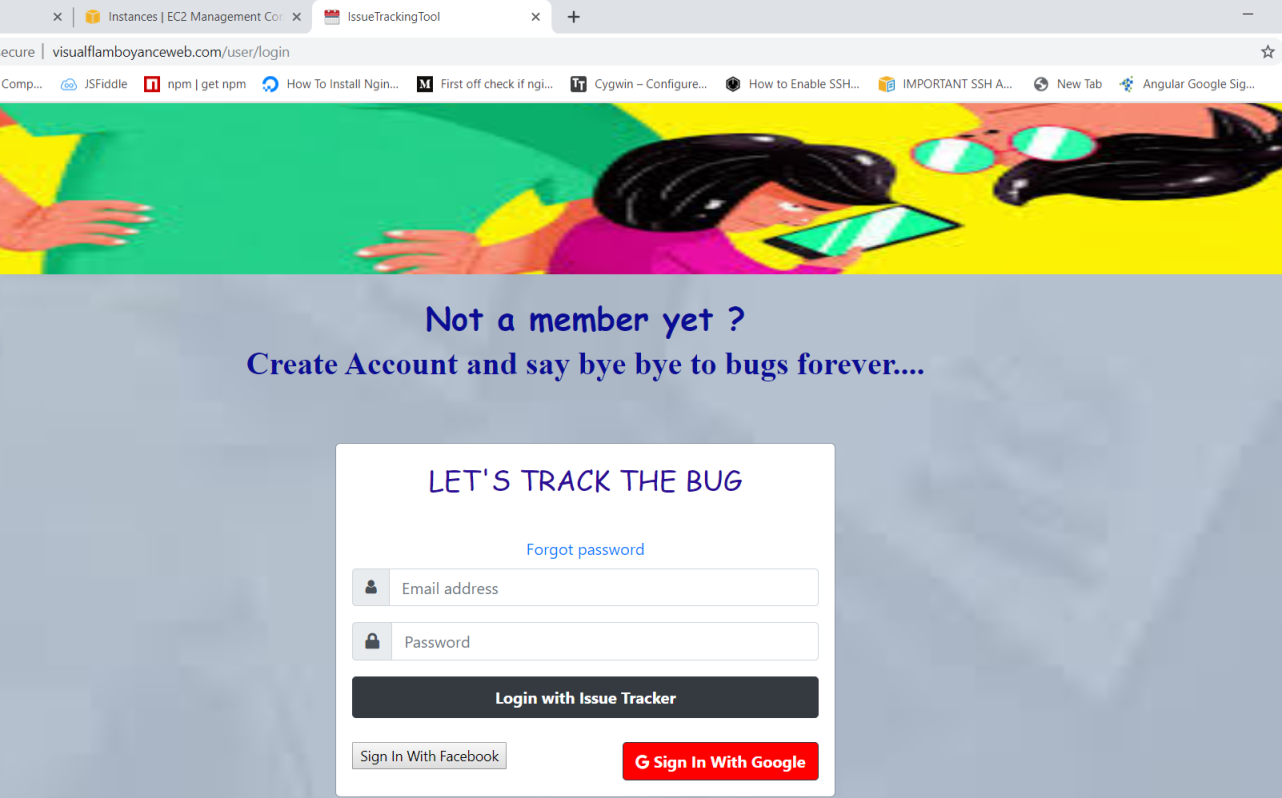
# [visualflamboyanceweb.com/user/login](http://visualflamboyanceweb.com/user/login)



s3 buckets: <https://visual-cloud.s3.ap-south-1.amazonaws.com/issue_tracker_apidoc.7z>

# ISSUE TRACKING TOOL

**Project Description**

This project is a ready to deploy, Issue Tracking Tool.

Reporter reports an issue and assigns it to a user.

Reporter can add/edit title of issue, Description of issue, add/edit any related attachments, like screenshots. Users including reporter and assignee, can assign the issue, to any another user (called assignee, hereafter). Also there a comments section, where any user, should can comment around this issue

Since it is a MEAN Stack application,there are two separate parts of the project.

1. FRONTEND APPLICATION
2. BACKEND APPLICATION

* The frontend part is developed using Angular,Bootstrap,CSS,HTML 5,JS
* The backend part(REST API with real time functionalities) is developed using Node JS,Express JS
* Nodemailer: It is a module for Node js applications to send emails.
* Database: MongoDB and Redis Server

APPLICATION MODULES :

1. Personalized-Dashboard Module
2. User Module
3. Shared Module
4. Issue-description Module
5. Server-error Module

Each module contains components based on the functionalities of the application.

COMPONENTS :

1- USER MODULE

Sign-up, Login ,Reset-password ,Update password.

2-PERSONALIZED-DASHBOARD MODULE :

Home Component

1. ISSUE-DESCRIPTION VIEW- Create-Issue component, Edit- Issue component, view-issue component.

Note : (I have kept the components separate intentionally and not merged in just one view for better understanding and my learning)

4-SHARED MODULE

my-nav

4- server-error component

5-Page-not-found component

**Features of the Application**

Features of the platform​-

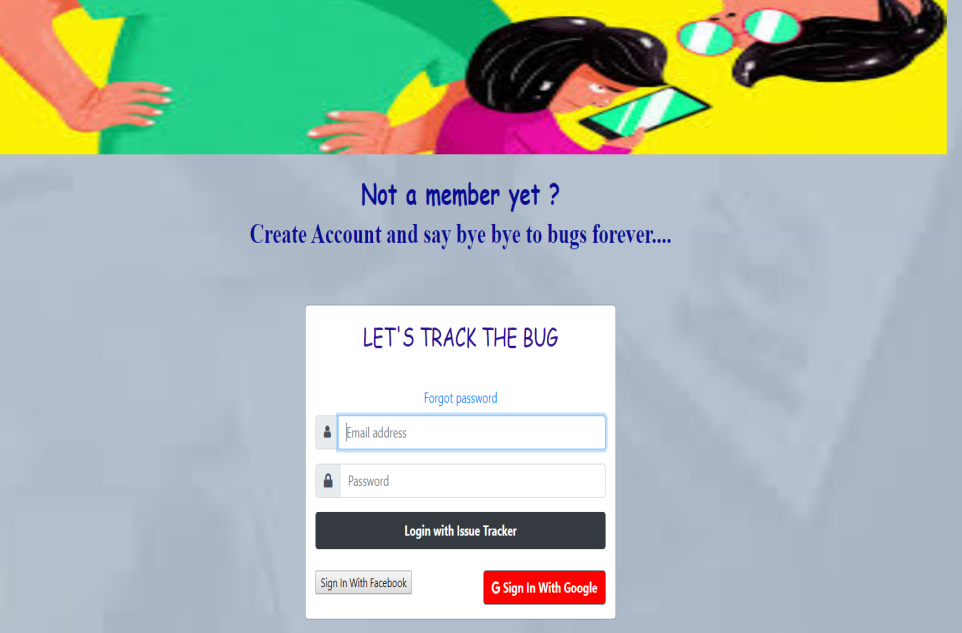
1. Login View : comes under user module🡪 login component

2. Personalized Dashboard View : comes under personalized dashboard module ->home component

3. Issue description view : comes under issue-description module -> (create issue component,edit –issue component,view-issue component)

4. Search view : comes under personalized dashboard module -> (home component in the form of table)

1. Login View



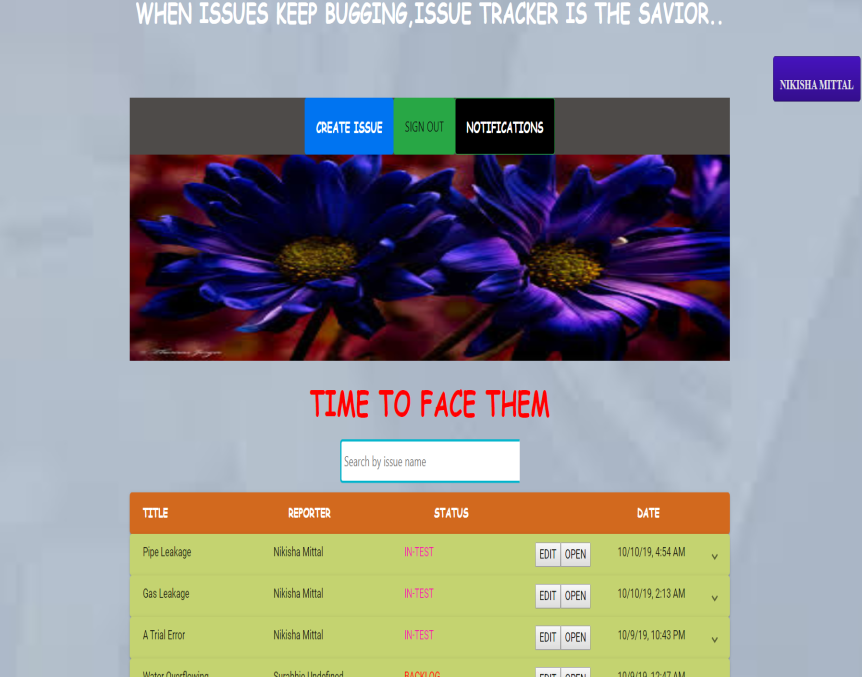
a. User is able to login to the system through his username/password or social logins. . (user module -> login component)

b. User is able to register also. (user module -> sign-up component)

c. Upon login, user should land on his ​Personalized Dashboard View​.

(personalized-dashboard view->home component)

2. Personalized Dashboard View



A table showing all issues currently assigned to logged-in user is shown.

It has following columns.

1. Status : current status of the issue i.e backlog, in- progress,­­­­­­­ in-test, done.

2. Title:​ Title of the bug.

3. Reporter:​ User who reported the bug

4. Date:​ Date when this bug was reported.

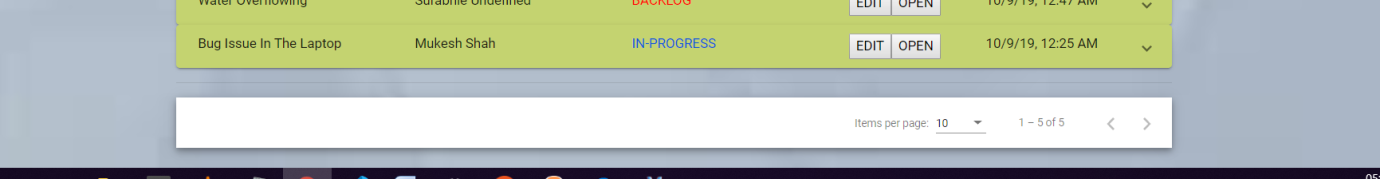
5. A search box where, User should be able to search for an issue, which would open

​Search View.

6. A create button: To log a new issue. On clicking, user should be taken to issue description view. (CREATE ISSUE BUTTON)

NOTE:

1. Table is paginated



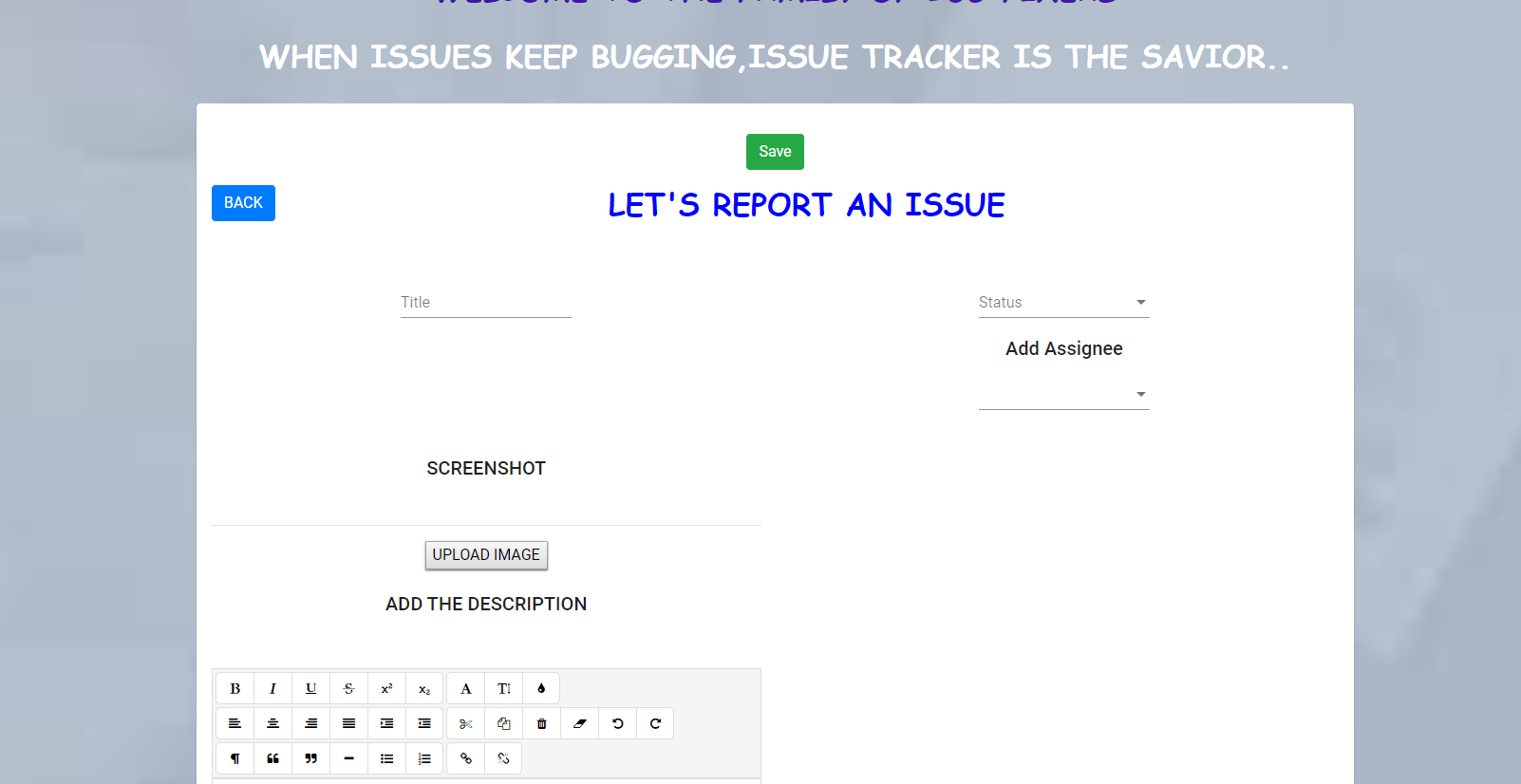
2. Table has sorting on columns

3. User can filter rows based on any columns.

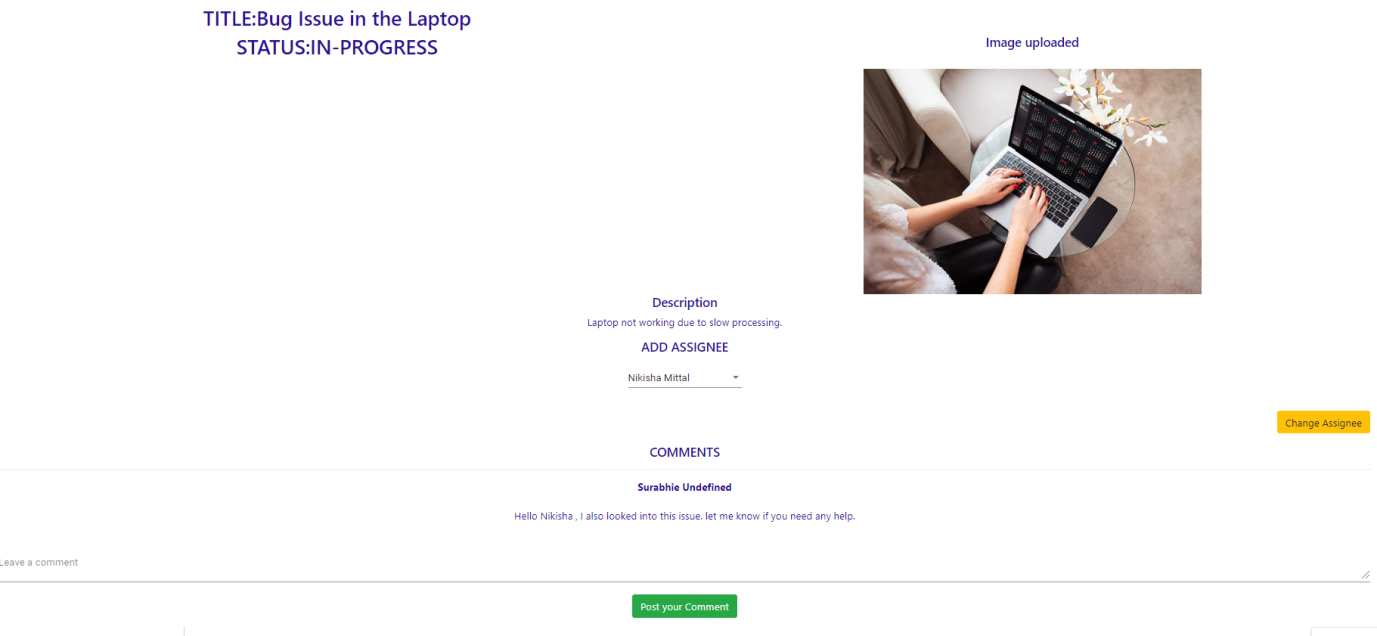
4. Upon clicking on any row, ​Issue Description View​ opens (ARROW IS AT THE LAST )

1. Issue description view:

a-create-issuecomponent



b-edit-issue component (this view comes when userid and reporter id are different)



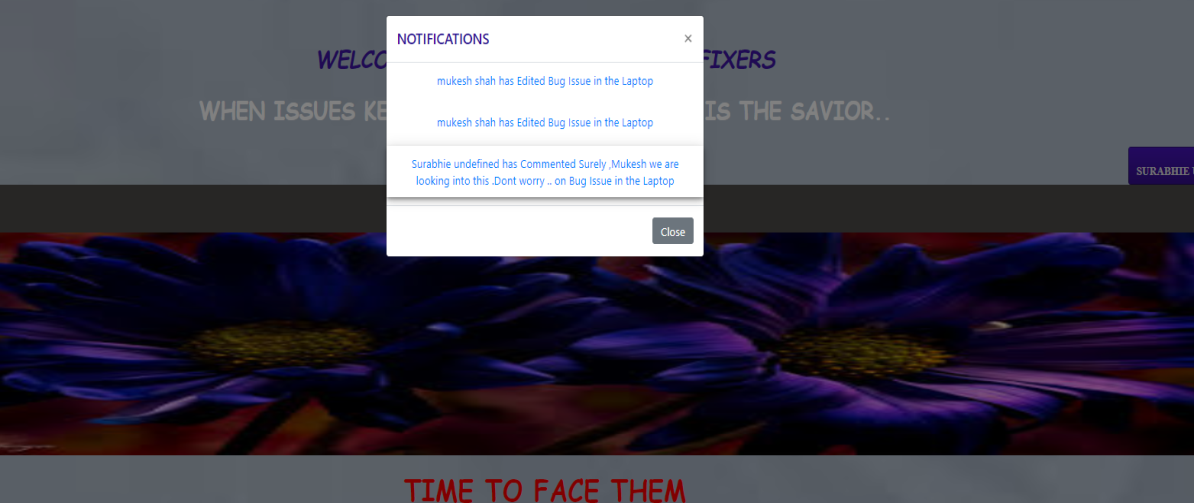
c-issue-view (simple-view)



1. Here user(Reporter) can add/edit title of issue(create issue and edit issue component) ,

Description of issue, add/edit any related attachments, like screenshots. Description Box is a rich text editor having options like font styling, underline etc. .

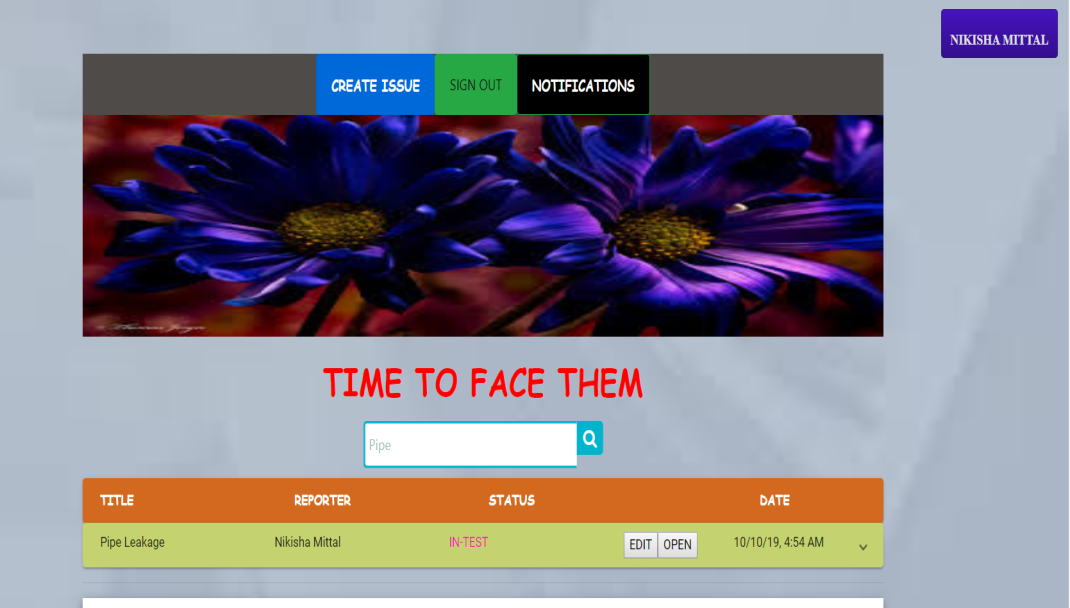
1. Any user, including reporter and assignee, is able to assign this issue, to any another user (called assignee, hereafter).
2. Any user can make changes to this issue.
3. 3. Also, there is a comments section, where any user, is able to make comments around this issue.
4. Further, Any user should is able to add himself, as watcher, to this issue.(if he is not the reporter)
5. A button called "Watch" is there, for this.
6. List of watchersis also shown(There can be more than one watcher).
7. All watchers, assignee and reporter, receive notifications, for any changes or comments on this issue.
8. Notification comes on screen, having a short description of what changed,



1. Also, when clicked on notification, user lands on the ​Issue Description View ​of related issue.

(edit-issue view)

4. Search view



1. User is able to search for any text.

2. User is shown a results table, having all the issues related to the search text.

3. This table is similar to the table in ​Personalized Dashboard​ ​View​.

GIT REPOSITORY LINK :

1. https://github.com/Surabhie/Issue-tracker-Frontend.git
2. https://github.com/Surabhie/Issue-tracker-Backend.git

SUBMITTED BY:

SURABHI