**Bugtracker SRS 1.0**

Product Description: Bugtracker (final name TBD) is a helpdesk application that tracks issues in a server-side database with user authentication, project tracking, and version control with team hierarchy.

Estimated Tech to be used: Java with Netbeans, GitHub database hosting with MySQL, PHP, Firebase or 0Auth, SQLDBM for data modelling

DB model: [https://app.sqldbm.com/SQLServer/Edit/p57592/#](https://app.sqldbm.com/SQLServer/Edit/p57592/)

ALT names: ProjectManager, BugHub,

Features: The UI will be built in Netbeans.

* **MAIN SCREEN**
  + Left-to-right tables rendered from database to select project, version, and bugs list.
  + UX: selective feature locking and “\*required” details to guide users
  + Login via Firebase/0Auth with simple profile details.
  + New project button: sets current user to project lead
    - New window popup for new database entry
  + Edit project button: New Window pops up autofilled with current details, overwrites database entries, records changes made
  + Delete project button: Only visible to project lead. Confirmation popup. Deletes tables associated with project.
    - Backup autosave in local space for restoration?
* **NOTIFICATIONS SCREEN**
  + Render from notifications database
  + Checkbox to hide read notifications
  + Project, leader, etc
* **NEW/EDIT PROJECT SCREEN**
  + Name textfield \*REQUIRED
  + Project Lead dropdown of team members \*REQUIRED
  + Add team member button: enter username/email of team member to add
    - Sends email/notification to user inviting them to access the project
    - Displays whether the user has accepted the invitation or not
    - Team-member gains access to project table -> needs security, dynamic relational linking, download of project details
  + Remove team member button: removes user from project, removes access to project
    - Needs security: authorization managed by cloud; login automatically loads accessible projects from cloud; nothing is kept in local storage
  + Repo link(optional)
* **NEW ISSUE SCREEN**
  + Popup window for new issue
  + Submit option sends notification to project lead for verification; delete likewise

Databases: MySQL with GitHub hosting

* Users sign in with github; builds new database on their repo? Or should I provide a server? Better if each team has their own database; project lead sets up database
* Users table
  + Agile-based roles for each user on each project
  + Notifications table
    - Id, read, timestamp, project, message etc.
  + Projects table
    - Versions?
    - Issues table
      * Id, severity, priority, OS, assignee, status, resolution, summary, timestamp
        + Database of progress entries for each day with timestamp?