MIT AITI
Problem Set 3 – Django
Due: Jun 28, 2012 4pm

1. Connecting to the Postgresql Database

We have created user accounts for you on the Postgresql database server hosted on the class server machine (it has this ip address: 10.50.27.31). If your family/last name is 'Castonguay', your user name for logging in to the database server is CASTONGUAY. If you have a two-word family/last name such as 'Dela Cruz', your user name will be DELA_CRUZ. We have created a dedicated database for each of you and we have given it the same name as your login name. We have given you privileges to create your own database(s) for the use of your final project. Please don't abuse this, and we ask you to **limit the number of databases per person to three**.

If your user name is CASTONGUAY, you can login to the database server as follows:
psql –h 10.50.27.31 –U CASTONGUAY –d CASTONGUAY

Once you login, you will see the psql command interface. You can execute any postgresql commands in that interface.

It is beyond the scope of this course to introduce you to database technologies, and we strongly suggest that you check out this cheat sheet for some of the useful commands you might have to use in creating and manipulating data with SQL:
http://www.petefreitag.com/cheatsheets/postgresql/

2. FriendBook: Creating the bare essentials

In this exercise you will create a mini social networking website called "FriendBook" using Django.

2.1 Create a new django project using this command:
$django-admin.py startproject friendbook

2.2 Change the friendbook/settings.py to connect to the database. Your database settings should be changed as follows:

```
DATABASES = {
   'default': {
      'ENGINE': 'django.db.backends.postgresql_psycopg2',
      'NAME': ' CASTONGUAY ',
      'USER': ' CASTONGUAY ',
      'PASSWORD': '',
      'HOST': '10.50.27.31',
```

```
        'PORT': '5432',
    }
}
```

2.3 Create and app called 'friends' by typing this command.
$ python manage.py startapp friends

If you successfully created the friends app, you should see the following directory/file structure inside friendbook:

```
friendbook
|__ __init__.py
|__ manage.py
|__ settings.py
|__ urls.py
|__ friends
    |__ __init__.py
    |__ models.py
    |__ tests.py
    |__ views.py
```

2.4 Sync the database and create a superuser by running:
$ python manage.py syncdb

2.5. Test the barebones website by running:
$ python manage.py runserver
and pointing your browser to http://localhost:8000

2.6. The Django admin site is not activated by default – it's an opt-in thing. To activate the admin site for your installation, do these three things:

- Uncomment "django.contrib.admin" in the INSTALLED_APPS setting.
- Run python manage.py syncdb. Since you have added a new application to INSTALLED_APPS, the database tables need to be updated.
- Edit your friendbook/urls.py file and uncomment the lines that reference the admin – there are three lines in total to uncomment

Login to the django admin site with your superuser credentials. The admin site can be found at: http://localhost:8000/admin

2.7 Create few users for your social network using the admin interface.

3. FriendBook: Models

3.1 All social networks have users. Luckily for us, we do not have to create users from scratch. There's very good built in support in django for users. Read the documentation at: https://docs.djangoproject.com/en/1.4/topics/auth/#overview to learn more about the User model in django. We encourage you to use this User

model in your social network application.
What module do you need to import to make Users available in your code?

3.2 We use the models..py file to define the relationships between our Users. The very basic relationship in a social network is 'follow'/'friend' relationship that signals there's a link between UserA and UserB. If you are not sure how to create a model in django refer to the documentation at: https://docs.djangoproject.com/en/1.4/topics/db/models

Create the model UserLink in models.py. This model should have the following fields:

- from_user
- to_user
- date_added

3.2 Add the appropriate method in the UserLink model to display the message "UserA is following UserB". If you are not sure how to do this, refer to: https://docs.djangoproject.com/en/1.4/ref/models/instances/#model-instance-methods

3.3 What should happen if UserA tries to follow UserA? This doesn't make sense as people don't usually follow /friend themselves in social networks! Write code to raise an error if this were to happen in your app.
Hint: you have to override the save() method of the Model class in your UserLink model.
Please see https://docs.djangoproject.com/en/dev/topics/db/models/#overriding-predefined-model-methods to get some inspiration on how to do this.

3.4 The friend relationship between the 'to_user' and 'from_user' should be unique when considered together in the data model. So, add a constraint on these two fields. Check https://docs.djangoproject.com/en/dev/ref/models/options for more information on this.

3.5 Use the admin site to play with the data models you have created.