ECE 495/595 Web Architectures/Cloud Computing – Homework #2

Due date: September 28, 2012, 11:59pm

1. **Using Git and GitHub.** This exercise will give you an opportunity to use Git and work with GitHub. The exercise involves sharing documents within a group. No programming is necessary, this exercise will only require the use of GitHub commands such as `fork`, `branch` and `merge`. (Uday, the class TA, will send out a message assigning you to a group, and he'll provide a list of the GitHub accounts of your group members.) Every person in your group will be required to create their own repository. Once you create the repository, add a text file. In the text file add the header "List of <your choice>", where <your choice> can be anything, NFL teams, baby names, breeds of dogs, whatever you want. Once you decide on a list of something, add one item to the list. For example,

   List of Websites Created with Ruby on Rails
   GitHub

   Once you have added one item, push your list to your GitHub repository. Can you guess what's next? Every member of the group will fork every other member's list, add one item, and send a merge request. Now, take everyone else's merge requests and add them to your project. When you are done, you should have a list of items. Submit the repository URL. Here is a quick rundown of how to accomplish this assignment: First, go to www.github.com/<username>. Now click the "New Repository" button and follow the prompts to create and name a new repository. The repository is now on GitHub, but you need to create a local copy on your machine. First, navigate to where you want the project to live on your machine, then run these commands:

   ```
   $ mkdir <repository-name>
   $ cd <repository-name>
   $ git init
   ```

   These three commands will create a directory and then initialize Git in it, allowing you to do Git commands. Now, create your text file as described above in the directory that you just created, giving it a creative name. Then execute the following commands to add the text file and commit it:

   ```
   $ git add <text-file-name>
   $ git commit -m '<commit-message>'
   ```

   Now, you need to add a remote and push to GitHub:

   ```
   $ git remote add origin https://github.com/<username>/<repository-name>.git
   $ git push origin master
   ```

   Congratulations! You just finished the first part of the assignment. Now you are going to need to fork everyone's assignment i your group. First, open a web browser and navigate to your teammate's repository and click the "fork button in the top right corner of the page. Run the following commands to pull the project into your local machine:

```
$ git clone https://github.com/<username>/<team-member-repository-name>.git
$ cd <team-member-repository-name>
```

Now, you can go ahead and add something to their list, and push it back up to your repository, and send them a merge request. All that is left to do is merge in your teammates' changes to your list. On the first merge, you can probably merge automatically. After that you are going to have to branch and merge. Go to the directory where you created your list, then run these commands:

```
$ git chekout -b <branch-name>
$ git pull <URL-of your-teammate's-forked-repo>
```

These commands will create a branch of your project, and switch you to it. It will also pull in his or her changes and allow you to check them out. You can check what branch you are on by typing:

```
$ git branch
```

Now, if you like the changes, add and commit them, and switch back to the master branch by typing

```
$ git checkout master
```

Now you can merge the two branches by running

```
$ git merge <branch-name>
```

Now, you can add, commit, and push the changes back to your repository. Since the branch is no longer needed, you can delete it.

```
$ git branch -d <branch-name>
```

All that is left to do is repeat these steps for the remaining members of your group.

2. **Ruby Programming.** Write a Ruby program that allows a user to create and maintain, in an a SQLite database, a list of "reminder" tasks along with their due dates and whether or not the task is complete. In order to understand how to use SQLite through the ActiveRecord class, an example is provided below:

```
#File: ar_test.rb
require "active_record"

#Adapter for the SQLite3
ActiveRecord::Base::establish_connection(:adapter => "sqlite3",
                  :database => "testdb.sqlite")

#Define database schema, and create database "people"
class PeopleTableScript < ActiveRecord::Migration
  def self.up
    create_table :people do |t|
      t.string  :first_name
      t.string  :last_name
    end
  end
```

```
  def self.down
    drop_table :people
  end
end

#Create the table that will be used in the database
PeopleTableScript.up

class Person < ActiveRecord::Base
end
```

Load this code using:

```
$ irb -r ./ar_test.rb
```

From the interactive Ruby prompt, you can now execute commands such as:

```
> Person.new(:first_name => "Frank", :last_name => "Furter").save
```

3. **Rails Programming.** Modify the blog application that was created in class, adding the following features:

   - Only the user who creates a blog post can edit or destroy the post.
   - Display the name of the author of a post and also the names of the commentators next to each post and comment.

   Upload your finished application to your pubic GitHub account, and provide a link in your homework report.