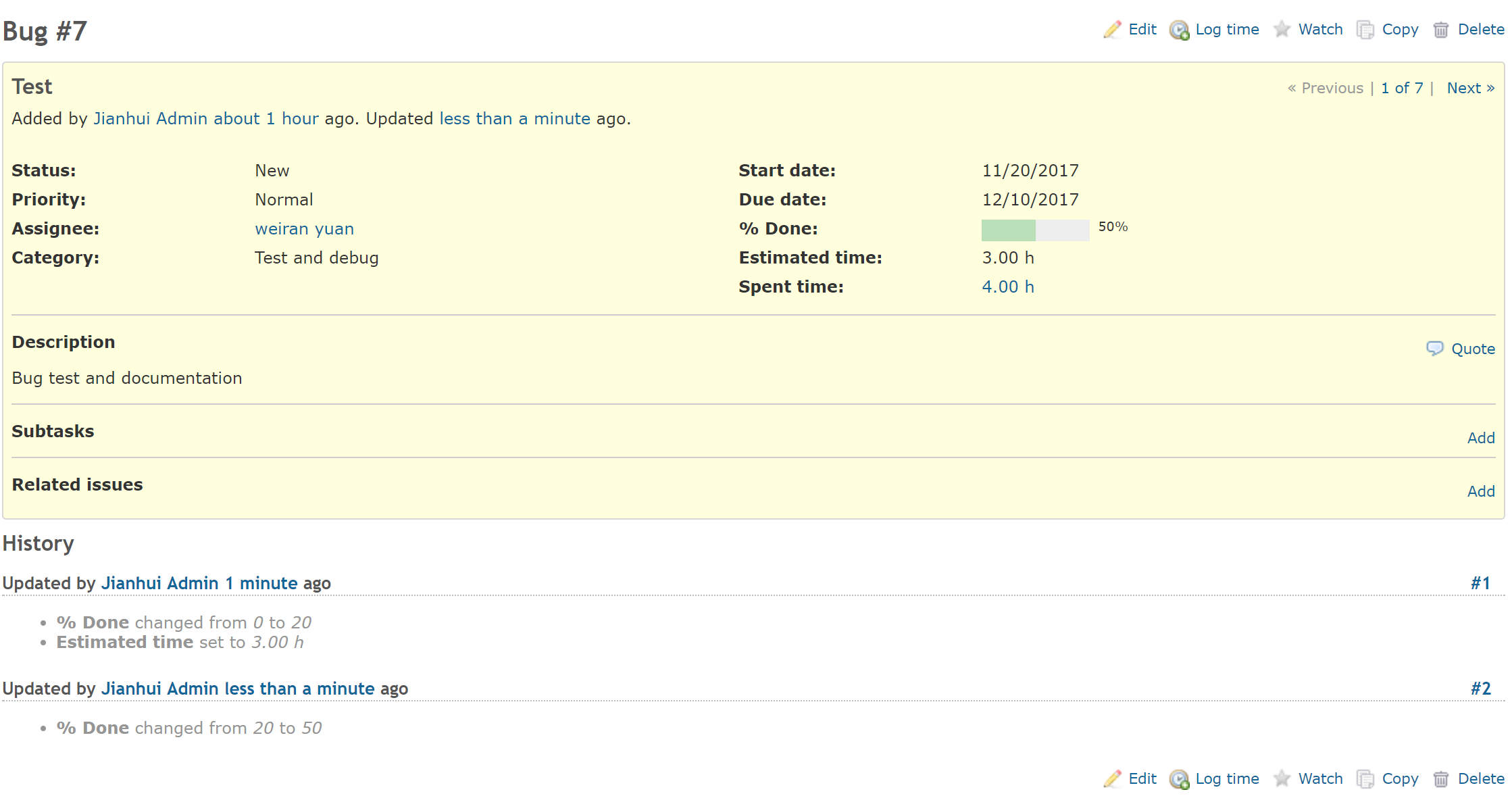


Fig. Control Task Work Flow

Redmine is a professional tool for tracking bugs, connecting it to the icescrum makes it perfect for the management of our project both on the project progress control and the bug tracking. The **bug trackers** are the categories that we assign to the issue, such as bug, feature and support. Each time we open Redmine or icescrum we are able to see issues and bugs need to be fixed. And each time we do some jobs to the bug and update the status of the bug, we will see the percentage of issue solved. And the **bug log**, or history, is right behind the status description of the bug so that we can **track the bugs**.



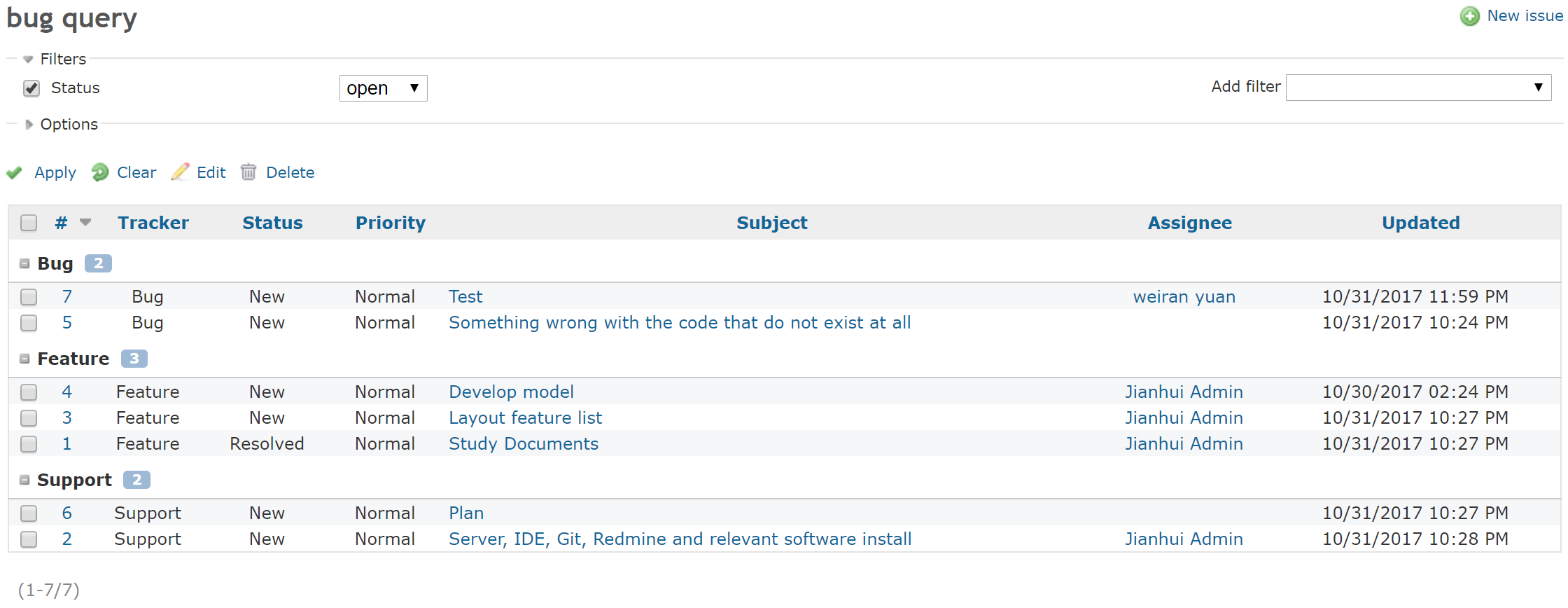


Fig. Bug Log and Tracking

# Connection of Redmine and Icescrum

In order to connect Redmine and Icescrum, we need to install an Redmine extension on Icescrum as followed.

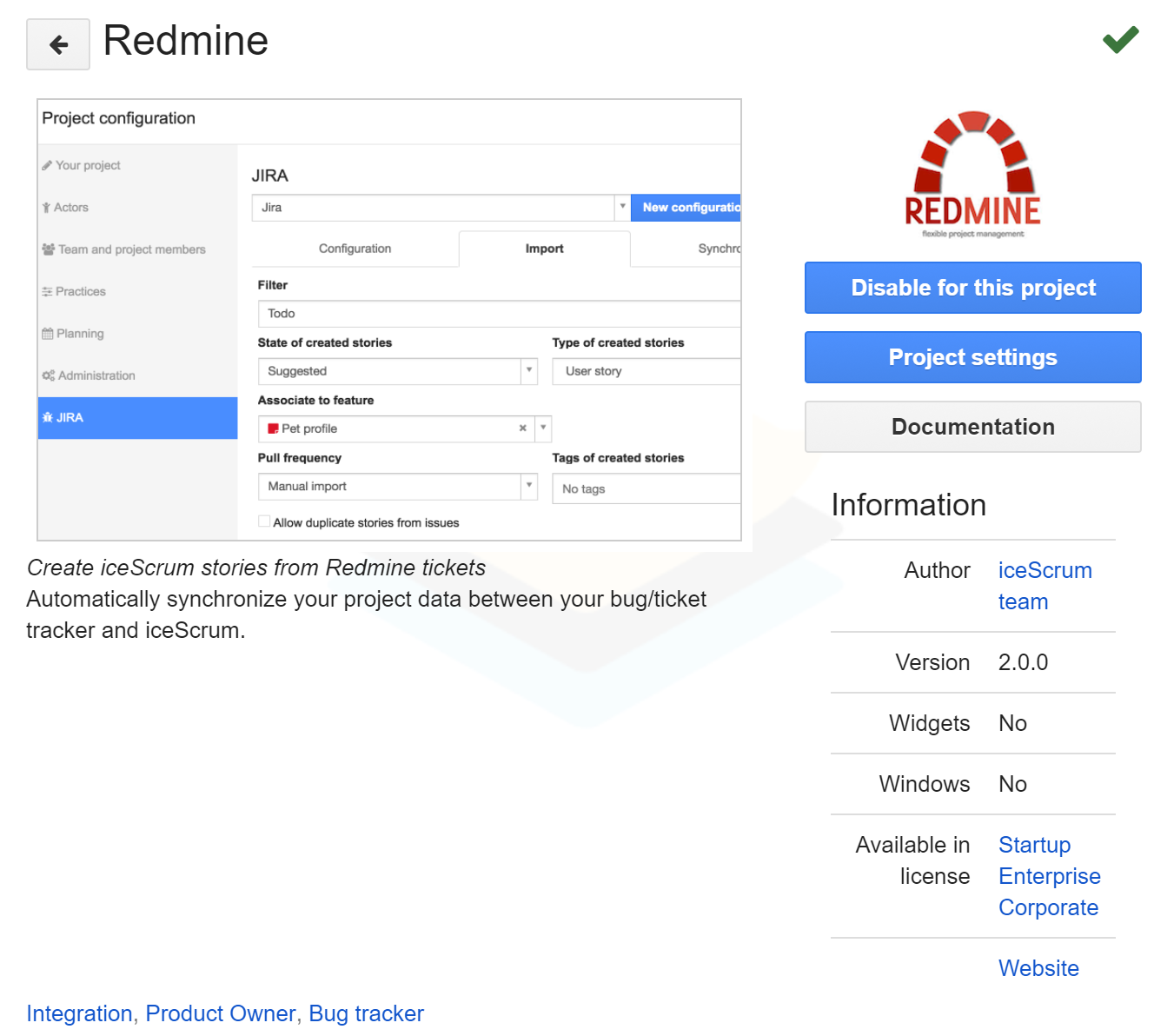


Fig. Redmine Add-on

It helps us to create icescrum stories from Redmine tickets. In project settings, we input the URL of our Redmine and log in as an administrator, then we can update both the Redmine and IceScrum by importing query from the Redmine.

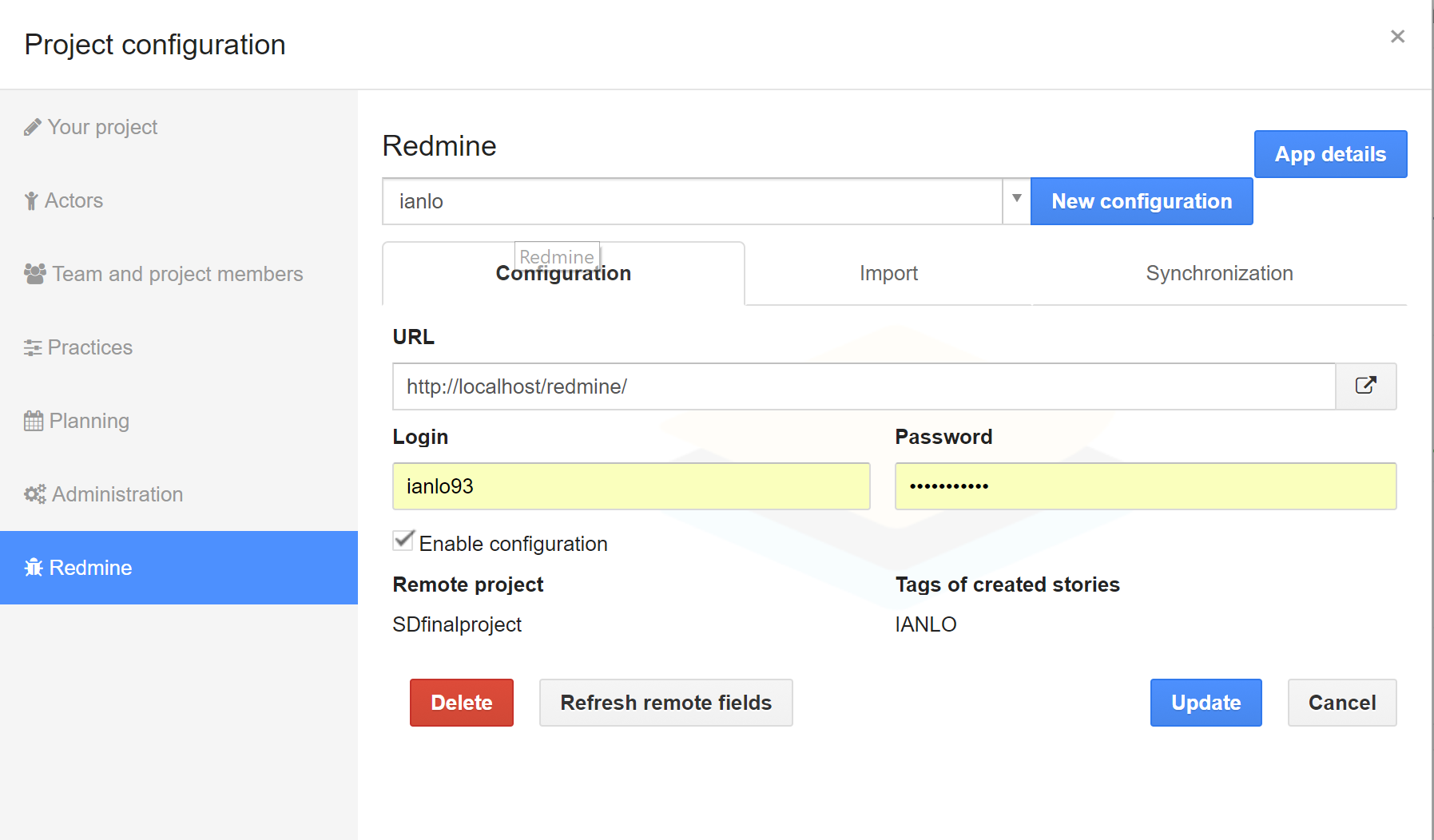


Fig. Connect Configuration

After connection, the bug track query can be found in the iceScrum.

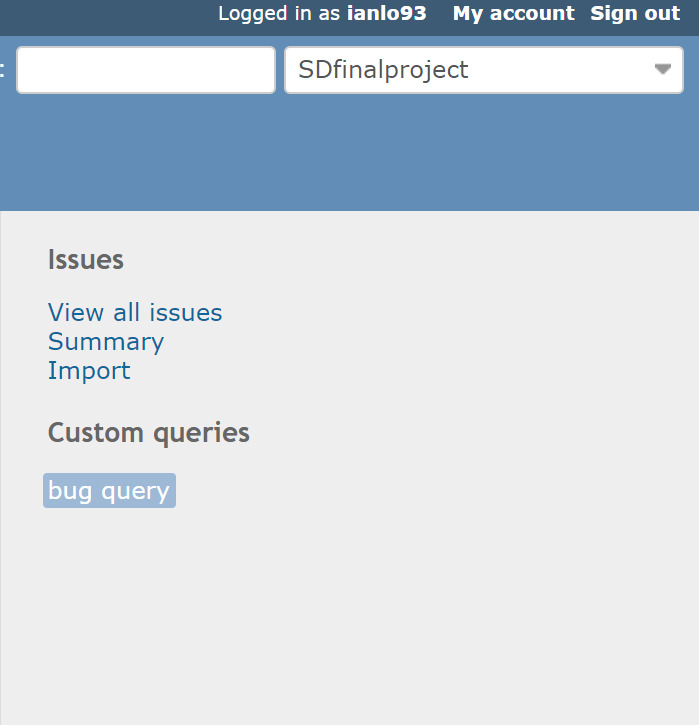
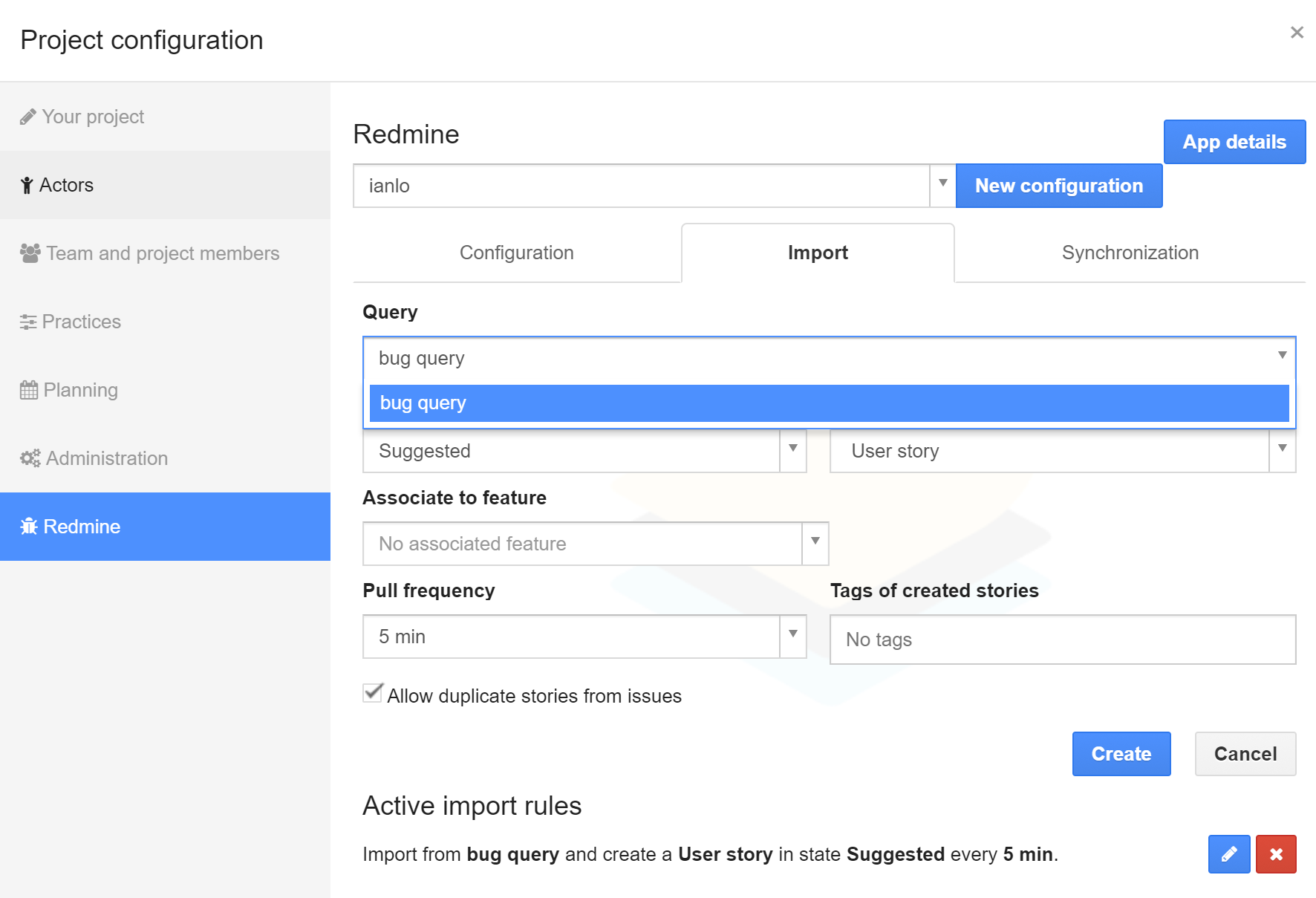
 

Fig. Bug Query in IceScrum