CIS 658 – Building a Bug Tracking App
Winter 2019
Homework #4

Due: 6pm, February 7, 2018

Note: Work in pairs on this assignment.

Part 1: Bug Overflow – Modeling Bugs

In this first part of the assignment you will create a simple web application using the Rails framework. In particular, we will be working on a sequence of homework assignments that will iteratively build a bug tracking application we will call Bug Overflow. Use a recent version of Rails (5.x) and Ruby.

Iteration #1
In this first iteration, use what you have learned in class to create a single model called "bug" that models the bugs in our application. Later we will add additional models to the app. Bugs are represented with this information:

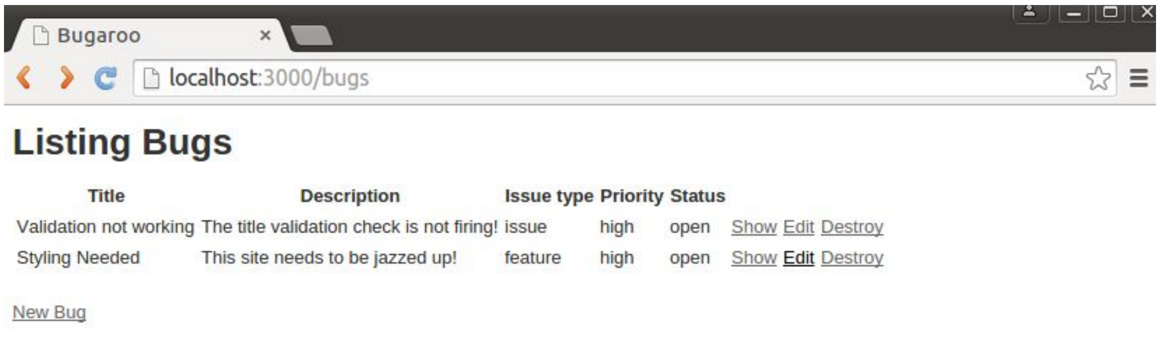
- title: a short descriptive title summarizing the problem.
- description: a longer piece of text which describes the details of the problem.
- issue_type: has three possible values: issue, enhancement, feature.
- priority: priority of the issue with three possible levels: low, medium, high.
- status: the current status of the bug with three possible settings: open, closed, monitor.

Use rails scaffolding to generate the basic screens for managing bugs (list of all bugs, show bug, edit bug, and delete bug).

Using the TDD approach outlined in lecture, write the automated tests and then implement the following model requirements:

1. Title and description cannot be blank whenever a bug is created or edited.
2. Issue_type, priority and status must have valid values (you should represent these values as enumerated types.)
3. The default value of issue_type should be feature.
4. The default value of priority should be medium.
5. The default value of status should be open.

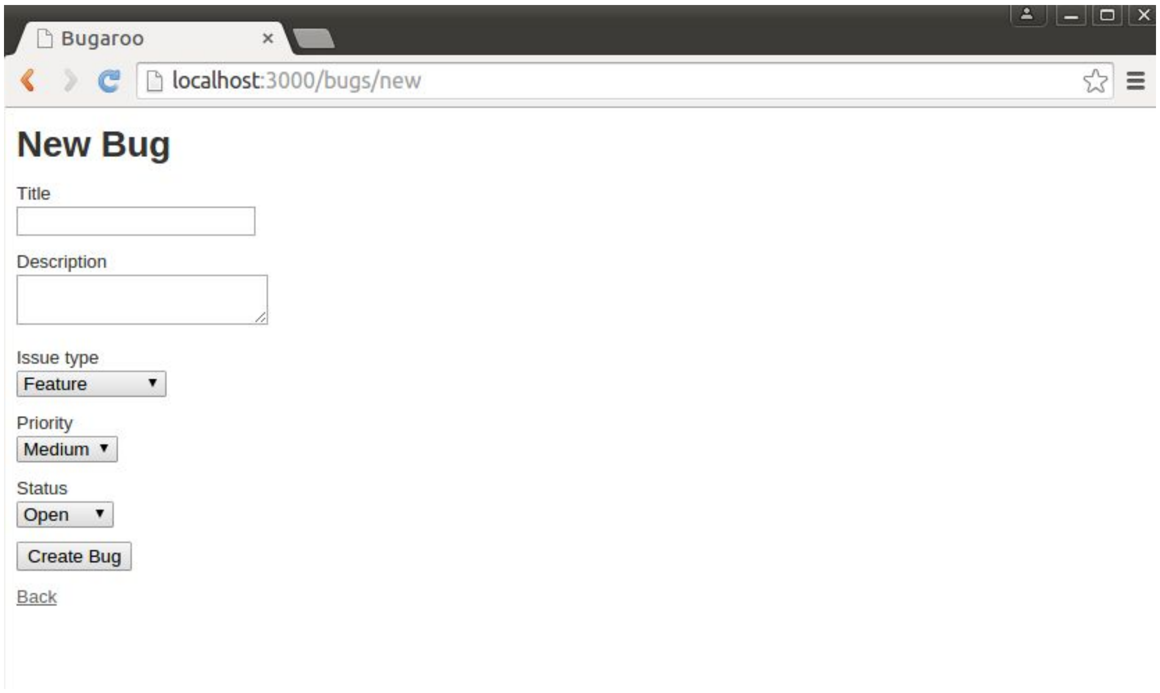
Do NOT pay any attention to styling at all in this point (e.g. you need not modify the generated CSS/SCCS files.) Your final product should look like the screenshots inserted below.



Listing Bugs

Title	Description	Issue type	Priority	Status			
Validation not working	The title validation check is not firing!	issue	high	open	Show	Edit	Destroy
Styling Needed	This site needs to be jazzed up!	feature	high	open	Show	Edit	Destroy

Figure 1. Bug listing view



New Bug

Title

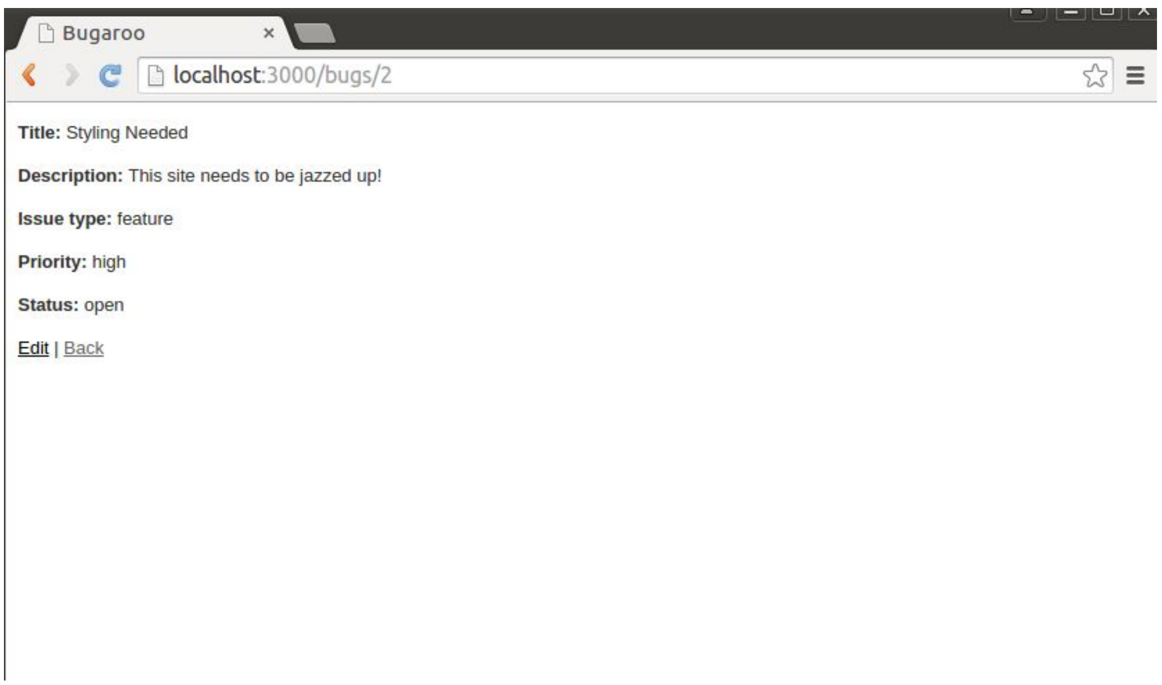
Description

Issue type: Feature

Priority: Medium

Status: Open

Figure 2. New bug view.



Title: Styling Needed

Description: This site needs to be jazzed up!

Issue type: feature

Priority: high

Status: open

Edit | Back

Figure 3. Bug show view.

Part 2: Bug Overflow – Associating Users With Bugs

In this part of the assignment you will extend the Bug Overflow application that you started in the previous section by adding a User model to the app and associating users and bugs.

Iteration #2
Users typically report bugs, so we need to have a model for users. In this iteration of the application, go ahead and add an additional model for users. For each user you should have the following fields:

- lname: user's last name.
- fname: user's first name.
- email: user's email address.
- thumbnail: a reference to the user's thumbnail image.

Use rails scaffolding to rapidly generate the model and basic screens for managing authors. (list of all authors, show author, edit author, and delete author).

Using the TDD approach outlined in lecture, write the automated tests and then implement the following model requirements:

1. lname, fname, email cannot be blank whenever an author is created or edited.

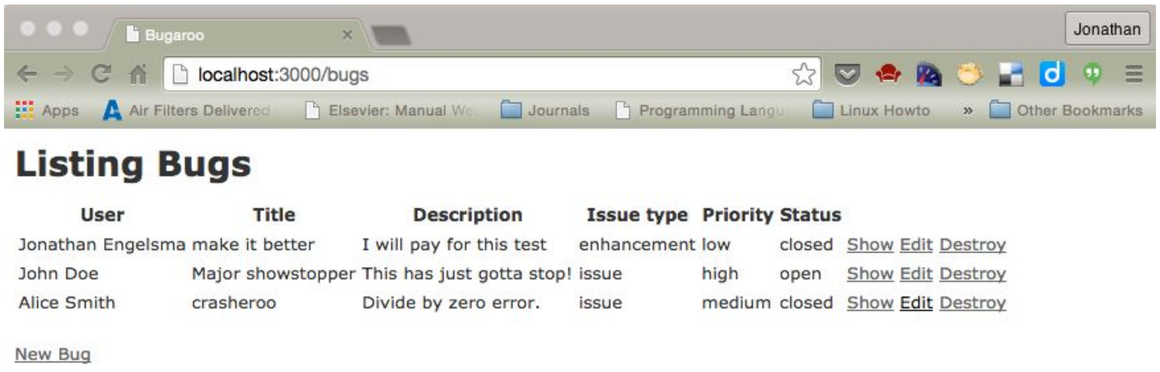
2. email must be unique and be a valid format for an email address.
3. thumbnail must end with either .png, .jpg, or .gif. It is ok to leave the thumbnail field blank.

Formally model the association between users and bugs so that every bug is owned by a single user, and a single user may own many bugs. In addition, make sure you modify the appropriate bug views so that a user can be specified when a bug is entered/updated, or shown in the bug listing or bug detail views. Make sure that you present users as "Firstname Lastname" when they are shown in views.

Do NOT pay any attention to styling at all in this point (e.g. you need not modify the generated CSS/SCCS files.) Your final product should look like the screenshots inserted below. Your user views (not shown below) should also be fully functional.

Deliverables:
Provide the instructor a demo of your working app in class on February 22, and post your github link to Slack in a direct conversation. Be sure to add your partner to the conversation so I know who to give credit to!

Note: In lieu of a live demo in class, you can simply post a link to a deployed copy of your app on Heroku in the Slack direct conversation. You still need to post your github URL as well.

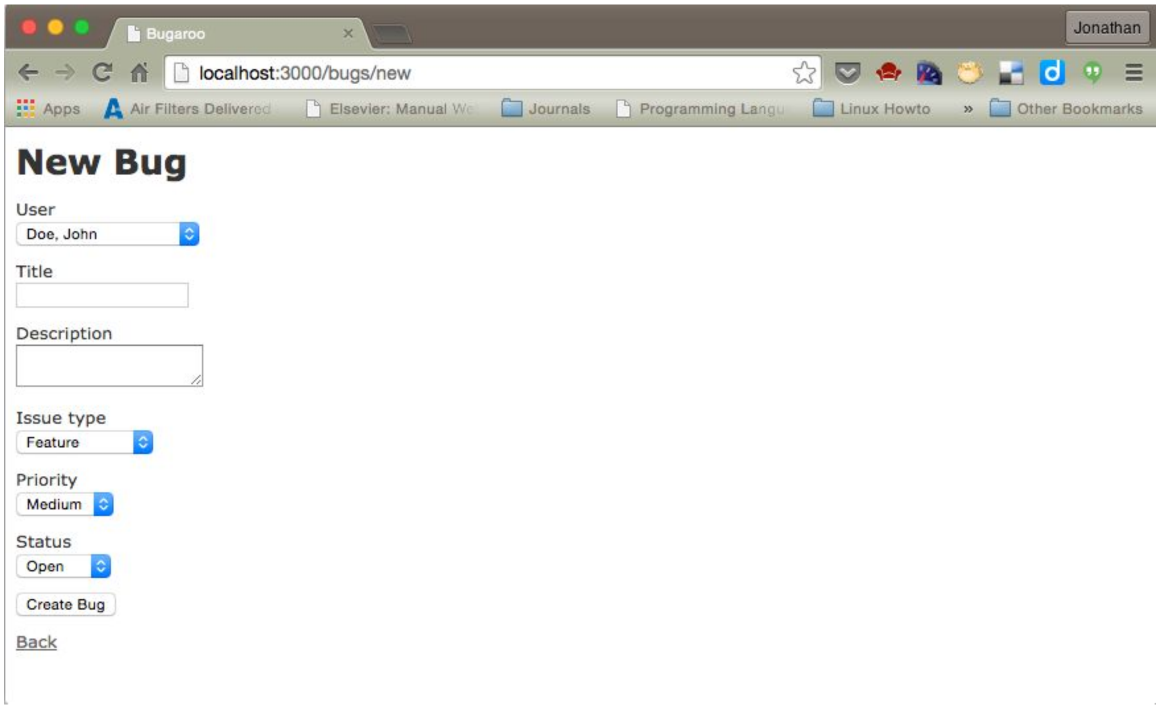


Listing Bugs

User	Title	Description	Issue type	Priority	Status			
Jonathan Engelsma	make it better	I will pay for this test	enhancement	low	closed	Show	Edit	Destroy
John Doe	Major showstopper	This has just gotta stop!	issue	high	open	Show	Edit	Destroy
Alice Smith	crasheroo	Divide by zero error.	issue	medium	closed	Show	Edit	Destroy

New Bug

Figure 4. Bug listing view with reporting user included.



New Bug

User
Doe, John

Title

Description

Issue type
Feature

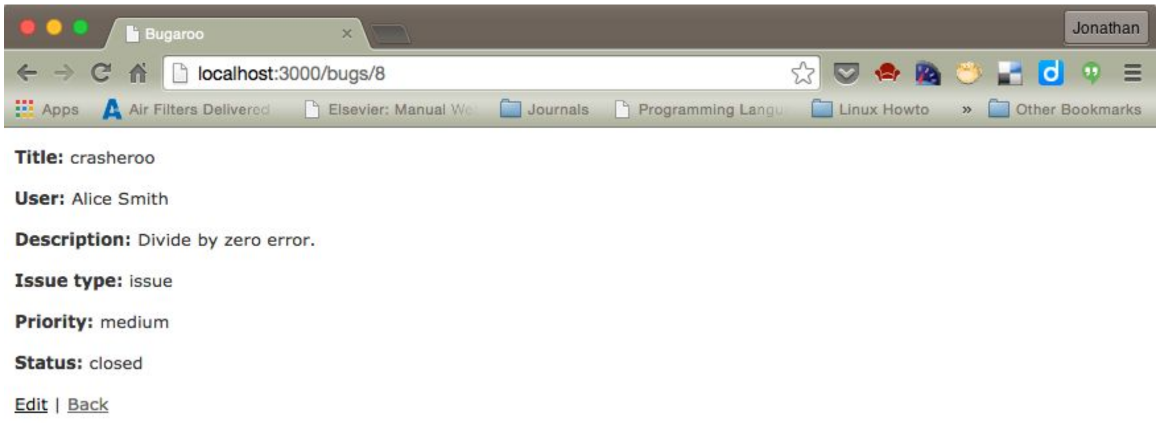
Priority
Medium

Status
Open

Create Bug

Back

Figure 5. New bug view with ability to associate user.



Title: crasheroo

User: Alice Smith

Description: Divide by zero error.

Issue type: issue

Priority: medium

Status: closed

Edit | Back

Figure 6. Show bug view with user info showing.