COMP 4081

Exam 1

Fall 2015

Name: Solutions , _____

Last name First name

Rules:

- No potty breaks.
- Turn off cell phones/devices.
- Closed book, closed note, closed neighbor.

- WEIRD! Do not write on the backs of pages. If you need more pages, ask me for some.

Reminders:

- Verify that you have all pages.
- Don't forget to write your name.
- Read each question carefully.
- Don't forget to answer every question.

1. [5pts] What type of system is Git? Describe three software engineering problems Git helps you solve.

Git is a version control system.

Here are three problems it helps you solve:

- Undoing changes or reverting to older versions
- Multiple developers collaboratively modifying a code base in parallel.
- Developing multiple versions of a system in parallel. For example, you might be maintaining a release version while also working on an experimental new version.

(There may be more acceptable answers than just these three.)

Consider the following list of Git commands:

a)	`git init`	f)	`git branch`
b)	`git push`	g)	`git clone`
c)	`git add`	h)	`git checkout`
d)	`git merge`	i)	`git status`
e)	`git commit`	j)	`git pull`

Alice is working on a collaborative software project with a team of seven other developers. The project is an airline-booking web app called *FlyMe*. The code for the project is housed in a GitHub repo. All work for the project is being done on the "master" branch (no other branches). Alice has been helping on the project for a while, and has a local copy of the repo and a working directory on her computer.

2. [2pts] Alice has just edited the web-app code on the master branch to add a feature that enables users to request window or aisle seats when they book flights. She wants to save these changes in her local repo. Which command(s) from the above list should she run next?

c, e

3. [2pts] Having saved the changes in her local repo, Alice now wants to share them with the rest of the team by uploading them to the GitHub repo. Which command(s) from the above list should she run next?

b

When she runs the command(s), she gets this message (with words that give away the answers hidden):

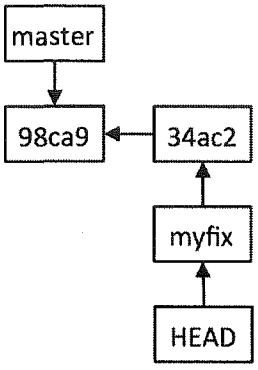
```
To https://github.com/.../flyme.git
 ! [rejected]        master -> master (fetch first)
error: failed to [redacted] some refs to 'https://github.com/.../flyme.git'
hint: Updates were rejected because the remote contains work that you do
hint: not have locally. This is usually caused by another repository [redacted]ing
hint: to the same ref. You may want to first integrate the remote changes
hint: (e.g., 'git [redacted] ...') before [redacted]ing again.
hint: See the 'Note about fast-forwards' in 'git [redacted] --help' for details.
```

4. [2pts] Alice wants to resolve this issue, so she can upload her changes. Which (one) command from the above list should she run next?

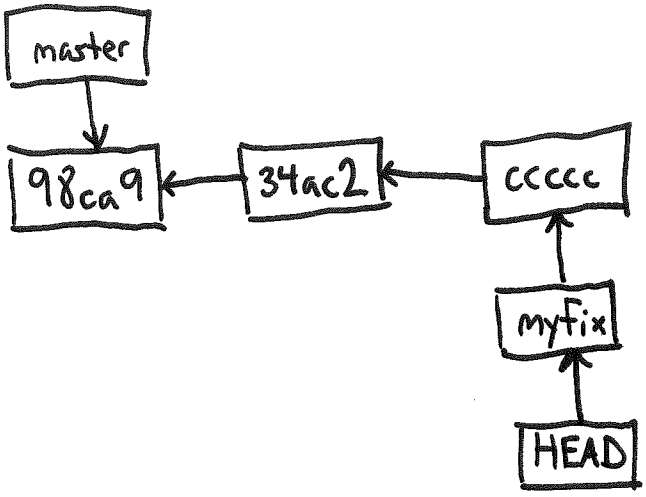
j

5. [3pts] Draw the state of the pictured repository after a Git **commit** operation (make up a hash).

Before:



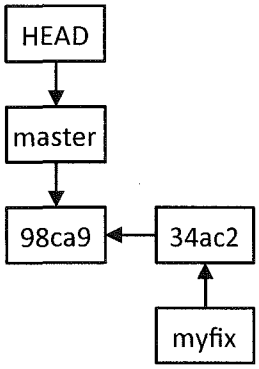
After:



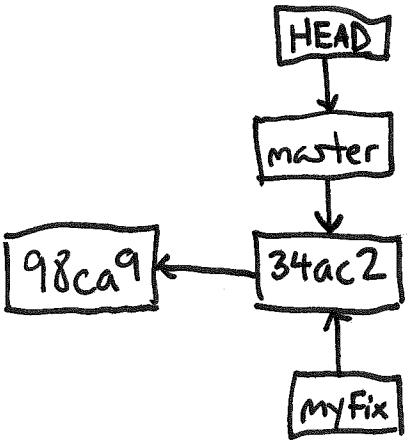
6. [3pts] Draw the state of the pictured repository after running the following command.

```
git merge myfix
```

Before:



After:



Note that this is a "fast forward" merge

7. [6pts] For each of the three major parts of the MVC architectural pattern, tell the name of the part and its main responsibilities in the pattern.

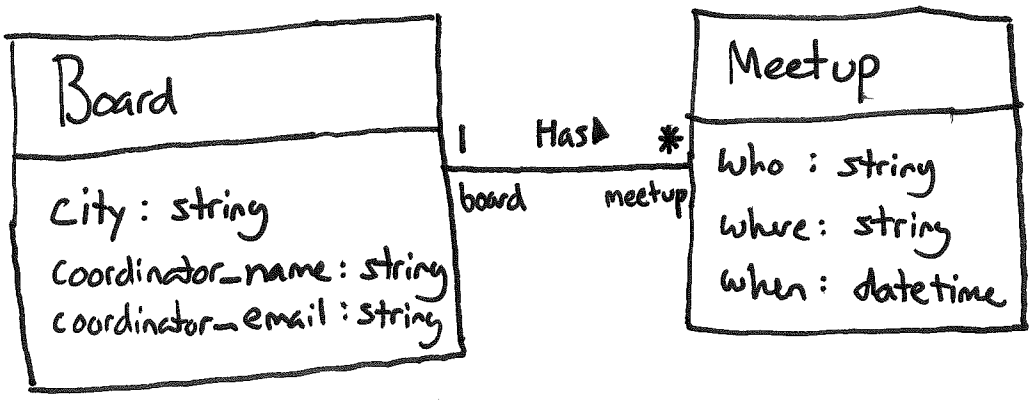
Model: Responsible for providing the app's "business logic".

View: Responsible for providing the app's user interface (UI).

Controller: Responsible for translating UI actions to operations on the model.

The questions on the following pages refer to the example figures. The figures show different aspects of the *MeetMe* web app that enables people to post "meetup" opportunities to "boards". Each city has its own board with one person who serves as coordinator.

8. [8pts] Draw a UML class diagram that represents the model classes given in Figure 1. Be sure to label all associations and association ends, and include all multiplicities. Don't include "id" attributes (objects have identity by default). You may also omit the "datetime" attributes that Rails provides by default.



9. [8pts] Fill in the missing test code in Figure 2 such that the test checks that the model class' validation will catch a "where" attribute that has too few characters. Recall that all Rails model classes have a `valid?` method, and the test base class provides `assert` and `assert_not` methods. Also, you can retrieve a model fixture object with a line like this:

```
subway = meetups(:subway)
```

```
subway = meetups(:subway)
subway.where = "X"
assert_not subway.valid?
```

10. [14pts] Write the missing ERB code in Figure 4 such that it renders pages that look like the page depicted in the figure. Do not hard code values. Rather, they should come from an `@meetups` object that is passed to the ERB. In particular, `@meetups` is an array of `Meetup` objects.

```
<% @meetups.each do |meetup| %>
  <tr>
    <td><%= meetup.who %></td>
    <td><%= meetup.where %></td>
    <td><%= meetup.when %></td>
    <td><%= link_to 'Cancel', meetup, method: :delete %></td>
    <td><%= link_to 'Change', edit_meetup_path(meetup) %></td>
  </tr>
<% end %>
```

11. [2pts] If you wanted to change the HTTP request URL that maps to a particular controller action, which Rails component would you need to modify?

    a. Controller class

    b. Model class

    (c.) Routes class

    d. Migration class

    e. All of the above

12. [2pts] Which of the following types of Rails components sets up the database tables?

    a. Controller classes

    b. Model classes

    c. Routes classes

    (d.) Migration classes

    e. All of the above

13. [2pts] What type of HTTP request would be generated by pressing the "Create Meetup" button on the form in Figure 5.

    a. GET

    (b.) POST

    c. PATCH

    d. DELETE

    e. None of the above

14. [2pts] Which of the following lines of code would the `MeetupsController#index` action contain?

    a. `@meetup = Meetup.new`

    b. `@meetup = Meetup.find(params[:id])`

    c. `@meetup = Meetup.new(meetup_params)`

    (d.) `@meetups = Meetup.all`

    e. None of the above

15. [2pts] Which of the following lines of code would the `MeetupsController#new` action likely contain?

    a. `@meetup = Meetup.new`

    b. `@meetup = Meetup.find(params[:id])`

    c. `@meetup = Meetup.new(meetup_params)`

    d. `@meetups = Meetup.all`

    e. None of the above

16. [2pts] True or false? Bundler, RVM, and Vagrant all help with configuration management.

    a. True

    b. False

17. [2pts] True or false? Controller actions that modify the database (such as the `create` action) should end by sending an HTTP redirect response to the browser (instead of rendering an HTML page to send in the response).

    a. True

    b. False

Figures

```
# == Schema Information
#
# Table name: boards
#
#  id                :integer          not null, primary key
#  city              :string
#  coordinator_name  :string
#  coordinator_email :string
#  created_at        :datetime         not null
#  updated_at        :datetime         not null
#

class Board < ActiveRecord::Base
  has_many :meetups
  validates :city, presence: true
  validates :coordinator_name, presence: true
  validates :coordinator_email, presence: true
end
```

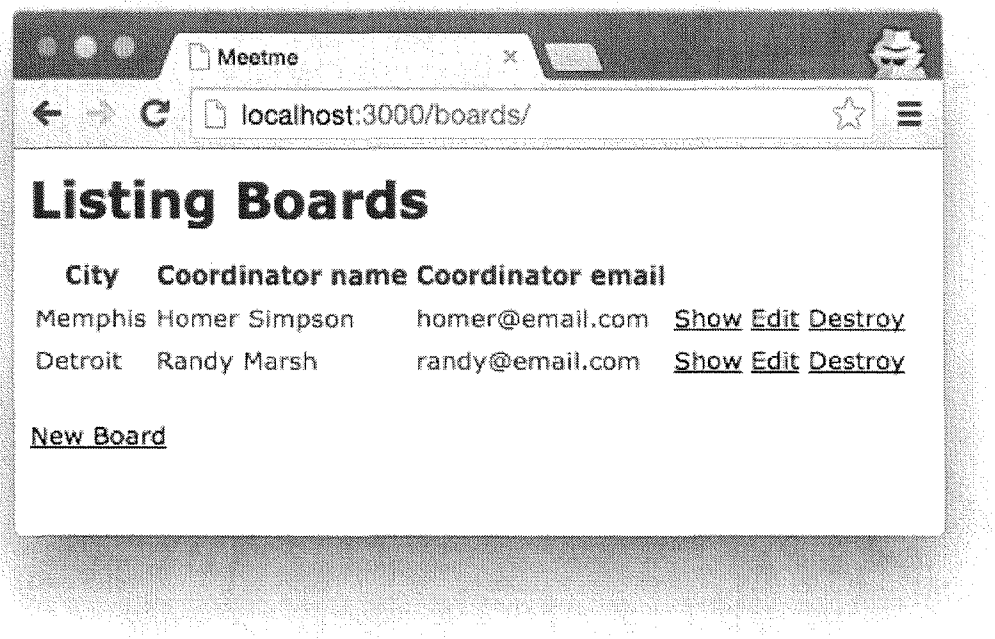
```
# == Schema Information
#
# Table name: meetups
#
#  id          :integer          not null, primary key
#  who         :string
#  where       :string
#  when        :datetime
#  created_at  :datetime         not null
#  updated_at  :datetime         not null
#  board_id    :integer
#

class Meetup < ActiveRecord::Base
  belongs_to :board
  validates :who, presence: true
  validates :where, length: { minimum: 3 }
  validates :when, presence: true
end
```

Figure 1. Model classes for the MeetMe web app.

```
mcdonalds:
  who: Ronald McDonald
  where: McDonald\'s
  when: 2015-10-10 22:00:00

subway:
  who: Jared Fogle
  where: Subway
  when: 2015-10-26 12:30:00
```

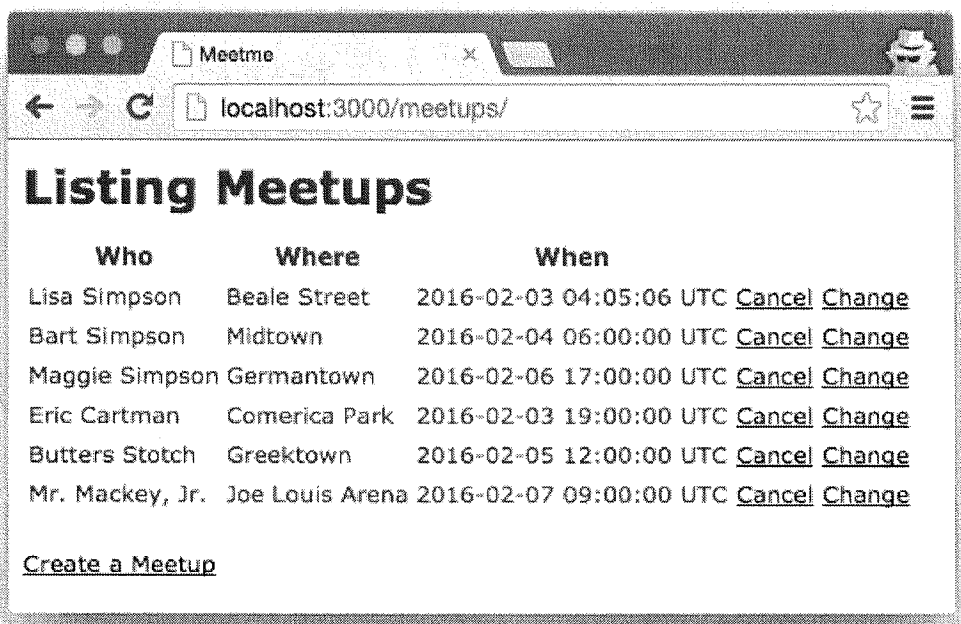
```
require 'test_helper'

class MeetupTest < ActiveSupport::TestCase

  # test "the truth" do
  #   assert true
  # end

  test "where should be longer than 3 characters" do
    Fill in this code
  end

end
```

Figure 2. Test fixture (upper) and test case (lower). [Oops. The test string should say "at least 3 characters".]




```
<h1>Listing Boards</h1>

<table>
 <thead>
  <tr>
   <th>City</th>
   <th>Coordinator name</th>
   <th>Coordinator email</th>
   <th colspan="3"></th>
  </tr>
 </thead>

 <tbody>
  <% @boards.each do |board| %>
   <tr>
    <td><%= board.city %></td>
    <td><%= board.coordinator_name %></td>
    <td><%= board.coordinator_email %></td>
    <td><%= link_to 'Show', board %></td>
    <td><%= link_to 'Edit', edit_board_path(board) %></td>
    <td><%= link_to 'Destroy', board, method: :delete, data: { confirm: 'Are you sure?' } %></td>
   </tr>
  <% end %>
 </tbody>
</table>

<br>

<%= link_to 'New Board', new_board_path %>
```

Figure 3. "index" page for the Board model class.




```
<p id="notice"><%= notice %></p>

<h1>Listing Meetups</h1>

<table>
  <thead>
    <tr>
      <th>Who</th>
      <th>Where</th>
      <th>When</th>
      <th colspan="2"></th>
    </tr>
  </thead>

  <tbody>
    Fill in this code
  </tbody>
</table>

<br>

<%= link_to 'Create a Meetup', new_meetup_path %>
```

Figure 4. "index" view for the Meetup model class. "Cancel" deletes a meetup, and "Change" links to an edit form.

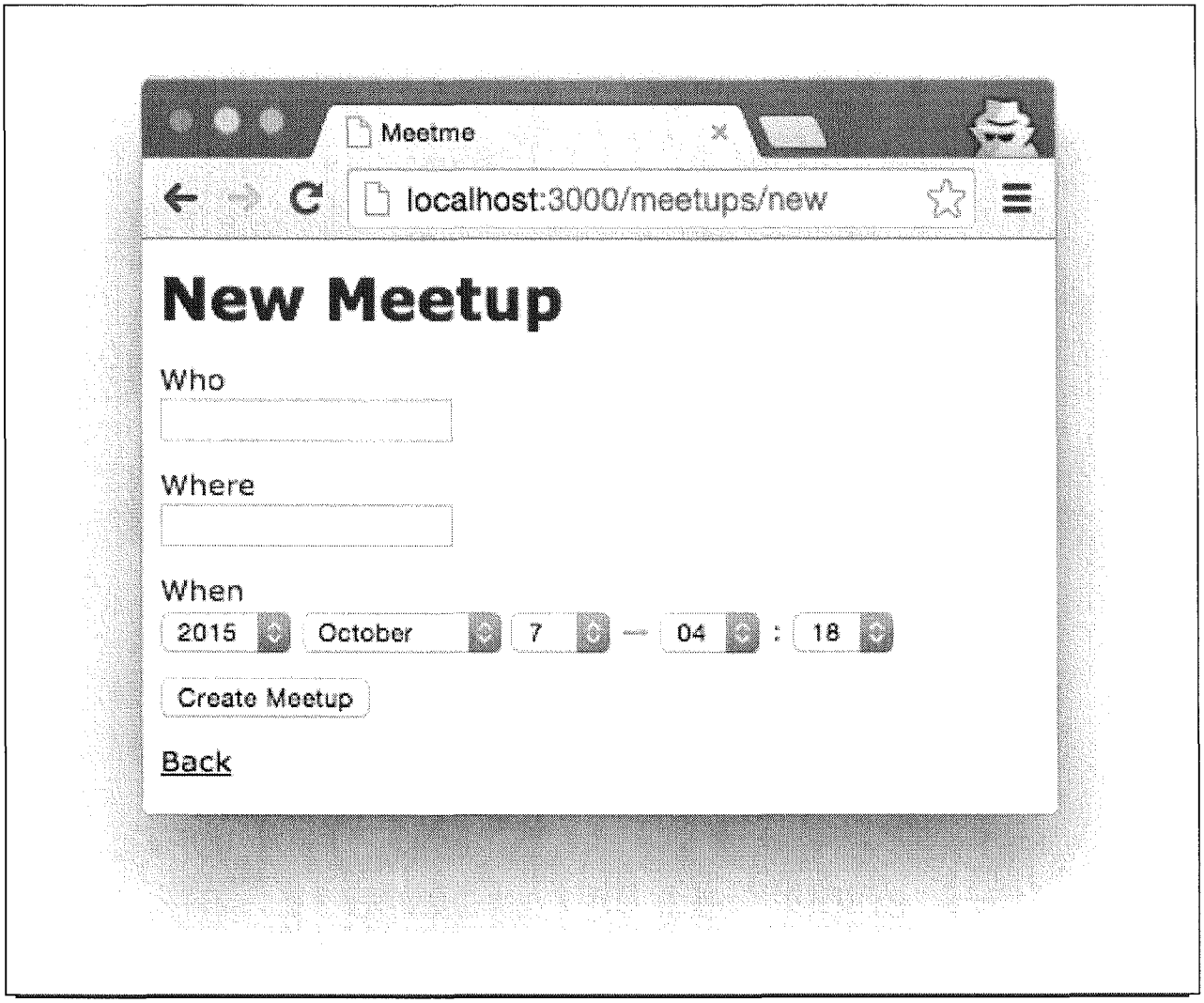


Figure 5. The form for creating a new meetup.