Bugs Life

Introduction

According to Wikipedia, issue tracker is a software application that keeps track of reported software bugs in software development projects. Many bug tracking systems, such as those used by most open-source software projects, allow end-users to enter bug reports directly. Other systems are used only internally in a company or organization doing software development. Typically bug tracking systems are integrated with other project management software.

Many famous software development platforms such as GitHub have offered this feature. You may refer to that. However, this issue tracker system that you are going to program will have much more features than GitHub's issue tracker.

In simple terms, an issue tracker system allows anyone to report a bug of the software project so that the software developer team is able to track it, discuss and resolve it in an organized way. Besides, issues can be recorded by priority, tags and timelines which makes it more organized and easier to track.

Problem Statement

The programmers in your company named "Bugs Everywhere Sdn Bhd" encounter bugs in their software projects more than anyone else. This directly causes some delays to the submission dateline of the software projects. As a result, your company has been receiving many complaints from the clients. Your company is on the verge of bankruptcy. Hence, the CEO of the company has assigned you and your team to program a Java desktop application to track all the issues from projects so that issues can be created, shared and discussed with the rest of the team to quicken the process of resolving the issues of projects.

Project Requirement

User Interface

- Users must login to perform action.
- Users can check issues on the dashboard.
- Users can create issues.
  - Set title, descriptive text, tag, priority, assignee.
- Users can comment under issues.
- Users can react to the comments.

Project Dashboard

Your application should have a project dashboard that will list out all the projects.
The project can be sorted alphanumerically or by project ID. A project can contain multiple issues, by selecting the project, the program will lead the user to the project's issue dashboard.

Sample I/O:

```
Project board
-------------
+----+------------------+--------+
| ID |   Project Name   | Issues |
+----+------------------+--------+
|  1 | TableView Widget |      3 |
+----+------------------+--------+
Enter selected issue ID to check project:
```

Issue Dashboard

Your application should have an issue dashboard that will list out all the issues.
The issue can be sorted by priority or timestamps, up to the user's needs. Users can also filter the issues by status or tag. A priority queue will help you store the issues better.

For issues, change histories, or known as changelog should be recorded and can be viewed by the user, on their request.

On this board, users can select which issue they want to look into, and lead them to the issue page.

Users can trigger the search feature on this issue board.

Sample I/O:

```
Issue board
+----+-------------------------+-------------+----------+----------+------------------+----------+------------+
| ID |          Title          |   Status    |   Tag    | Priority |       Time       | Assignee | CreatedBy  |
+----+-------------------------+-------------+----------+----------+------------------+----------+------------+
|  1 | Can't display the table | In Progress | Frontend |        4 | 2019/08/07 15:44 | btan     | jhoe       |
|  2 | Can't open file         | Open        | Backend  |        3 | 2019/08/08 15:44 | jhoe     | btan       |
+----+-------------------------+-------------+----------+----------+------------------+----------+------------+
Enter selected issue ID to check issue
or 's' to search
or 'c' to create issue:
```

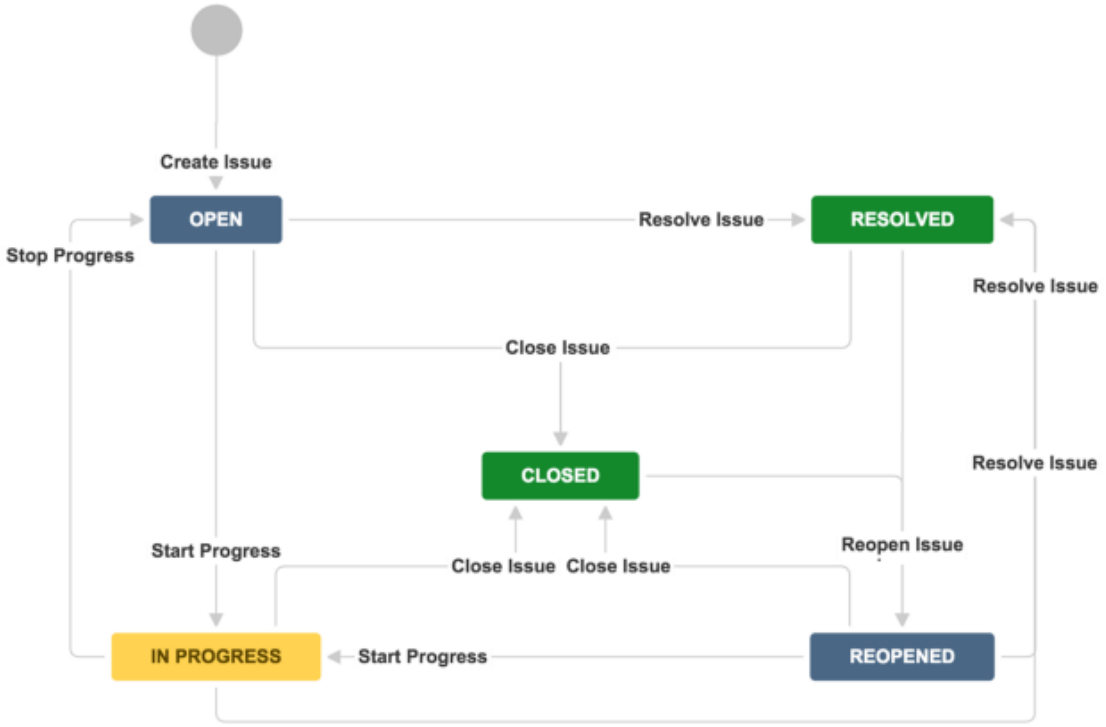
Issue page

Your application should show all the details of the issue, and also all the comments related to this issue, with all their respective information.

Sample I/O:

```
Issue ID: 1     Status: In Progress
Tag: Frontend   Priority: 4     Created On: 2019/08/07 15:44
Can't display the table
Assigned to: btan               Created By: jhoe

Issue Description
-----------------
Hi,

I'm receiving the "Can't display the table because Power BI can't determine the relationship between two or more
fields." error message and I do not understand why. I have searched the net for help but can't find a post that
answers my question.

It may well be a straight modelling restriction but best to check.

I have two datasources (Sheet 1 and Sheet 2). Both are Excel files.

Comments
---------
#1      Created on: 2019/08/07 16:20    By: liew
Please find below links to the two .pbix files I've created in an attempt to conquer this error I'm receiving.
$$ 1 people react with happy

#2      Created on: 2019/09/01 12:22    By: ang
Try this https://www.dropbox.com/home?preview=Relationships.pbix

Just I removed the relationship between sheet1 and sheet1BridgingTable, sheet2 and sheet2BridgingTable
$$

Enter
'r' to react
or 'c' to comment
or 'help'for more commands:
```

Issue Lifecycle



Your issue tracker should allow the issues to be created, put into process, closed, resolved, and reopened based on the issue lifecycle.

Search feature

Users should be able to make a search on the issues, and the search results should return a list of possible results like in the issue board. The search should look into the title, descriptive text and the comments as well.

Issue Class

You should have an issue ADT that store the issues with following attributes:

- Issue no.
  This is the unique number of the issue, you can assume it is a record no.
- Issue title
  The title of this issue.
- Description text
  This is the issue description text, which describes the issue, and the size of the description text is unlimited, there can be few MBs of data.
- Timestamp

The exact time when the issue is created.

- Creator_user
  Show who created this issue.
- Assignee_user
  Issue must be assigned to the assignee to solve it. This store the assignee that will solve this. (Can be null)
- Comments
  The issue can have multiple comments under it which enable the discussion to solve the issue. The comments should be the one of the Comment class. You may use ArrayList to store the comments. (Can be empty)
- Tag
  Issue can be tagged, and this stores the tag of this issue. Some examples of tags include: “frontend”, “backend”, and other self-defined tags. (Can be null)
- Priority
  This is the priority of the issue, and it is an Integer that is in range of 1-9. The higher the value is, the higher the priority of the issue is.
- Status
  The status will tell whether this issue is now “Resolved”, “Closed”, “Open”, or “In Progress”. Reopened issue will have the status “Open”.

Comment Class

Comment ADT should be implemented to store the following attributes:

- Comment text
- Comment ID
- Comment_user
- Timestamp
- Reaction (angry, happy, smile, etc. You can create more reaction.)

Undo/Redo Feature

The text in issue and comments should be able to redo/undo. You can use stack data structure to implement this.

Data Storage

You should implement a local file database that stores the data in JSON document format. You can use some external library like **gson, jackson** to implement this feature, and of course, you are encouraged to create your own json parser. Importing libraries (if you wanted to) can be done by using package managers like maven or gradle.

Your program should at least be able to read the provided initial data into your own local database. (Download the initial json file here: https://jiuntian.com/data.json.) You can also modify the json file to have extra attributes if you have created some extra features later.

Learn more on how to parse JSON object to POJO (Plain Old Java Object): https://github.com/google/gson/blob/master/UserGuide.md (gson)
https://github.com/FasterXML/jackson (jackson)

If you have decided to use an external database as data storage, you should still develop an option for admin to import and export data into JSON objects.

Report Generation

Able to create weekly reports for projects, so that higher management can easily understand the performance of the projects or teams. Your report should at least contain: numbers of issues resolved, unresolved, in progress, top performer in team, most frequent label, and so on. You can make even better reports if you decide to implement the “Statistic” part mentioned in the extra feature.

Some Crazy Ideas

Chat

Chat is useful especially when the communication between team members is needed. You can try to create an online chat feature, that allows users to chat to other users.

Security and Access Control

You can create an Access Control List, which defines which type of users is allowed to perform particular actions. This ACL will be strictly imposed to restrict the user to perform unauthorized actions. A logger can also be developed to allow admin to trace the abnormal activities.

Client-Server Architecture

This project requires client-server architecture, which consists of a server that serves the needs of multiple client software. The server application has to respond to the client request of service, and the server will be in charge of data storage, data processing, data retrieval, data query, and so on. The server should be able to serve multiple clients at same time. The client application should not store any data, and only a user can use one application at same time. The client application can be connected to the server with some sort of connection like socket, TCP, etc.

External Database

Use external databases such as MySQL, PostgreSQL, MongoDB, MSSQL, Oracle SQL, with

JDBC driver to provide persistent data storage.

Advance Searching

Implement a full text fuzzy searching, that can search the issue in an approximate manner (viz., not exact pattern). It should be able to ignore some typos like what search engines do. A search engine is very important so that users can easily trace for resolved or closed issues for their reference.

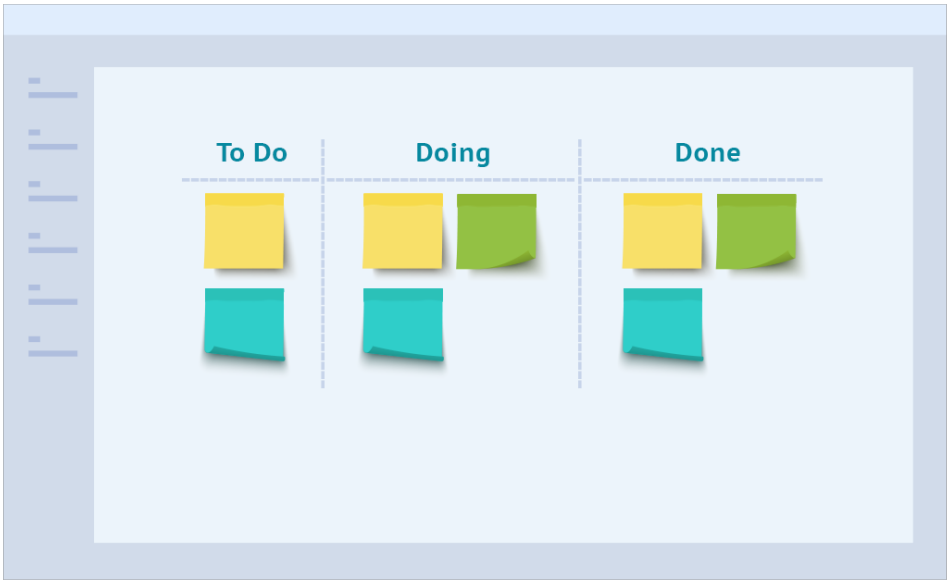
Graphical User Interface (JavaFX or Java Swing)

You should implement either Web application or standalone application with Graphical User Interface (GUI), doing both doesn't sum up marks for you. Build the project with a GUI that allows the user to interact with. The GUI must be able to perform all of the required features and proposed crazy ideas. If you are doing GUI, you should enable the user to upload a picture in the comment as well. (It's 2020, only ancient human post issue without picture that couldn't describe the problem well.)

As for web application, it should be implemented as client server architecture. However, the java application codes are only for the backend, and it may just run on any frontend of your choice. You can try to use some framework like Spring to start with if you're interested in making this.

Kanban Board

In the real world, the tasks or issues of programmers in a project development are typically managed using the Agile methodology and assisted via Kanban, these have made programmers' work somehow more organized and efficient.
Agile is a structured and iterative approach to project management and product development. Kanban is a visual system for managing work as it moves through a process. Kanban visualizes both the process (the workflow) and the actual work passing through that process. The goal of Kanban is to identify potential bottlenecks in your process and fix them so work can flow through it cost-effectively at an optimal speed or throughput. You could read more about kanban here: https://www.digite.com/kanban/what-is-kanban/.



Markdown Support

Your application can read the text that is in markdown format and display it in the way it should be. Markdown support is meant to be provided for the text area in posts and comments. Read more here: https://www.markdownguide.org/getting-started/

Statistic

It is the era of Big Data and Data Science, hence visualizing data is an important skill for all of us. Your application can plot the graph of the issues. You can have any type of statistic plot you wanted, but should at least have the following:

- Statistic of tags (the number of issues for each tag)
- Status (the number of issues for each status)
- Time (the number of issues in day/week/month)
- Plot a line graph that show the issue frequency over selected time range
- And others, based on your own creativity

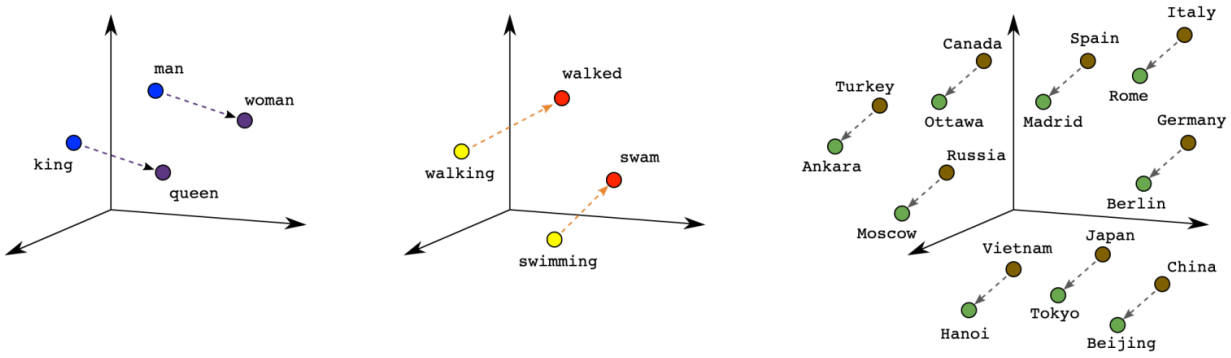
So, you have created an easy to understand graphical statistic for your busy boss, you can get your raise now.

Machine Learning in Tagging

Train a machine learning model that can automatically do the tagging? Some libraries you can try are Deeplearning4j, etc.

Clustering Analysis

Have you learnt K-NN in machine learning before? You must have heard of clustering. Clustering can help us find similar issues. How about we make a clustering analysis that groups similar questions together and show them in a graph? There are some machine learning models that can do this, such as Word Embedding, PCA, t-SNE, etc. You can make use of your own creativity.



Male-Female Verb Tense Country-Capital

Your Own Creative Ideas

Of course, we would be more than happy if you can come up with any creative extra features which are **BETTER than ours** and **PRACTICAL** to this system.

Enquiries / Clarification

If you have found any part of the instruction unclear or you have any inquiries about the question, or even if you want to report any problem, you may raise your issues on GitHub Issue at https://github.com/jiuntian/IssueTrackerAssignment/issues. All answered questions will be publicly listed too, but we do not provide straight answers for you. Why do we use GitHub while this question is requiring you to do an issue tracker? Well, we need a better issue tracker that is done by you and your mates. Good luck in your journey in doing better issue tracker, hope to see your great work.

END OF THE QUESTION