

Personal Issue Tracker Proposal

XTeam-212

April 3, 2020

Contributors:

Martin Diges: mdiges@wisc.edu

Tyler Johnston: tjjohnston2@wisc.edu, tjohnston@cs.wisc.edu

Mingrui Leng: mleng2@wisc.edu

Alec Lowry: lowry3@wisc.edu

Contents

1	Problem	2
2	Primary Stakeholder	2
3	Graphical User interface	2
4	Data Structure	3
5	Input Data File Format	3
6	Output Example	4
7	Milestones	4
7.1	Front End (Tyler and Martin)	4
7.2	Back End (Alec and Mingrui)	4

1 Problem

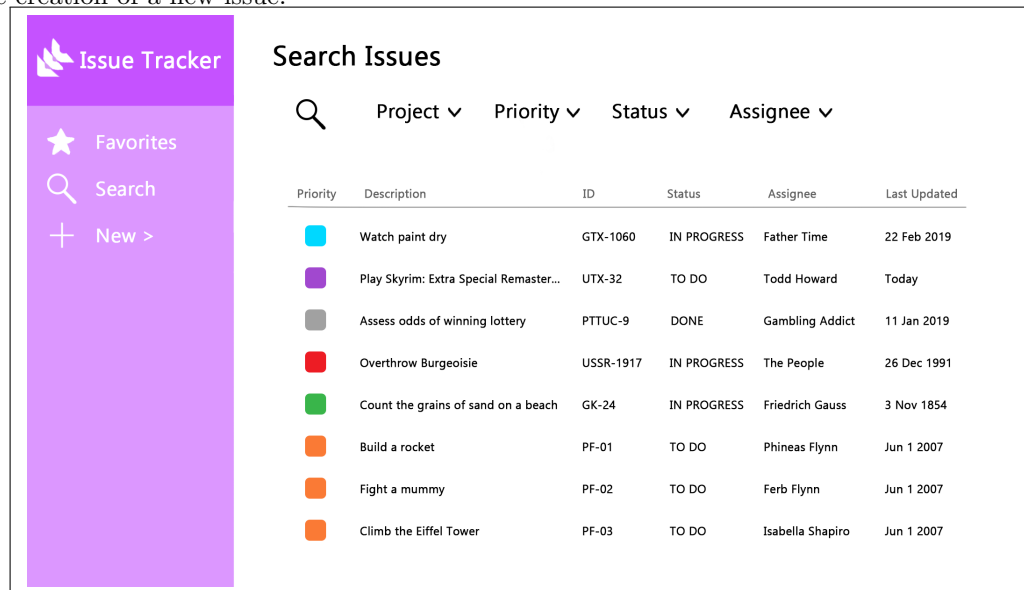
The goal of this product is to offer a comprehensive tool for tracking issues and features to be implemented. Will compliment git version control for easy tracking of the development and implementation process. In addition, it will allow for easy handling of issues that pop up throughout the development process.


2 Primary Stakeholder

Primarily designed with software development in mind, but could be used for most projects in the development and maintenance phase.

3 Graphical User interface

Upon application launch, IssueTracker initializes to a screen representing all current projects and issues (figure 1). Figure 2 represents a sample screen for the creation of a new issue.




Issue Tracker

★ Favorites

🔍 Search

+ New >

New Issue

Title
 Take Back Middle Earth

Description
 The kings of men will lead the frontal assault on Mordor's armies, hopefully with the aid of the elves. Aragorn will flank the enemy with a legion of phantoms. Frodo and Sam will be moving closer to Mount Doom as the battle progresses.

Issue ID
 FFT-90 Generate ID

Add Assignees +
 Legolas
 Gollum
 Gandalf

Publish Issue

4 Data Structure

<<ProjectList>> ProjectList <ul style="list-style-type: none"> * Field: int size * Field: Project head * Field: Project tail * Op: insert(project) -> {true,false} * Op: remove(project) -> {true,false} * Op: get(project) -> {Project project, null} 	<<Project>> Project <ul style="list-style-type: none"> * Field: String name * Field: String description * Field: Date lastUpdated * Field: IssueList roadmap * Field: IssueList unresolved * Op: getters * Op: setters
<<IssueList>> IssueList <ul style="list-style-type: none"> * Field: int size * Field: Issue head * Field: Issue tail * Op: insert(issue) -> {true,false} * Op: remove(issue) -> {true,false} * Op: get(issue) -> {Issue issue, null} 	<<Issue>> Issue <ul style="list-style-type: none"> * Field: Long ID * Field: String title * Field: String description * Field: Date lastUpdated // Update in every setter * Field: Boolean resolved * Op: getters * Op: setters

5 Input Data File Format

N/A

6 Output Example

See GUI sketches

7 Milestones

7.1 Front End (Tyler and Martin)

- Initialize Program - Martin
- Kill Program - Martin
- Deserializing JSON - Tyler
- Add new Issue - Tyler
- Edit existing Issue - Tyler
- Remove Issue - Tyler
- Represent fields of Issue - Tyler
- Create new Project - Martin
- Edit existing Project - Martin
- Remove Project - Martin

7.2 Back End (Alec and Mingrui)

- Build an Issue Object - Alec
- Build a generic list DT - Mingrui
- Build a list of Issues - Mingrui
- Build a Project Object - Alec
- Implement a list of Projects - Mingrui
- Writing to Database (JSON) - Alec