Program 1: Bank app  
Due: March 6, 2017

You are tasked with creating a Bank application.

Points applicable throughout the bank app:

* A user can have multiple accounts to manage money.
* User needs an admin’s authorization to deposit money.
* User needs an admin’s authorization to withdraw money if  the transaction amount is greater than $1000 from his/her account.
* User should be able to search through the list of users ( by name or email address) and add them as friends.
* User can transfer money between his/her accounts and to a friend’s account without permission from an admin.
* An admin has the authorization to approve or decline any pending transactions.

There will be two types of users in the system:

* Admin
* User

**Admins**

The system should have a preconfigured Admin, which cannot be deleted or modified by other admins.

Any admin user will have the following attributes: email (unique for each admin), name and password.

All admins can perform the following tasks:

* Log in with an email and password
* Edit her/his profile details
* Manage admins
  + Create new admins
  + View the list of all the admins and their profile details (except password)
  + Delete admins (except herself/himself and the preconfigured Admin)
* Manage account
  + Create an account for a user. An account will have the following attributes: Account Number (unique for each account, 9-digit number), status, balance
    - Possible values for the status attribute:
      * Active - all the transaction operations, deposit, withdraw, send and borrow are available.
      * Closed - All the transaction operations are unavailable.
  + View the list of all the accounts. View the balance for the accounts
  + View the details of an account. Also, show the user who own the account.
  + Edit the details of an account
    - Admin can change the status of an account
  + Approve or decline a transaction request
    - Once the transaction is approved, the change should be reflected to the account immediately.
  + View the transaction history of an account
    - a history of the pending transactions that are not canceled by user before approved or declined.
    - a history of the transactions that have been approved and declined.
  + Delete an account from the system
* Manage Users
  + View the list of all the users and their profile details (except password)
  + View the transaction history of a user
    - a history of the pending transactions that did not canceled before approved or declined.
    - a history of the transactions that have been approved and declined.
  + Delete a user

**Users**

A user will have the following attributes: email (unique for each member), name and password.

Anyone can sign up as a user using their email, name and password. After signup, she/he can do the following tasks:

* Log in with email and password
* Edit her/his profile details
* View the details of his/her accounts
* Request the admin to create bank accounts
* Search other users using email or name.
* Add other users as friend
  + A user can only transfer money to friends.
  + Users who are not friends are not allowed to transfer money between themselves.
* View the list of his friends.
* View the list of transactions with their details. For example, the amount, from which account to which, status, start date and effective date.
* Cancel a transaction request.
* View their own transaction history.

**Miscellaneous**

**Testing**

* Thoroughly test one model and one controller. Feel free to use any testing framework.

**Deployment**

Please ensure that your deployment is always accessible. You can use the following

* Heroku or any similar PaaS (OpenShift etc) with free plans
* Amazon AWS
* NCSU VCL

**Bonus (Extra Credit)**

You can do any or all of the below for extra credit (each item would be an extra credit item).

* If a user has successfully completed a transaction(it maybe a deposit/withdraw or transfer between friends) the system should send a notification message(email) to them with the details(account number, amount and effective timestamp of a transaction of the transaction.
* User can send a borrow money request message to his/her friend. Once the friend approves this request, the transaction should be reflected to an account immediately

**Submission**

Your submission in Expertiza should consist of the following-

* A link to your deployed application
* A link to a youtube video where you should explain how to verify the functionalities of your application.
* A link to your repository (use github.ncsu.edu to host your repository)
  + Credentials for the preconfigured admin and any other information that reviewers would find useful should be in the README.md file in this repository
  + Some information that reviewers may find useful are:
    - how to test various features (e.g., how to access certain pages, what details to enter in the form etc.)
    - use cases for some of the edge case scenarios (e.g., what will happen if admin tries to delete a user who currently has a transaction not checked by admin.).