CSCE 606 Iteration 2
Team: Flyer

Customer meeting date to demo this iteration:

We tried to meet on 11/18/2016 Friday afternoon. However, our custom is not free. We'll try to schedule another time to demonstrate and discuss current work.

User stories implemented in this iteration:

Feature: Extracting the data set from public websites
As an administrator
So that I can automatically assign the data set to users
I want to provide the term for searching in ArrayExpress (https://www.ebi.ac.uk/arrayexpress/)

Feature: Assigning Questions with dataset
As an administrator
So that I can assign 1-5 Yes or No questions to a set of dataset
I want to ask user to show their curation results related to these questions

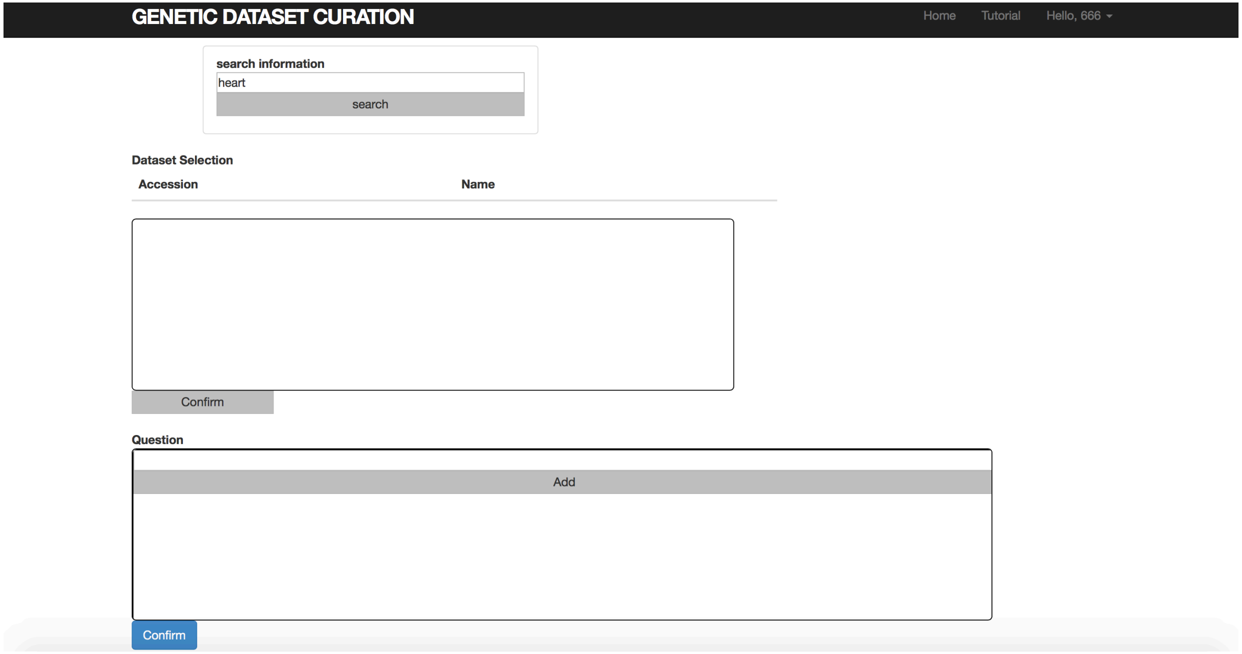
Feature: Assigning datasets to a certain group
As an administrator
So that I can assign datasets to a group of students
I want to choose which group the datasets are sent to.

Feature: Adding Groups Mechanism to the System
As a stakeholder (power admin),
So that I can manage groups(CRUD) and promote/demote users to become a group admin.
I want the group admin to be able to manage his/her own group, too.

Views

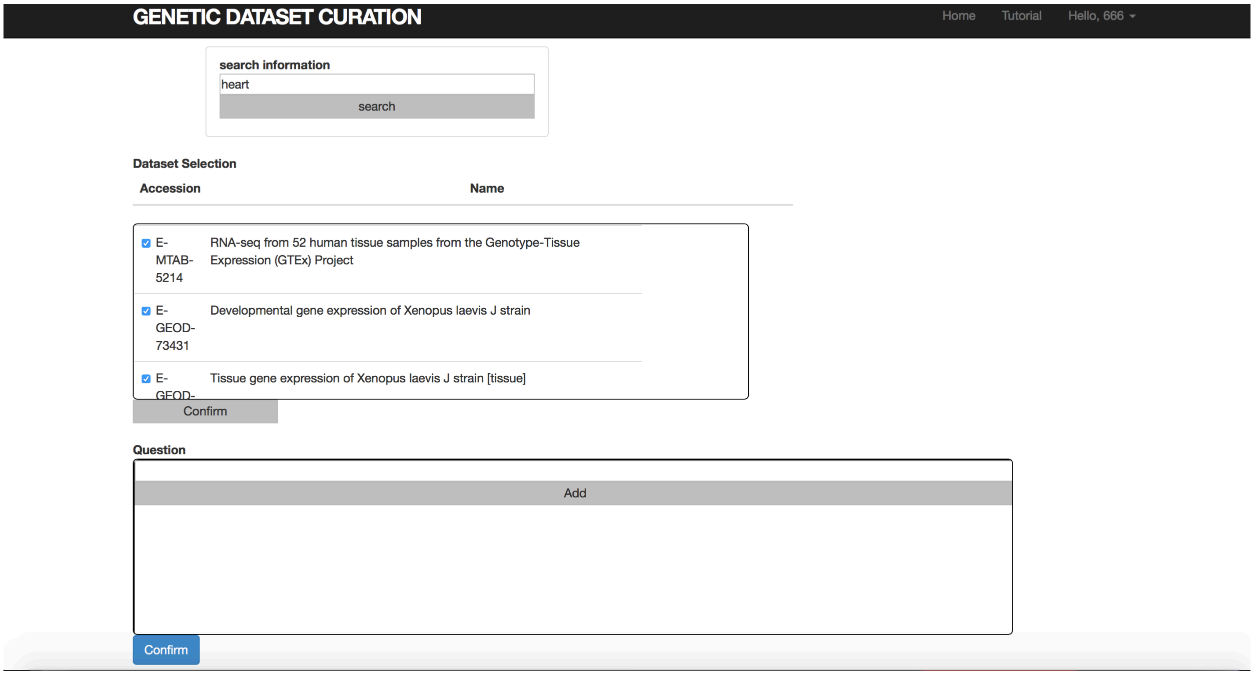
User view for searching dataset:

Search for keywords:



search information: heart

show search results:

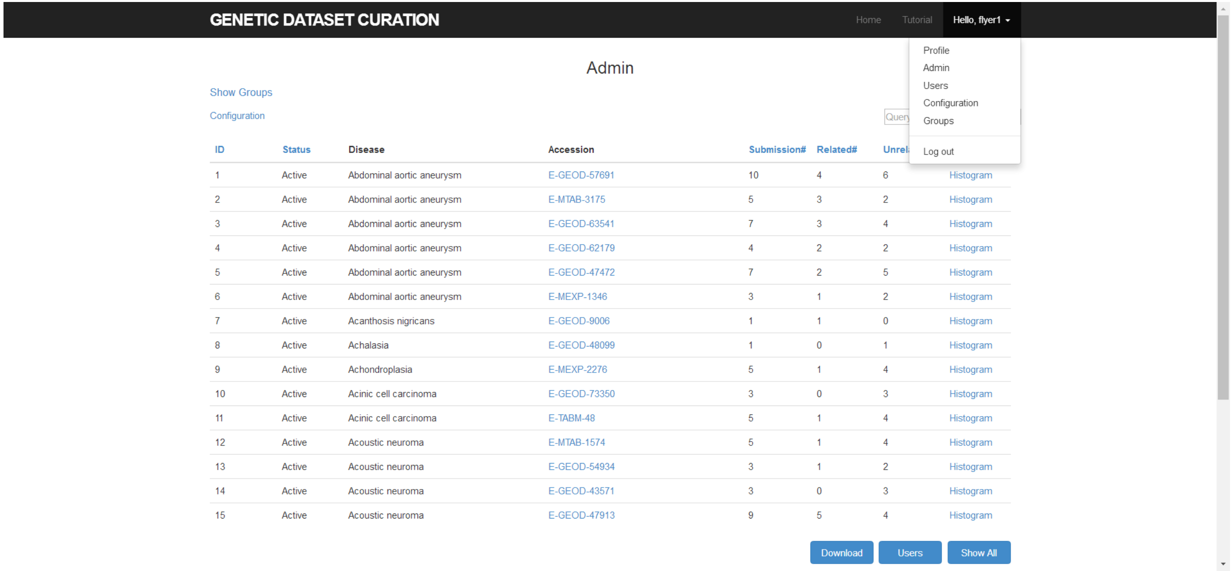


search information: heart

Accession	Name
☑ E-MTAB-5214	RNA-seq from 52 human tissue samples from the Genotype-Tissue Expression (GTEx) Project
☑ E-GEOD-73431	Developmental gene expression of Xenopus laevis J strain
☑ E-GEOD-	Tissue gene expression of Xenopus laevis J strain [tissue]

Group Function:

Links to the group index page (only available to admins):



Admin

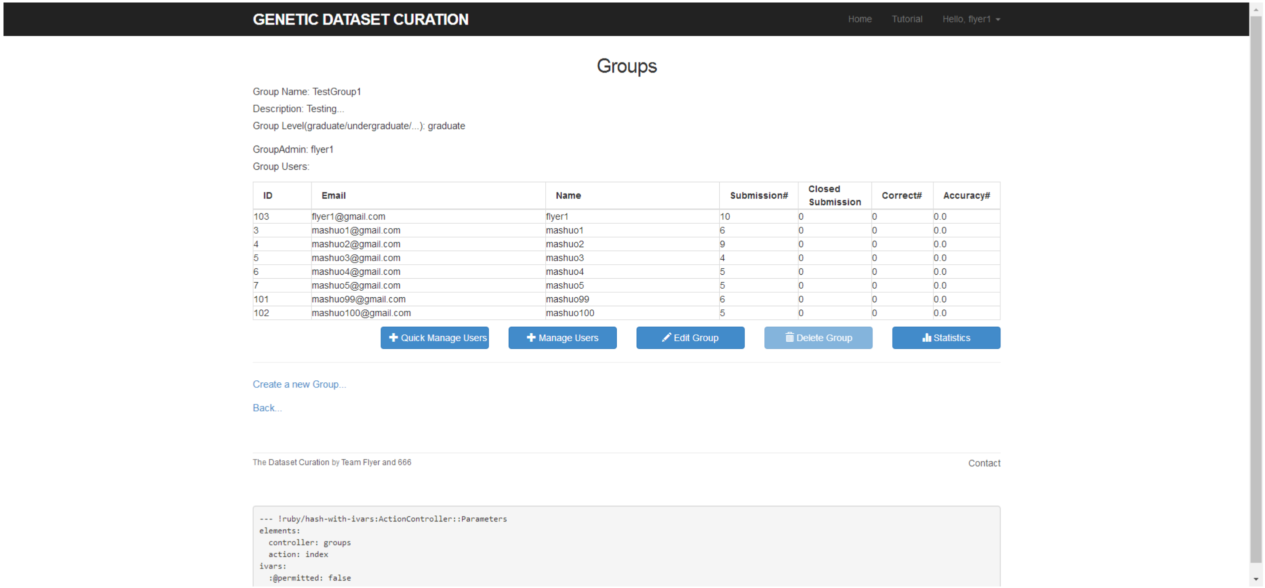
Show Groups

Configuration

ID	Status	Disease	Accession	Submission#	Related#	Unrel	
1	Active	Abdominal aortic aneurysm	E-GEOD-57691	10	4	6	Histogram
2	Active	Abdominal aortic aneurysm	E-MTAB-3175	5	3	2	Histogram
3	Active	Abdominal aortic aneurysm	E-GEOD-63541	7	3	4	Histogram
4	Active	Abdominal aortic aneurysm	E-GEOD-62179	4	2	2	Histogram
5	Active	Abdominal aortic aneurysm	E-GEOD-47472	7	2	5	Histogram
6	Active	Abdominal aortic aneurysm	E-MEXP-1346	3	1	2	Histogram
7	Active	Acanthosis nigricans	E-GEOD-9006	1	1	0	Histogram
8	Active	Achalasia	E-GEOD-48099	1	0	1	Histogram
9	Active	Achondroplasia	E-MEXP-2276	5	1	4	Histogram
10	Active	Acinic cell carcinoma	E-GEOD-73350	3	0	3	Histogram
11	Active	Acinic cell carcinoma	E-TABM-48	5	1	4	Histogram
12	Active	Acoustic neuroma	E-MTAB-1574	5	1	4	Histogram
13	Active	Acoustic neuroma	E-GEOD-54934	3	1	2	Histogram
14	Active	Acoustic neuroma	E-GEOD-43571	3	0	3	Histogram
15	Active	Acoustic neuroma	E-GEOD-47913	9	5	4	Histogram

Group index pages:

For group admins:



Groups

Group Name: TestGroup1

Description: Testing...

Group Level(graduate/undergraduate/...): graduate

GroupAdmin: flyer1

Group Users:

ID	Email	Name	Submission#	Closed Submission	Correct#	Accuracy#
103	flyer1@gmail.com	flyer1	10	0	0	0.0
3	mashuo1@gmail.com	mashuo1	6	0	0	0.0
4	mashuo2@gmail.com	mashuo2	9	0	0	0.0
5	mashuo3@gmail.com	mashuo3	4	0	0	0.0
6	mashuo4@gmail.com	mashuo4	5	0	0	0.0
7	mashuo5@gmail.com	mashuo5	5	0	0	0.0
101	mashuo99@gmail.com	mashuo99	6	0	0	0.0
102	mashuo100@gmail.com	mashuo100	5	0	0	0.0

Create a new Group...

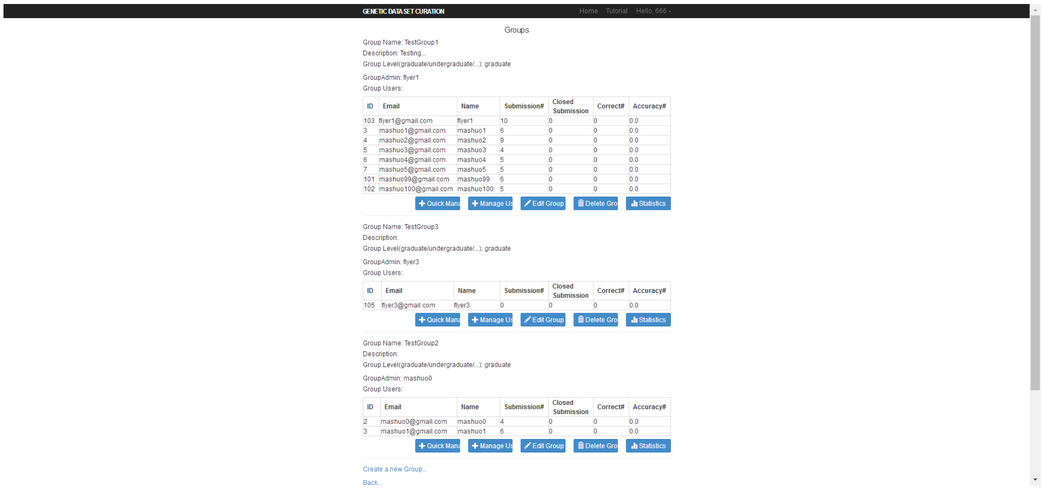
Back...

The Dataset Curation by Team Flyer and 666

Contact

```
--- !ruby/hash-with-ivars:ActionController::Parameters
elements:
  controller: groups
  action: index
ivars:
  :@permitted: false
```

For power admins:



Groups

Group Name: TestGroup1

Description: Testing...

Group Level(graduate/undergraduate/...): graduate

GroupAdmin: flyer1

Group Users:

ID	Email	Name	Submission#	Closed Submission	Correct#	Accuracy#
103	flyer1@gmail.com	flyer1	10	0	0	0.0
3	mashuo1@gmail.com	mashuo1	6	0	0	0.0
4	mashuo2@gmail.com	mashuo2	9	0	0	0.0
5	mashuo3@gmail.com	mashuo3	4	0	0	0.0
6	mashuo4@gmail.com	mashuo4	5	0	0	0.0
7	mashuo5@gmail.com	mashuo5	5	0	0	0.0
101	mashuo99@gmail.com	mashuo99	6	0	0	0.0
102	mashuo100@gmail.com	mashuo100	5	0	0	0.0

Group Name: TestGroup3

Description:

Group Level(graduate/undergraduate/...): graduate

GroupAdmin: flyer3

Group Users:

ID	Email	Name	Submission#	Closed Submission	Correct#	Accuracy#
105	flyer3@gmail.com	flyer3	0	0	0	0.0

Group Name: TestGroup2

Description:

Group Level(graduate/undergraduate/...): graduate

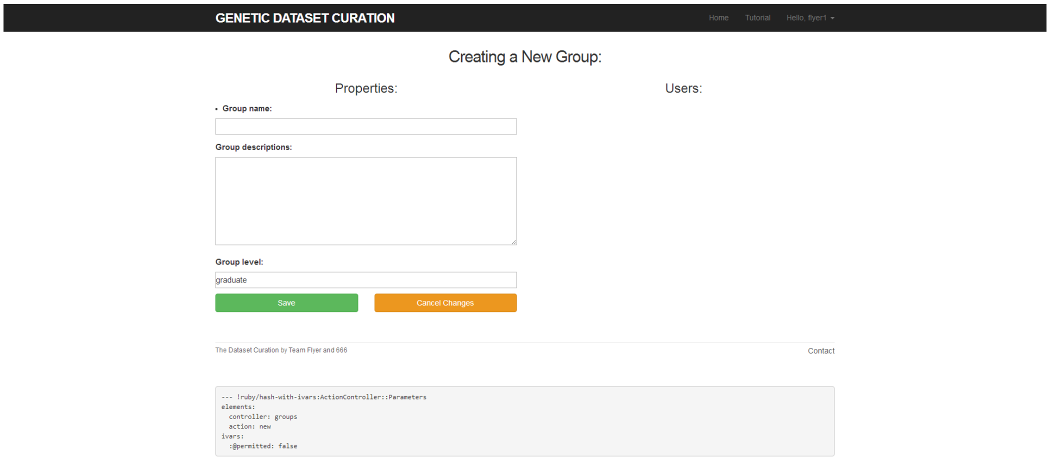
GroupAdmin: mashuo0

Group Users:

ID	Email	Name	Submission#	Closed Submission	Correct#	Accuracy#
2	mashuo0@gmail.com	mashuo0	4	0	0	0.0
3	mashuo1@gmail.com	mashuo1	6	0	0	0.0

Groups CRUD pages:

Create:



Creating a New Group:

Properties:

Users:

- Group name:

Group descriptions:

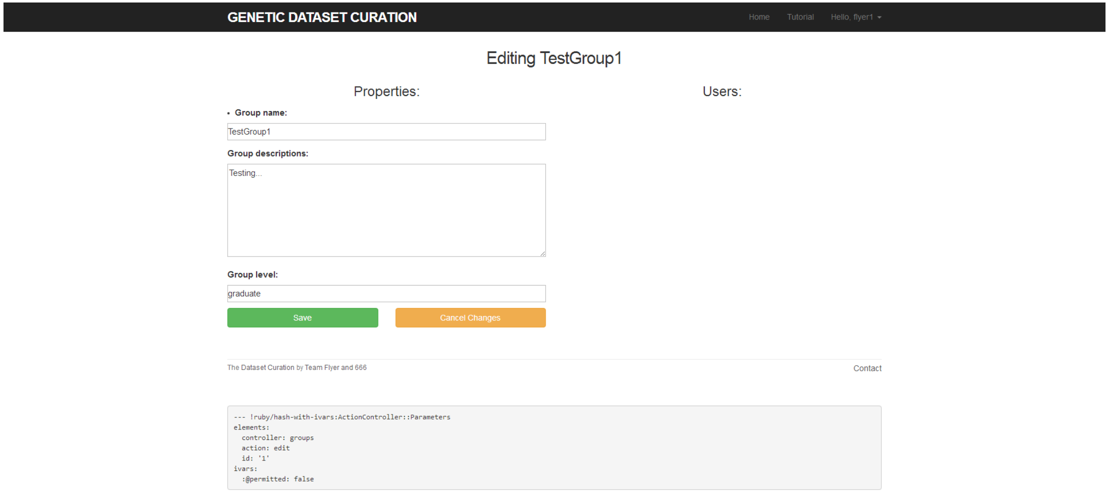
Group level:

graduate

```
--- !ruby/hash-with-ivars:ActionController::Parameters
elements:
  controller: groups
  action: new
ivars:
  :@permitted: false
```

The right half is reserved for future development.

Update:



Editing TestGroup1

Properties:

Users:

- Group name:

TestGroup1

Group descriptions:

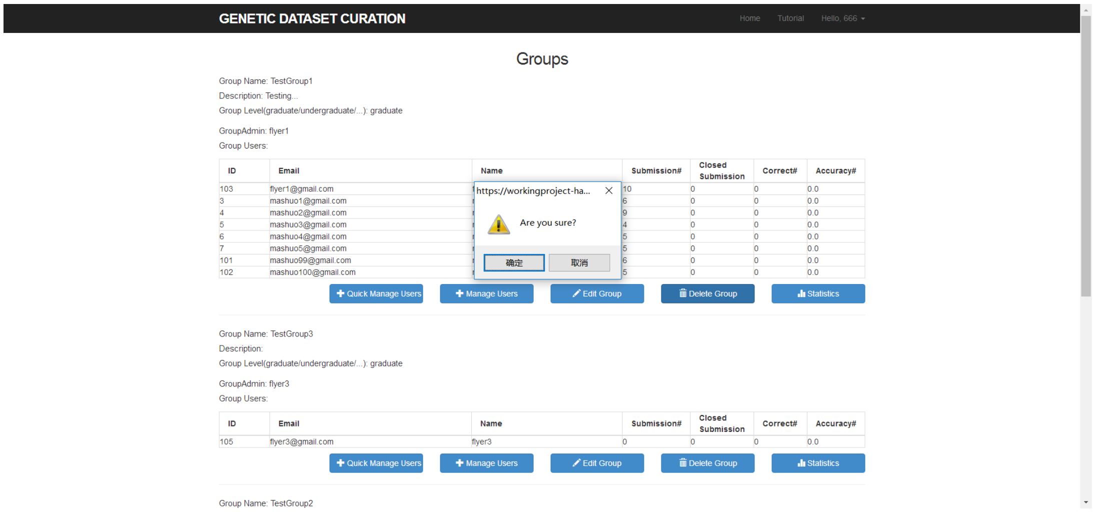
Testing...

Group level:

graduate

```
--- !ruby/hash-with-ivars:ActionController::Parameters
elements:
  controller: groups
  action: edit
  id: '1'
ivars:
  :@permitted: false
```

Delete:



Groups

Group Name: TestGroup1
Description: Testing...
Group Level(graduate/undergraduate/...): graduate

GroupAdmin: flyer1
Group Users:

ID	Email	Name	Submission#	Closed Submission	Correct#	Accuracy#
103	flyer1@gmail.com		10	0	0	0.0
3	mashuo1@gmail.com		6	0	0	0.0
4	mashuo2@gmail.com		9	0	0	0.0
5	mashuo3@gmail.com		4	0	0	0.0
6	mashuo4@gmail.com		5	0	0	0.0
7	mashuo5@gmail.com		5	0	0	0.0
101	mashuo99@gmail.com		6	0	0	0.0
102	mashuo100@gmail.com		5	0	0	0.0

https://workingproject-ha...

Are you sure?

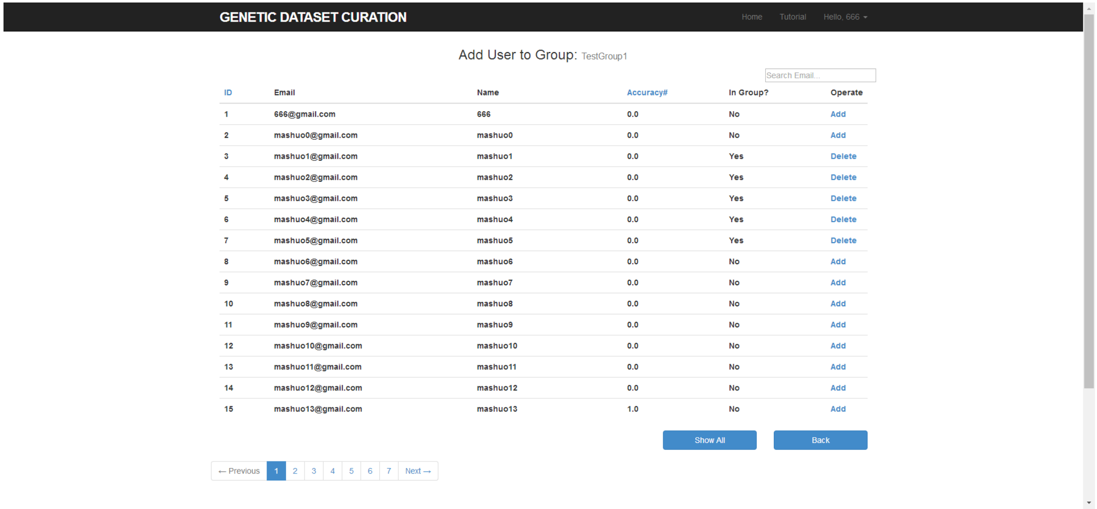
Group Name: TestGroup3
Description:
Group Level(graduate/undergraduate/...): graduate

GroupAdmin: flyer3
Group Users:

ID	Email	Name	Submission#	Closed Submission	Correct#	Accuracy#
105	flyer3@gmail.com	flyer3	0	0	0	0.0

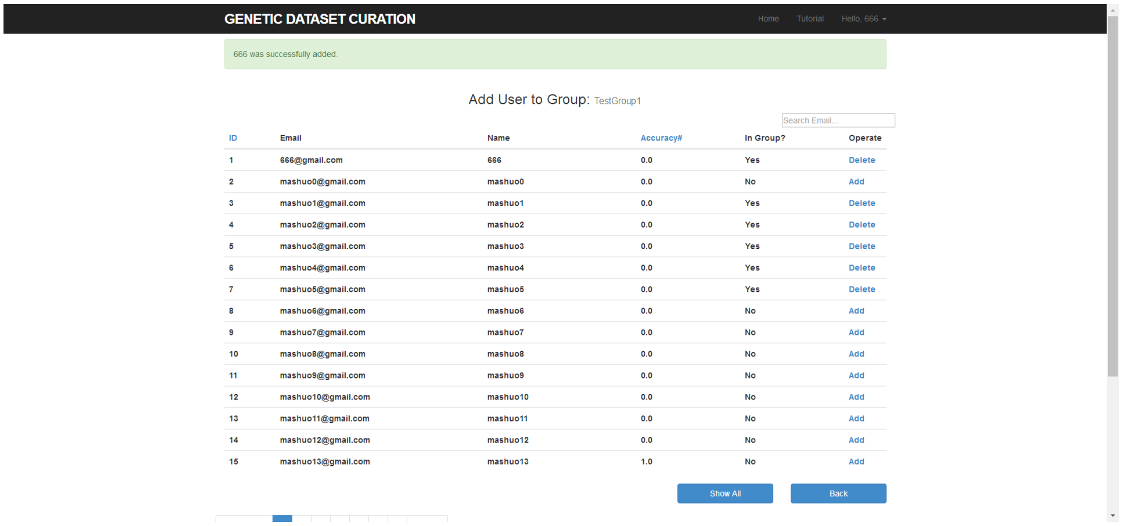
Group Name: TestGroup2

Add users to group with detailed user information:



Add User to Group: TestGroup1

ID	Email	Name	Accuracy#	In Group?	Operate
1	666@gmail.com	666	0.0	No	Add
2	mashuo0@gmail.com	mashuo0	0.0	No	Add
3	mashuo1@gmail.com	mashuo1	0.0	Yes	Delete
4	mashuo2@gmail.com	mashuo2	0.0	Yes	Delete
5	mashuo3@gmail.com	mashuo3	0.0	Yes	Delete
6	mashuo4@gmail.com	mashuo4	0.0	Yes	Delete
7	mashuo5@gmail.com	mashuo5	0.0	Yes	Delete
8	mashuo6@gmail.com	mashuo6	0.0	No	Add
9	mashuo7@gmail.com	mashuo7	0.0	No	Add
10	mashuo8@gmail.com	mashuo8	0.0	No	Add
11	mashuo9@gmail.com	mashuo9	0.0	No	Add
12	mashuo10@gmail.com	mashuo10	0.0	No	Add
13	mashuo11@gmail.com	mashuo11	0.0	No	Add
14	mashuo12@gmail.com	mashuo12	0.0	No	Add
15	mashuo13@gmail.com	mashuo13	1.0	No	Add

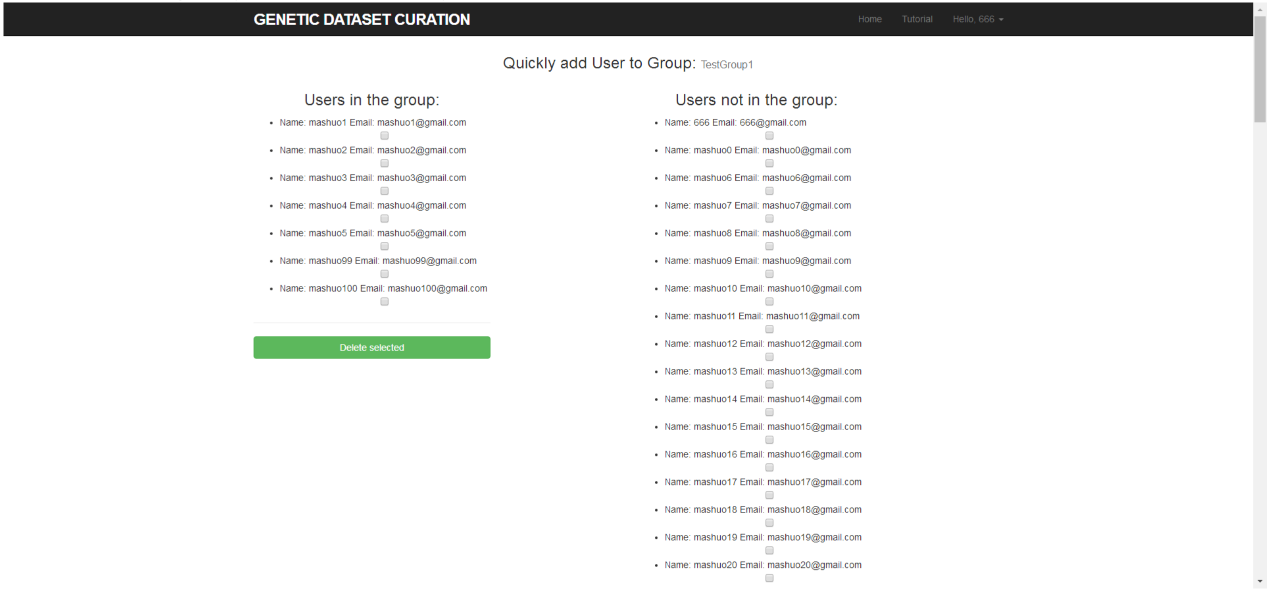


666 was successfully added.

Add User to Group: TestGroup1

ID	Email	Name	Accuracy#	In Group?	Operate
1	666@gmail.com	666	0.0	Yes	Delete
2	mashuo0@gmail.com	mashuo0	0.0	No	Add
3	mashuo1@gmail.com	mashuo1	0.0	Yes	Delete
4	mashuo2@gmail.com	mashuo2	0.0	Yes	Delete
5	mashuo3@gmail.com	mashuo3	0.0	Yes	Delete
6	mashuo4@gmail.com	mashuo4	0.0	Yes	Delete
7	mashuo5@gmail.com	mashuo5	0.0	Yes	Delete
8	mashuo6@gmail.com	mashuo6	0.0	No	Add
9	mashuo7@gmail.com	mashuo7	0.0	No	Add
10	mashuo8@gmail.com	mashuo8	0.0	No	Add
11	mashuo9@gmail.com	mashuo9	0.0	No	Add
12	mashuo10@gmail.com	mashuo10	0.0	No	Add
13	mashuo11@gmail.com	mashuo11	0.0	No	Add
14	mashuo12@gmail.com	mashuo12	0.0	No	Add
15	mashuo13@gmail.com	mashuo13	1.0	No	Add

Add users to group simultaneously with concise user information:



Quickly add User to Group: TestGroup1

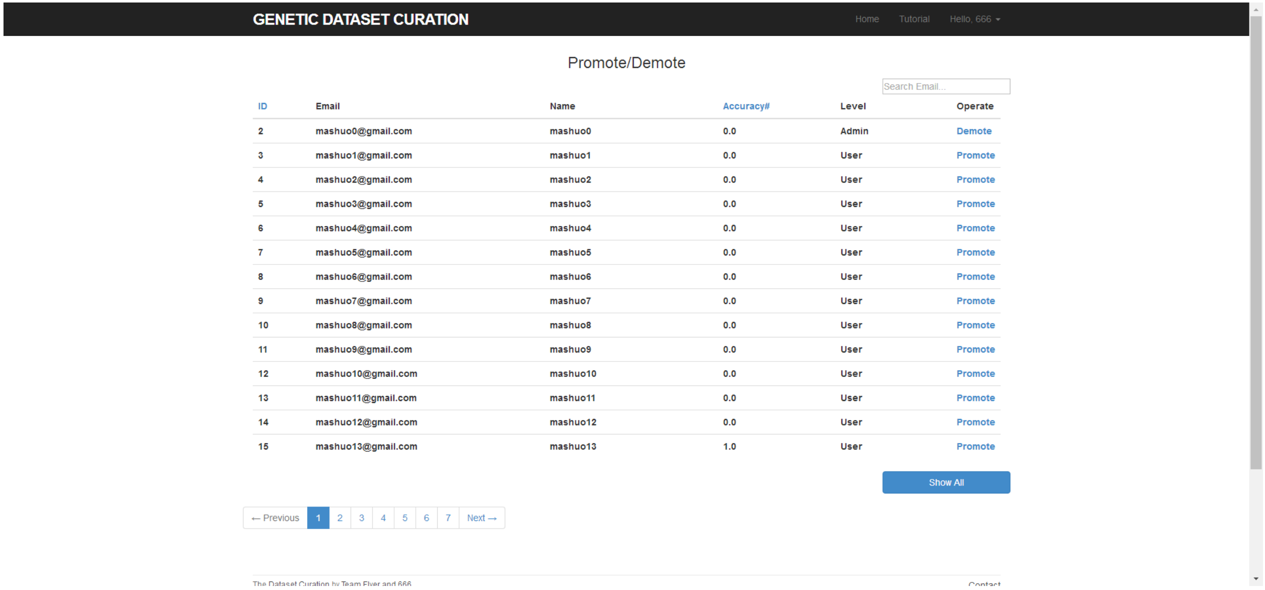
Users in the group:

- Name: mashuo1 Email: mashuo1@gmail.com ☐
- Name: mashuo2 Email: mashuo2@gmail.com ☐
- Name: mashuo3 Email: mashuo3@gmail.com ☐
- Name: mashuo4 Email: mashuo4@gmail.com ☐
- Name: mashuo5 Email: mashuo5@gmail.com ☐
- Name: mashuo99 Email: mashuo99@gmail.com ☐
- Name: mashuo100 Email: mashuo100@gmail.com ☐

Users not in the group:

- Name: 666 Email: 666@gmail.com ☐
- Name: mashuo0 Email: mashuo0@gmail.com ☐
- Name: mashuo6 Email: mashuo6@gmail.com ☐
- Name: mashuo7 Email: mashuo7@gmail.com ☐
- Name: mashuo8 Email: mashuo8@gmail.com ☐
- Name: mashuo9 Email: mashuo9@gmail.com ☐
- Name: mashuo10 Email: mashuo10@gmail.com ☐
- Name: mashuo11 Email: mashuo11@gmail.com ☐
- Name: mashuo12 Email: mashuo12@gmail.com ☐
- Name: mashuo13 Email: mashuo13@gmail.com ☐
- Name: mashuo14 Email: mashuo14@gmail.com ☐
- Name: mashuo15 Email: mashuo15@gmail.com ☐
- Name: mashuo16 Email: mashuo16@gmail.com ☐
- Name: mashuo17 Email: mashuo17@gmail.com ☐
- Name: mashuo18 Email: mashuo18@gmail.com ☐
- Name: mashuo19 Email: mashuo19@gmail.com ☐
- Name: mashuo20 Email: mashuo20@gmail.com ☐

Promote user to be a group admin (only power admin can do this):



Promote/Demote

ID	Email	Name	Accuracy#	Level	Operate
2	mashuo0@gmail.com	mashuo0	0.0	Admin	Demote
3	mashuo1@gmail.com	mashuo1	0.0	User	Promote
4	mashuo2@gmail.com	mashuo2	0.0	User	Promote
5	mashuo3@gmail.com	mashuo3	0.0	User	Promote
6	mashuo4@gmail.com	mashuo4	0.0	User	Promote
7	mashuo5@gmail.com	mashuo5	0.0	User	Promote
8	mashuo6@gmail.com	mashuo6	0.0	User	Promote
9	mashuo7@gmail.com	mashuo7	0.0	User	Promote
10	mashuo8@gmail.com	mashuo8	0.0	User	Promote
11	mashuo9@gmail.com	mashuo9	0.0	User	Promote
12	mashuo10@gmail.com	mashuo10	0.0	User	Promote
13	mashuo11@gmail.com	mashuo11	0.0	User	Promote
14	mashuo12@gmail.com	mashuo12	0.0	User	Promote
15	mashuo13@gmail.com	mashuo13	1.0	User	Promote

The Dataset Curation by Team Flyer and 666

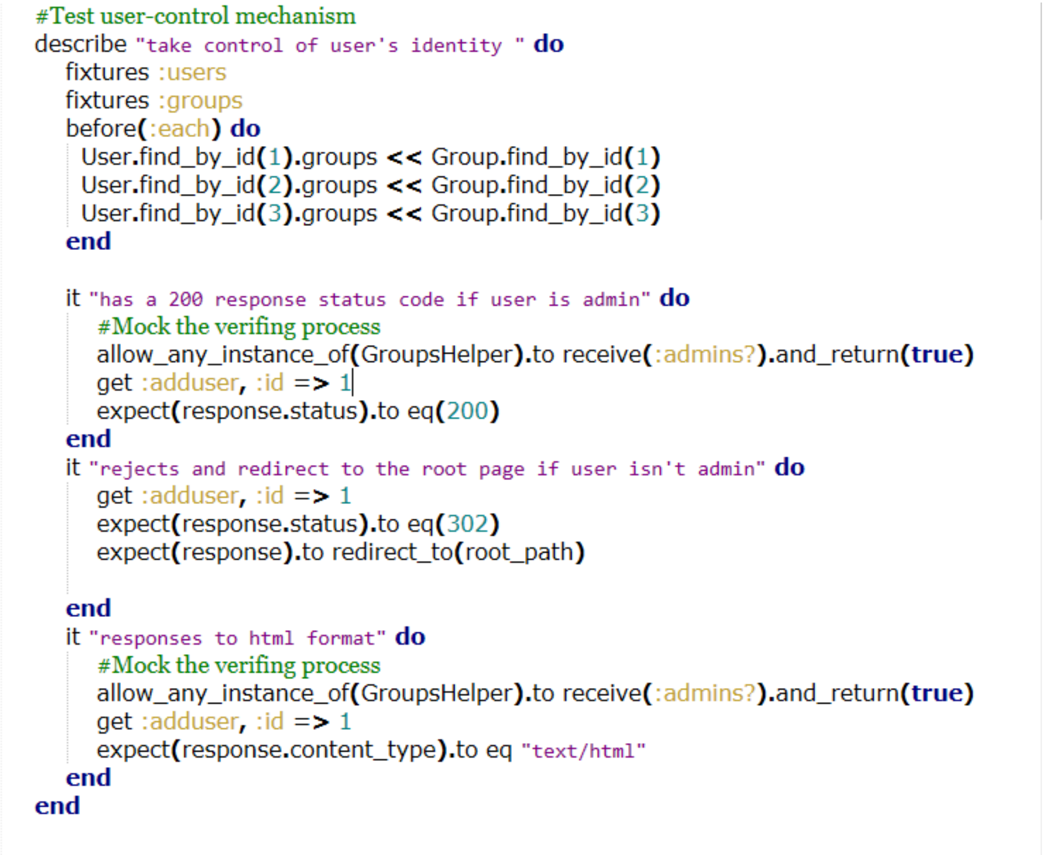
For now, basic functions are achieved.

TDD test

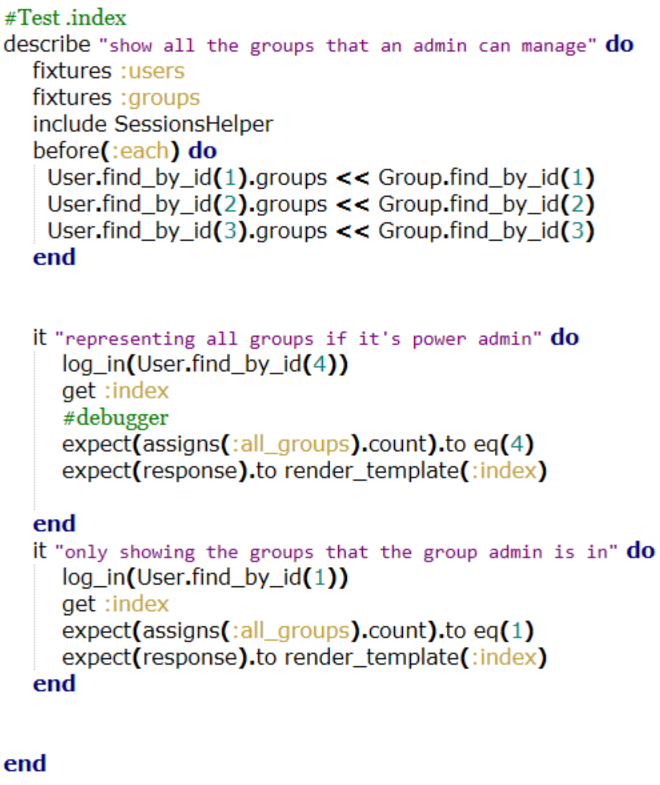
Group Model test:

```
describe Group do
    #Test function .get_users
    describe "finding users of the group" do
        fixtures :users
        fixtures :groups
        before(:each) do
          User.find_by_id(1).groups << Group.find_by_id(1)
          User.find_by_id(2).groups << Group.find_by_id(2)
          User.find_by_id(3).groups << Group.find_by_id(3)
        end
        #Happy Path
        it "returns the group users if there are users found." do
            @grp1=Group.find_by_id(1)
            @has_user = false
            @grp1.get_users.each do |user|
            #ActiveRecord::Associations::CollectionProxy
                user.groups.to_a.each do |grp|
                    #Test if all the returned user's group name is "G1"
                    #which is desired.
                    if(grp.name == "G1")
                        @has_user = true
                    else
                        @has_user = false
                    end
                end
                expect(@has_user).to eq(true)
            end
        end
        #Sad Path
        it "returns empty collection set if no user in the group" do
            @grp4=Group.find_by_id(4)
            expect(@grp4.get_users.count).to eq(0)
        end
    end
  #Test function .get_admins
  describe "finding admins of the group" do
      fixtures :users
      fixtures :groups
      before(:each) do
        User.find_by_id(1).groups << Group.find_by_id(1)
        User.find_by_id(2).groups << Group.find_by_id(2)
        User.find_by_id(3).groups << Group.find_by_id(3)
      end
      #Happy Path
      it "returns the group admins if there are such admins. For now,
          we assume there's only one group_admin in each group." do
          @grp1=Group.find_by_id(1)
          @has_user = false
          expect(@grp1.get_admins.count).to eq(1)
          expect(@grp1.get_admins.first).to eq(User.find_by_id(@grp1.admin_uid).name)
          #ActiveRecord::Associations::CollectionProxy
      end
      #Sad Path
      it "returns empty collection set if no such admin in the group" do
          @grp4=Group.find_by_id(4)
          expect(@grp4.get_admins.count).to eq(0)
      end
  end
```

Group Controller test:



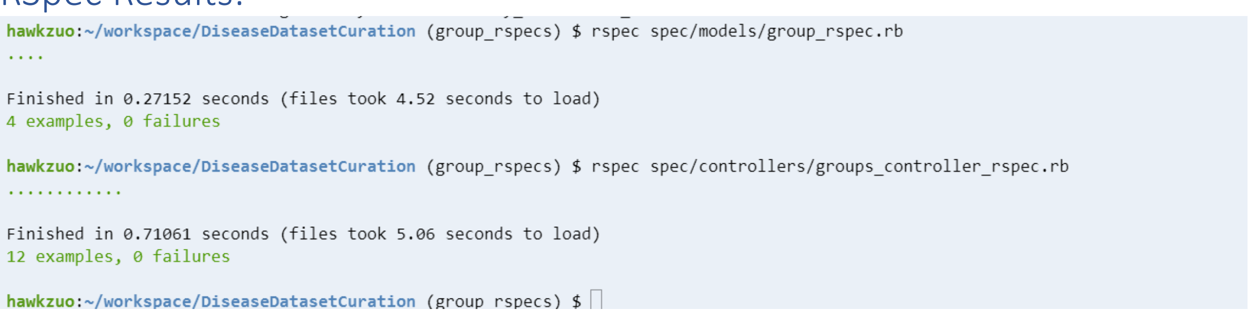
```
#Test user-control mechanism
describe "take control of user's identity " do
  fixtures :users
  fixtures :groups
  before(:each) do
   User.find_by_id(1).groups << Group.find_by_id(1)
   User.find_by_id(2).groups << Group.find_by_id(2)
   User.find_by_id(3).groups << Group.find_by_id(3)
  end

  it "has a 200 response status code if user is admin" do
    #Mock the verifing process
    allow_any_instance_of(GroupsHelper).to receive(:admins?).and_return(true)
    get :adduser, :id => 1
    expect(response.status).to eq(200)
  end
  it "rejects and redirect to the root page if user isn't admin" do
    get :adduser, :id => 1
    expect(response.status).to eq(302)
    expect(response).to redirect_to(root_path)

  end
  it "responses to html format" do
    #Mock the verifing process
    allow_any_instance_of(GroupsHelper).to receive(:admins?).and_return(true)
    get :adduser, :id => 1
    expect(response.content_type).to eq "text/html"
  end
end
```



```
#Test .index
describe "show all the groups that an admin can manage" do
  fixtures :users
  fixtures :groups
  include SessionsHelper
  before(:each) do
   User.find_by_id(1).groups << Group.find_by_id(1)
   User.find_by_id(2).groups << Group.find_by_id(2)
   User.find_by_id(3).groups << Group.find_by_id(3)
  end


  it "representing all groups if it's power admin" do
    log_in(User.find_by_id(4))
    get :index
    #debugger
    expect(assigns(:all_groups).count).to eq(4)
    expect(response).to render_template(:index)

  end
  it "only showing the groups that the group admin is in" do
    log_in(User.find_by_id(1))
    get :index
    expect(assigns(:all_groups).count).to eq(1)
    expect(response).to render_template(:index)
  end


end
```

```
#.new/.create/.destroy/.edit/.update
describe "Admins can create a new group" do
  fixtures :users
  fixtures :groups
  include SessionsHelper
  before(:each) do
   User.find_by_id(1).groups << Group.find_by_id(1)
   User.find_by_id(2).groups << Group.find_by_id(2)
   User.find_by_id(3).groups << Group.find_by_id(3)
  end
  it "should have a controller method and have
       a instance variable to handle this request" do
    log_in(User.find_by_id(1))
    get :new
    expect(assigns(:group).nil?).to be_falsy
    expect(assigns(:group)).to be_a_new(Group)
    expect(response).to render_template(:new)
  end

  it "should create a new group with correct infos" do
    log_in(User.find_by_id(4))
    post :create, { :group => { :name => "New Group" } }
    expect(assigns(:group).save).to eq(true)
    expect(response).to redirect_to(groups_path)
  end

  it "should reject creating if there's something wrong" do
    log_in(User.find_by_id(4))
    post :create, { :group => { :name => nil } }
    expect(assigns(:group).save).to eq(false)
    expect(response).to render_template(:new)
  end

  it "should delete an exisiting group with confirmation" do
    log_in(User.find_by_id(4))
    delete :destroy, {:id => 1}
    expect(Group.count).to eq(3)
  end

  it "should be able to edit exisiting groups" do
    log_in(User.find_by_id(4))
    get  :edit,{:id => 1}
    expect(assigns(:group)).to eql(Group.find(1))
    expect(response).to render_template(:edit)
  end

  it "should be able to update with correct infos" do
    log_in(User.find_by_id(4))
    post :update, {:id => 1, :group => { :name => "New Group" }}
    expect(Group.find(1).name).to eql("New Group")
    expect(response).to redirect_to(groups_path)
  end
  it "should be redirected to editing page if there's something wrong." do
    log_in(User.find_by_id(4))
    post :update, {:id => 1, :group => { :name => nil }}
    expect(Group.find(1).name).to eql("G1")
    expect(response).to render_template(:edit)
  end
end
```

RSpec Results:



```
hawkzuo:~/workspace/DiseaseDatasetCuration (group_rspecs) $ rspec spec/models/group_rspec.rb
....

Finished in 0.27152 seconds (files took 4.52 seconds to load)
4 examples, 0 failures

hawkzuo:~/workspace/DiseaseDatasetCuration (group_rspecs) $ rspec spec/controllers/groups_controller_rspec.rb
............

Finished in 0.71061 seconds (files took 5.06 seconds to load)
12 examples, 0 failures

hawkzuo:~/workspace/DiseaseDatasetCuration (group rspecs) $ 
```

Pivotal Tracker, Github repo, and Heroku links:

Pivotal:	https://www.pivotaltracker.com/n/projects/1887551
Github:	https://github.com/xvxiaopei/CSCE606-project
Heroku:	http://disease.herokuapp.com/