CS 251: [Code Warrior:] Webapp (Outlab Project) Lab 11

- Handed out: 9/27 Due: 10/27 10PM.
- Please write (only if true) the honor code. If you used any source (person or thing) explicitly state it. You can find the honor code on the web page.

Overview

The goal of the project is to produce an "industry strength" web app for branch change. It is the responsibility of the coders to make sure of

- Good User Interface
- Handling "dumb" users (remember, no user is dumb. the user is always right).
- Gracefully handing errors
- Understanding and exploring a web development framework.

Related Tasks

Download format of data. These will be provided in the usual Assign directory (you have all the info you need in this file itself).

Background

We wish to automate the process of branch change for candidates at the end of the year based on academic performance and capacity controls.

There are `C` candidates who desire branch change. There are `P` programmes. A programme (or branch, or program, or stream, or course) is a particular stream in the institute, e.g., CSE, EE.)

Each programme allows students based on a merit list. All programmes may share a common merit list.

Each interested student has a preference list over the programmes. We assume that this is a strict ordered list (no ties). A student need not include all possible programmes in their preference list. The goal is to allot seats to students. It is desirable for a student to get the highest preference possible; while being fair to the other students based on the merit list. That is,

1. **Merit** Change of branch is in descending order of academic performance (aka CPI).
2. **Capacity** In permitting changes from branch A to branch B, capacity controls that must be respected are (a) A should not fall below a fraction α of original capacity, and (b) B should not exceed β times the original capacity. α and β should be parameters in the implementation with default values of 0.75 and 1.1.
3. **Fairness** If a student S belonging to branch X is unable to move to branch B, no other student T with CPI less than or equal to S is allowed to move to branch B (regardless of source branch of T).(See example below).

Branch Change Rules

Here are the rules for 2015 from IITB.

1. The eligibility criteria for applying for a change of branch program are:
   (a) Must apply before the start of the third semester.
   (b) Must complete prescribed course credits in the first two semesters
   (c) No backlog at the end of the two semesters
   (d) Secured "reasonable" CPI of: at least 8.00 for general and OBC category students; and at least 7.00 for SC, ST and PwD category students.
2. Before beginning the allocation, additional seats are made available in each branch, limited to a maximum of 10% of its sanctioned strength (The β factor)
3. Each available seat may be occupied by students of any category
4. An eligible student's request for a shift from branch A to branch B will be considered valid if any one of the following two sets of criteria are satisfied.
   (a) Set 1
      i. the CPI of the student is at least 9.00
      ii. there is a seat available in branch B
   (b) Set 2
      i. There is a seat available in branch B
      ii. The strength in branch A, from which a change is being sought, does not fall below its sanctioned strength by more than 25%. (The α factor)
      iii. There is no student with higher CPI who is currently being denied a change of branch to B due to rule ii above.

Implementation:

1. Preprocessing: Initially a merit list is created based on the CPI of eligible candidates. Eligible candidates are determined by using the birth category of students and applying the eligibility CPI cutoff of 8.00 for General and OBC candidates, 7.00 for SC, ST and PwD candidates. A list of all the academic programs is also maintained with data about their sanctioned strengths and current strength.
2. Main Idea: We process the candidates in merit order (CPI) by going through each candidate in merit list and also for every candidate iterate over his preference list. Our optimality condition is to give the best possible program for the top candidate. If a student x gets one of his preferences, we proceed to the remaining candidates; we need to come back to x unless he has previously obtained his first preference.
3. Details: We iterate over the merit list and for each candidate iterate over his choices. We remove the candidate if he is allotted his first preference, else we keep the candidate in merit list order, to potentially update his branch in further iterations over the merit list. We stop the whole process only when one of the following two conditions satisfy: either the list of candidates becomes empty denoting the case where everyone gets his first choice,

or when there is no change of branch for any candidate in the merit list in the current iteration. This latter step denotes a saturation stage where we cant improve the allocation by further iterations.

4. Tricky Point on final capacities: The capacity control is done using rounding, as per Academic office. Thus if the source capacity of a branch is 82, eight more candidates can be accommodated, whereas if the source capacity is 87, nine more candidates are to be accommodated. Thus, the percentage increase can be slightly more than 10% because of method used for rounding of the number of extra seats. Significant differences can also be caused because of extending the seats so as to accommodate students with identical CPI.

5. Blocks: Although the above method appear complete, there are special blocks which prevent some of the candidates from getting choices better than what they are currently assigned.

   Example: Consider three students S1,S2,S3 and they are currently in branches A1,A2,A3 and would like to move to A2,A3,A1 respectively but none could because the branches have already reached their maximum capacity (considering the 10% overhead)(β). As per the above method none of them would be able to move onto new branches. This cycle situation could be easily resolved and all the three students can be satisfied simultaneously if we permit a temporary breach in the upper limit.

Tasks

1. Use a web development framework (say Django) to create a webapp to accomplish the branch change allocation.

2. The standard (free:) version of this project require you to implement the algorithm as specified by points 1, 2, 3, 4. The point in 5 is the challenge task for this project (paid version). (Discuss algorithm on Piazza before proceeding. May need to seek approval)

3. The webapp should be able to take user inputs (i.e., the details of students interested in branch change and show the branch allocation according to the algorithm stated above (after the deadline for submission is over).

   - Though the algorithm caters for CPI below cutoff, incorporate into the webapp UI a mechanism to prevent, or warn students having an ineligble CPI. To this end, have a standalone admin interface which can generate and populate random data for all possible 15XXXXXXX numbers and include random CPI for each. As the user tries to apply for branch change, check the database and notify the user as appropriate. The user is warned but his (her) branch change application is pushed into the queue (and later rejected by the implementation).

4. Very important: Put more emphasis on usability and robustness from the perspective of users. (i.e., the UI should be intuitive and handle error checking, you may use other frameworks like Bootstrap, AngularJS or depend on the inbuilt Django features alone for this) . The user should be able to update his previous entries. You may need to leverage the authentication system of your web framework for this. Take this as an opportunity for you to be creative and innovative. For example, instead of providing a fixed number of branch preference input boxes, the website may offer to increase or decrease as per the users preference to increase the number of options

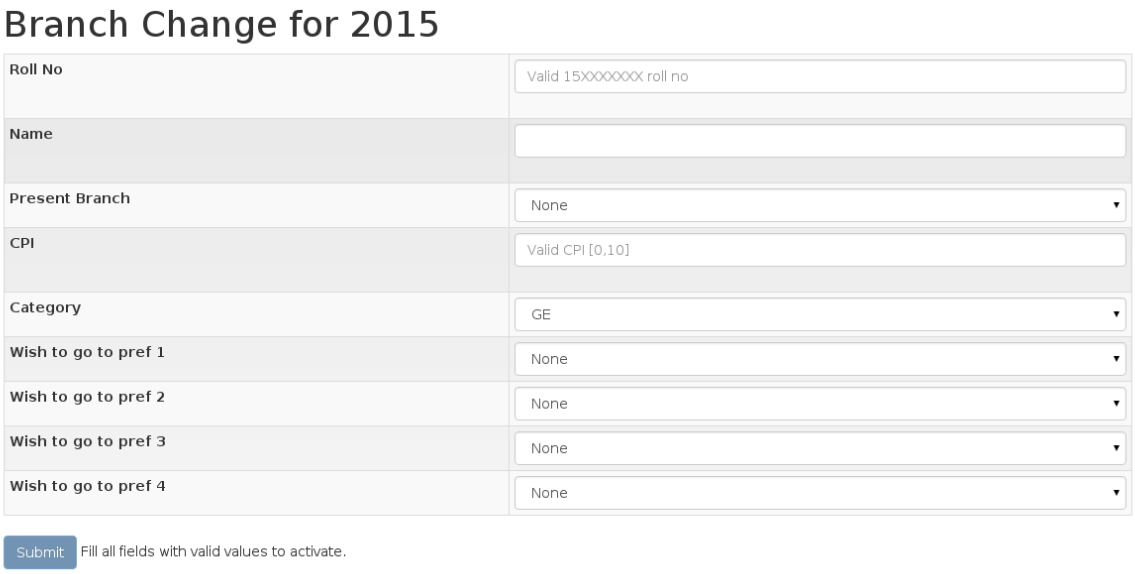

Figure 1: Deactivated Submit Button

5. The admin of the website is also a user. Allow the admin to upload / download data to the system from a csv file.(This feature is essential for testing your algorithm with sample input and output).

6. Document features of the website and create a feature document. For example here is how the feature document entry of the same feature implemented in different ways in two separate implementations :

   - Feature: RollNo field accepts only roll numbers starting with 14 and containing 9 digits.
     TestCases:
     - Enter roll number staring with 15
       Expected Behavior: The UI shows a notification to the user and submit button deactivated until the field is correctly populated. See Fig : 1 and Fig: 2
     - Enter roll number having characters greater than 9
       Expected Behavior: The UI shows a notification to the user and submit button deactivates until user fixes the issue.
   - Feature: RollNo field accepts only roll numbers starting with 15 and containing 9digits.
     TestCases:
     - Enter roll number staring with 15 and submit
       Expected Behavior: The page loads again with invalid inputs being highlighted.Data wont be accepted until correctly submitted.
     - Enter roll number having characters greater than 9 and submit
       Expected Behavior: The page loads again with invalid inputs being highlighted.Data wont be accepted until correctly submitted.

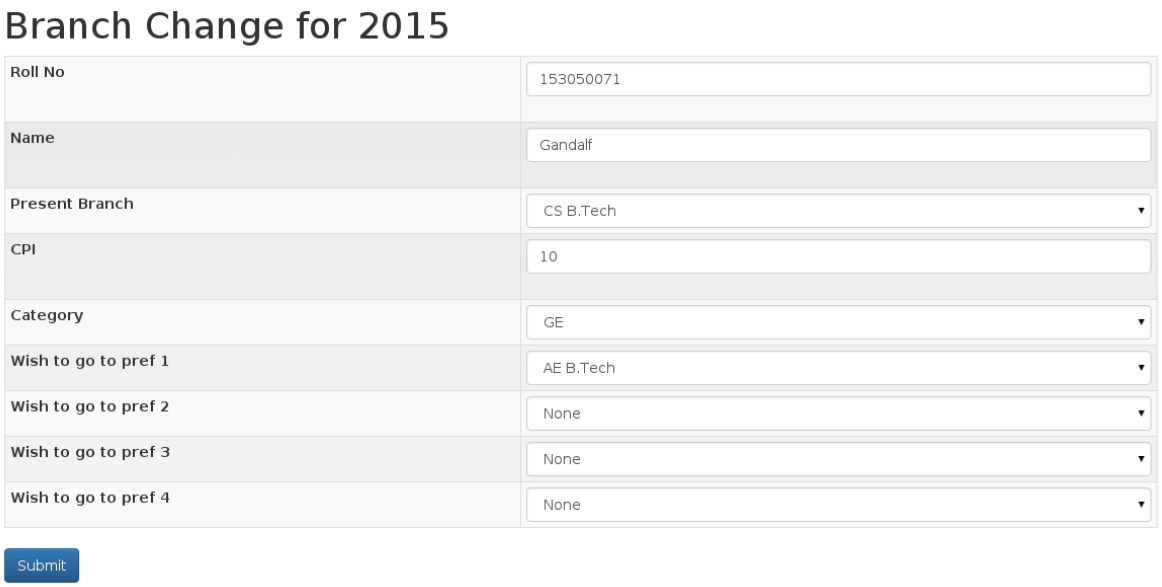

Figure 2: Activated Submit Button

Files To Submit

Create a folder `lab11_groupXY_final` e.g. `lab11_group07_final` . The folder should contain README, Django application root directory and `feature_doc.txt`. README must contain a list of commands to run in order to execute your code. README must also contain the usual information (group no, group name, members, percent of effort, honour code, citations and other relevant instructions).

Compress it to create `lab11_groupXY_final.tar.gz` e.g. `lab11_group07_final.tar.gz`.

How We Will Grade You

The following parameters will be used to grade your submission.

- Correctness of the algorithm (standard version) [40% marks]
- Basic Error checking [10% marks]
- Interactive UI [10% marks]
- Login system and ability to update previous input [10% marks]
- Comprehensive feature doc [20% marks]
- Export and import data as csv [10% marks]
- Challenge task: Challenge task will be graded only if all the above steps are in the range of superior. The algorithmic challenge will be worth 15% and the implementation another 10%

Sample Input and Output

Input

A test case where a branch hits the maximum 10% extra limit. Options given by students:

```
RollNo,Name,CurrentBranch,CPI,Category,Options
150XXXxxx,HCI,CS B.Tech,10.0,GE,EP B.Tech,EE B.Tech
150XXXxxx,OSH,EE B.Tech,9.95,GE,CS B.Tech
150XXXxxx,VAI,EE B.Tech,9.95,GE,CS B.Tech
150XXXxxx,ASN,CE B.Tech,9.74,GE,CS B.Tech,EE B.Tech,EE Dual Deg E2,EE Dual Deg E1
150XXXxxx,ASU,EE B.Tech,9.73,GE,CS B.Tech
150XXXxxx,EDS,MM B.Tech,9.71,GE,CS B.Tech,EE B.Tech,EE Dual Deg E2,EE Dual Deg E1,
ME B.Tech,ME Dual Deg M2,EP B.Tech,EP Dual Deg N1,CL B.Tech,CE B.Tech,AE B.Tech
150XXXxxx,ANV,ME B.Tech,9.71,GE,CS B.Tech,EE B.Tech
15DXXXxxx,ASL,EE Dual Deg E2,9.65,GE,EE B.Tech
15DXXXxxx,ISD,EE Dual Deg E2,9.65,GE,CS B.Tech,EE B.Tech
150XXXxxx,AGU,CL B.Tech,9.58,GE,CS B.Tech,EE B.Tech,EE Dual Deg E2
150XXXxxx,ATN,ME B.Tech,9.57,GE,CS B.Tech
15DXXXxxx,IRT,EE Dual Deg E1,9.57,GE,CS B.Tech,EE B.Tech,EE Dual Deg E2
150XXXxxx,AKL,EE B.Tech,9.53,GE,CS B.Tech
```

Current Branch strength:

```
BranchName,SanctionedStrength,CurrentStrength
AE B.Tech,62,62
CL B.Tech,124,124
CE B.Tech,117,116
CS B.Tech,87,87
EE B.Tech,60,60
EP B.Tech,30,30
ME B.Tech,116,116
MM B.Tech,98,98
EE Dual Deg E1,32,32
EE Dual Deg E2,32,32
EN Dual Deg,30,30
EP Dual Deg N1,12,12
ME Dual Deg M2,22,22
MM Dual Deg Y1,13,13
MM Dual Deg Y2,13,13
CL Dual Deg,36,36
CH,32,32
```

Output

Allotment output:

```
RollNumber, Name, Current Branch, Destination Branch
150XXXxxx,HCI,CS B.Tech,EP B.Tech
150XXXxxx,OSH,EE B.Tech,CS B.Tech
150XXXxxx,VAI,EE B.Tech,CS B.Tech
```

```
150XXXxxx,ASN,CE B.Tech,CS B.Tech
150XXXxxx,ASU,EE B.Tech,CS B.Tech
150XXXxxx,EDS,MM B.Tech,CS B.Tech
150XXXxxx,ANV,ME B.Tech,CS B.Tech
15DXXXxxx,ASL,EE Dual Deg E2,EE B.Tech
15DXXXxxx,ISD,EE Dual Deg E2,CS B.Tech
150XXXxxx,AGU,CL B.Tech,CS B.Tech
150XXXxxx,ATN,ME B.Tech,CS B.Tech
15DXXXxxx,IRT,EE Dual Deg E1,CS B.Tech
150XXXxxx,AKL,EE B.Tech,Branch Unchanged
```

Stats: (optional, may help in debugging)

```
Program,Santioned Strength,Original Strength,Final Strength
AE B.Tech,62,62,62
CL B.Tech,124,124,123
CE B.Tech,117,116,115
CS B.Tech,87,87,96
EE B.Tech,60,60,58
EP B.Tech,30,30,31
ME B.Tech,116,116,114
MM B.Tech,98,98,97
EE Dual Deg E1,32,32,31
EE Dual Deg E2,32,32,30
EN Dual Deg,30,30,30
EP Dual Deg N1,12,12,12
ME Dual Deg M2,22,22,22
MM Dual Deg Y1,13,13,13
MM Dual Deg Y2,13,13,13
CL Dual Deg,36,36,36
CH,32,32,32
```

Larger test case may be included later in the data folder.