COMP 7012

Exam 1

Spring 2018

Name: ____________________________, ________________________________

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. [3pts] What three things does great software deliver?

2. [2pts] What type of system is Git?

3. [2pts] Name two problems that Git helps to solve?

Consider the following list of Git commands:

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

Ruchi has just joined a team of developers collaboratively working on a fashion advice web app called *GetDressedRight*. 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).

4. [2pts] Ruchi wants to get a local copy of the code and repo, so she can begin contributing to the project. Which command(s) from the above list should she run?

5. [2pts] She makes some changes to the code. Which command(s) from the above list should she run to save her edits to the local repo?

6. [2pts] Having saved to her local repo, she would now like to share her work with the rest of the team (via GitHub). Which command(s) from the above list should she run?

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

```
To https://github.com/.../getdressedright.git
 ! [rejected]        master -> master (fetch first)
error: failed to [redacted] some refs to 'https://github.com/.../getdressedright.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.
```

7. [2pts] Ruchi wants to resolve this issue, so she can upload her changes. Which (one) command from the above list should she run next?

8. [2pts] The command completes with no conflicts. What command(s) should Ruchi run to at last share her work with the rest of the team?

9. [2pts] If Ruchi wanted to look at a previous version of the code, which command would she use?

10. [4pts] Draw the state of the pictured repository after a Git **commit** operation (call your hash *cccccc*).

Before:

$$\begin{tikzcd}
\boxed{\text{aaaaaa}} & \boxed{\text{bbbbbb}} \arrow[l] \\
 & \boxed{\text{master}} \arrow[u]
\end{tikzcd}$$

After:

11. [4pts] Draw the state of the local repository after a Git **pull** operation.

Local Repo Before:

$$\begin{tikzcd}
\boxed{\text{aaaaaa}} & \boxed{\text{bbbbbb}} \arrow[l] \\
 & \boxed{\text{master}} \arrow[u]
\end{tikzcd}$$

Remote Repo:

$$\begin{tikzcd}
\boxed{\text{aaaaaa}} & \boxed{\text{bbbbbb}} \arrow[l] & \boxed{\text{cccccc}} \arrow[l] \\
 & & \boxed{\text{master}} \arrow[u]
\end{tikzcd}$$

Local Repo After:

The questions on the following pages refer to the example figures. The figures show different aspects of the *beebopdb* web app that is a free and open online music database. Users can use the app to browse and manage data on music artists, albums, and tracks data.

12. [10pts] Draw a UML class diagram that represents the three model classes given in Figure 1. Be sure to label all associations and association ends, and include all multiplicities. Don't include any "id" attributes (including foreign keys). You may also omit the "datetime" attributes that Rails provides by default.

13. [6pts] Consider the model classes in Figure 1 and the fixtures in Figure 2. Using the lines of code in Figure 3, complete the following model test classes such that each model class has test for a valid instance of the class and such that each validation has a test which demonstrates that the validation catches an invalid value. You should fill all blanks and use all lines at least once. Some lines may be used more than once.

```
class ArtistTest < ActiveSupport::TestCase

________

________

________

________

________

________

________

________

________

________
```

```
class AlbumTest < ActiveSupport::TestCase

________

________

________

________

________

________

________

________

________

________
```

```
class TrackTest < ActiveSupport::TestCase

________

________

________

________

________
```

14. [9pts] Consider the Albums *index* page in Figure 4. Using the lines of code in Figure 6, reverse engineer the view code that produced this page. You should fill every blank and use all lines at least once. Some lines may be used more than once.

____
____
____
____
____
____
____
____
____
____
____
____
____
____
____
____
____
____
____
____
____
____
____
____

15. [3pts] It is possible to add an “Artist” column to the Albums *index* page by inserting two lines of code. What are the two lines of code, and where should they be inserted in your answer to the previous question?

16. [2pts] Which of the following routes corresponds to the form in Figure 5?

   a) `get '/albums/:id', to: 'albums#show', as: 'album'`

   b) `patch '/albums/:id', to: 'albums#update'`

   c) `post '/album', to: 'albums#create'`

   d) `get '/albums/:id/edit', to: 'albums#edit', as: 'edit_album'`

   e) `get '/albums', to: 'albums#index', as: 'albums'`

17. [2pts] Which of the following lines of code would the controller need to execute before rendering the form view from Figure 5?

   a) `@albums = Album.all`

   b) `@album = Album.new`

   c) `@album = Album.find(params[:id])`

   d)
```
@album = Album.new(params.require(:album).permit(
                   :title,:year_released, :genre, :artist_id))
```

   e) None of the above

18. [1pt] True or false? State-affecting controller actions (such as create, update, and destroy) should always render a view, which produces an HTTP response containing HTML for the browser to display.

    a) True

    b) False

19. [10pts] For each component below, give the corresponding letter from the Rails architectural diagram in Figure 7.

__________ Model

__________ Browser

__________ Tests

__________ Controller

__________ Migrations

__________ Internet

__________ Database

__________ View

__________ Router

Figures

```
# == Schema Information
#
# Table name: artists
#
#  id            :integer          not null, primary key
#  name          :string
#  year_founded  :integer
#  place_founded :string
#  about         :text
#  created_at    :datetime         not null
#  updated_at    :datetime         not null
#

class Artist < ApplicationRecord
    has_many :albums
    validates :year_founded, numericality: { less_than_or_equal_to: Date.today.year }
end
```

```
# == Schema Information
#
# Table name: albums
#
#  id            :integer          not null, primary key
#  title         :string
#  year_released :integer
#  genre         :string
#  artist_id     :integer
#  created_at    :datetime         not null
#  updated_at    :datetime         not null
#
# Indexes
#
#  index_albums_on_artist_id  (artist_id)
#

class Album < ApplicationRecord
  belongs_to :artist
  has_many :tracks
  validates :genre, inclusion: { in: ['Rock', 'R&B/HipHop', 'Pop', 'Country', 'Latin'] }
end
```

```
# == Schema Information
#
# Table name: tracks
#
#  id             :integer          not null, primary key
#  title          :string
#  track_number   :integer
#  length_seconds :integer
#  album_id       :integer
#  created_at     :datetime         not null
#  updated_at     :datetime         not null
#
# Indexes
#
#  index_tracks_on_album_id  (album_id)
#

class Track < ApplicationRecord
  belongs_to :album
end
```

Figure 1. Three model classes from the beebopdb app.

```
one:
  name: LCD Soundsystem
  year_founded: 2002
  place_founded: Brooklyn
  about: LCD Soundsystem is an American rock band from Brooklyn, New York City...

two:
  name: Arcade Fire
  year_founded: 2001
  place_founded: Montreal
  about: Arcade Fire is a Canadian indie rock band, consisting ...
```

```
one:
  title: This Is Happening
  year_released: 2010
  genre: Rock
  artist: one

two:
  title: The Suburbs
  year_released: 2010
  genre: Rock
  artist: two
```

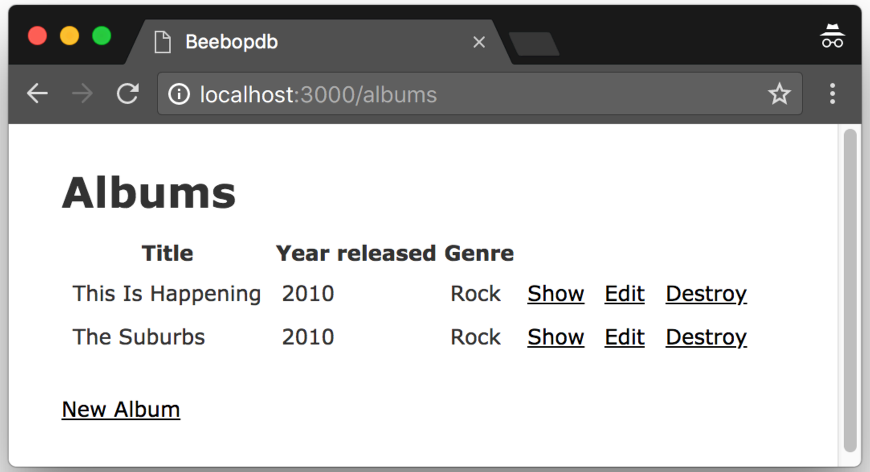
```
one:
  title: Dance Yrself Clean
  track_number: 1
  length_seconds: 536
  album: one

two:
  title: Ready to Start
  track_number: 2
  length_seconds: 255
  album: two
```

Figure 2. Test fixtures for the beebopdb model classes.

```
(a) end
(b) one.genre = 'INVALID'
(c) test "should be invalid genre" do
(d) one = tracks(:one)
(e) assert one.valid?
(f) test "should be valid artist" do
(g) one.year_founded = Date.today.year + 1
(h) test "should be valid album" do
(i) one = artists(:one)
(j) assert_not one.valid?
(k) test "should be valid track" do
(l) test "should be invalid year founded" do
(m) one = albums(:one)
```

Figure 3. Model unit test lines of code.

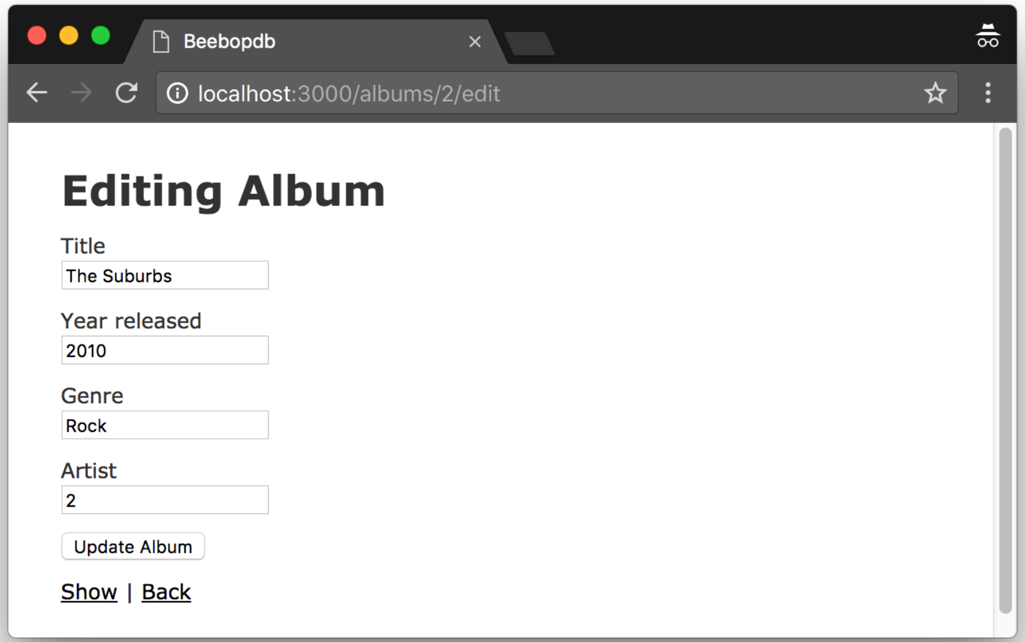


Albums

Title	Year released	Genre			
This Is Happening	2010	Rock	Show	Edit	Destroy
The Suburbs	2010	Rock	Show	Edit	Destroy

New Album

Figure 4. Albums *index* page.



Editing Album

Title	The Suburbs
Year released	2010
Genre	Rock
Artist	2

Figure 5. Form for updating an Album.

```
(a) <tbody>
(b) <table>
(c) <td><%= album.year_released %></td>
(d) <% @albums.each do |album| %>
(e) <% end %>
(f) </tbody>
(g) <tr>
(h) <td><%= link_to 'Show', album %></td>
(i) <%= link_to 'New Album', new_album_path %>
(j) </tr>
(k) <td><%= album.genre %></td>
(l) <th>Title</th>
(m) </table>
(n) <td><%= link_to 'Edit', edit_album_path(album) %></td>
(o) <th>Year released</th>
(p) <h1>Albums</h1>
(q) <td><%= link_to 'Destroy', album, method: :delete, data: { confirm: 'Are you sure?'
    } %></td>
(r) <td><%= album.title %></td>
(s) <th colspan="3"></th>
(t) <thead>
(u) </thead>
(v) <th>Genre</th>
```

Figure 6. Lines of ERB code for the Albums *index* page.

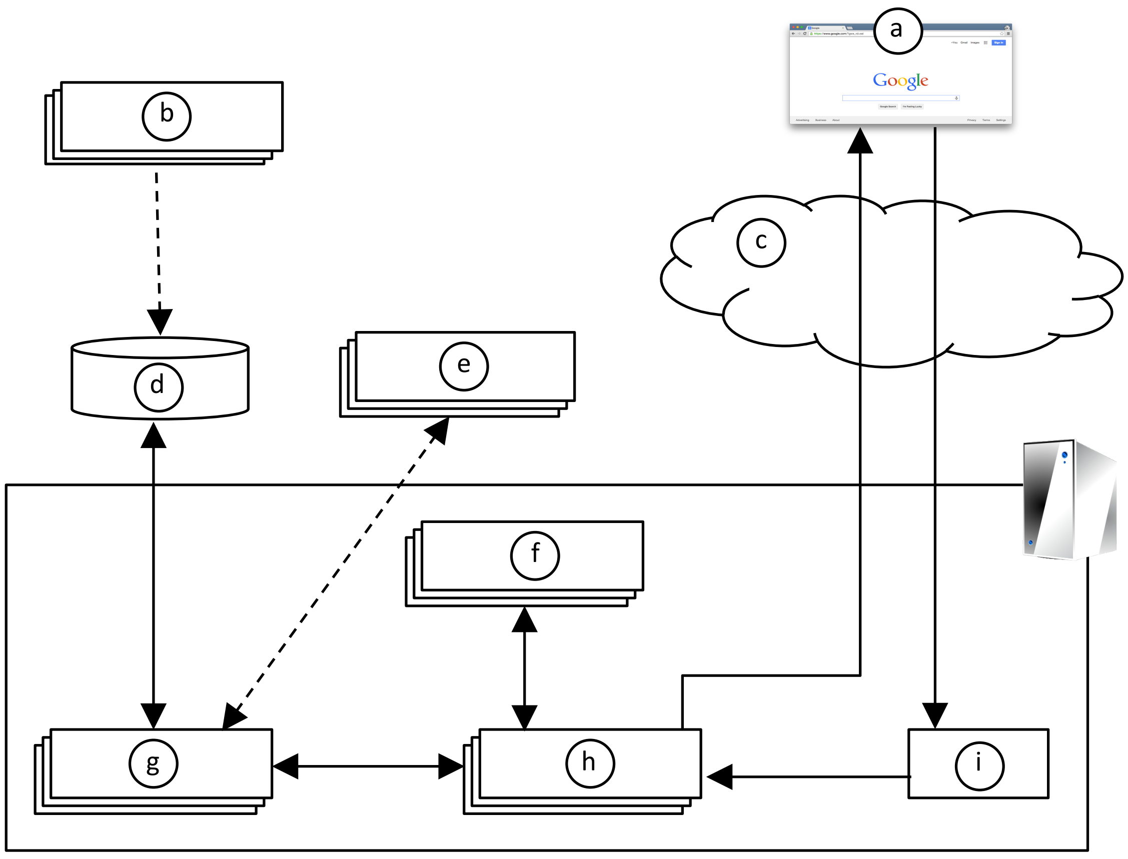


Figure 7. Rails architectural diagram.