Lab 11: Jira Searching, Filtering, and JQL

Exercise 1: Using Basic Search

1. In the left menu, go to: **Filters → View all filters → Create filter**
2. Make sure the search mode is **Basic** not JQL
3. Apply filters:
    - Project: Select your project
    - Status: Done
    - Assignee: Yourself
4. Play around with filters again and observe how the results update.
5. Switch to **JQL** mode to see the JQL generated automatically.

Deliverable

Screenshot of basic search being used.

Exercise 2: Using JQL Search

Steps

1. Write the following JQL:

   project = YOURPROJECTKEY

3. Add more conditions:

   AND assignee = currentUser()

   AND status != Done

4. Add ordering:

   ORDER BY priority DESC, created ASC

5. Try other useful filters:

   issuetype = Bug

   labels in (frontend, backend)

   text ~ "login"

6. Experiment with at least **three different queries.**

Deliverable

Screenshot of queries used.

Exercise 3: Create and Save a Filter

1. Remember the **In Review** status we created last lab? Let's make a filter for it. Write the following JQL:

   project = YOURPROJECTKEY

   AND status = "In Review"

2. Click **Save as**.
3. Name it: **Issues in Review**
4. Set permissions:
   - Viewers: Space
5. Verify:
   - Go to **Filters → All filters**
   - See the newly created filter

Deliverable

Screenshot of created filter with its output

Exercise 4: Create a Quick Filter using JQL in Board (Optional)

This exercise is for company-managed projects only. For team-managed projects it is slightly different under the name *Custom Filters*. Steps will not be the same but feel free to explore if you want.

1. Open your project board.
2. Go to **Board Settings → Layout → Quick Filters**
3. Create a new Quick Filter:
    - Name: **Only My Issues**
    - JQL:

      assignee = currentUser()
4. Create another Quick Filter:
    - Name: **Only Bugs**
    - JQL:

      issuetype = Bug
5. Switch back to the board and test both filters.

Extra Extension: Create a Swimlane based on JQL

1. Board **Board Settings → Layout → Swimlanes**
2. Select: **Queries**
3. Add one:
    - Name: **High Priority**
    - JQL:

      priority = High

Deliverable

Screenshot of created quick filter and optionally the created swimlane

Exercise 5: Using Jira REST API (Completely Optional)

Use a bash script to call Jira Cloud's REST API and retrieve issues from a project.

Prerequisites

1. You must have:
    - A Jira Cloud account
    - A project key (ex: **SCRUM**, **DEMO**, **TEST**)
2. Generate an API token:
    - Go to https://id.atlassian.com/manage/api-tokens
    - Create API token
    - Make sure you copy the API key and store it somewhere secure
3. Your Jira domain, for example:
   https://yourcompany.atlassian.net

Jira API Documentation

https://developer.atlassian.com/cloud/jira/platform/rest/v2/intro/

Step 1

Create a new file named: **jira_create_issue.sh**

Copy this code and replace the placeholders with your data.

```
#!/bin/bash

JIRA_EMAIL="your-email@example.com"
JIRA_API_TOKEN="your_generated_api_token"
JIRA_DOMAIN="yourcompany.atlassian.net"
PROJECT_KEY="YOURPROJECTKEY"
```

Note: DO NOT upload your API key to GitHub or share it with anyone. There are other ways to securely use API keys in scripts but for the purpose of this lab just use it in the script.

Step 2

Add this script then save the file. This will create a new issue in Jira using the API.

```
curl -s -u $JIRA_EMAIL:$JIRA_API_TOKEN \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{
    "fields": {
      "summary": "Created from Bash script",
      "project": { "key": "'"$PROJECT_KEY"'" },
      "issuetype": { "name": "Task" }
    }
  }' \
  "https://${JIRA_DOMAIN}/rest/api/3/issue"
```

Step 3

Run the file using **./jira.sh** from a Bash terminal (you can use Git Bash)

Deliverable

Screenshot of script output and created issue in Jira.