2018

CEE Course Guru

Project Report

Asish Yadav Madala & Bhavya Duvvuri
amadala@andrew.cmu.edu
10/21/2018

1. Introduction

It is very often noticed that students face difficulties while selecting courses at the beginning of each semester. There exists a trade off between kind of skill set a student is looking for and what the course actually offers. Courses chosen have a major impact on student's career. As the course selection process is cumbersome, it is highly recommended to have appropriate course mapping to gain desired skill set. As an endeavour to overcome this problem, we hereby planned to design an application that would take student's interest as an input and facilitates him/her with a right set of courses.

1.1 Project Description

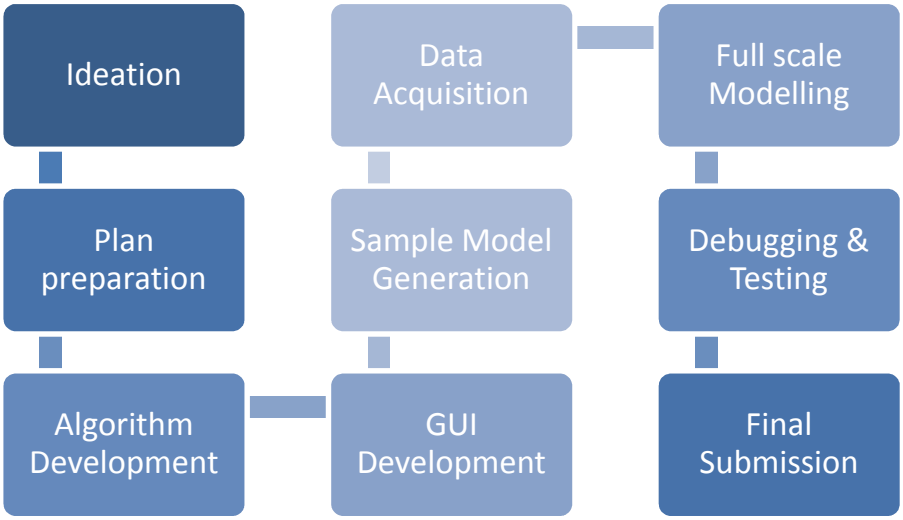
The title of this project is 'CEE Course Guru' ('Guru' stands for 'teacher'). As mentioned earlier the purpose of this application is to display the courses in the skill set chosen along with the number of units the set is having inclusive of all courses. Also the individual course wise units will be displayed in the skill set chosen.

1.2 Objectives of project

- ✓ Proper course mapping
- ✓ Minimizes the number of stakeholders involved in whole process
- ✓ Makes the activity time efficient
- ✓ Accurate unit count associated with each course can be achieved
- ✓ Complexity involved in data collection by user will be overcome
- ✓ User friendly access

1.3 Overview of approaches taken

Project went through following phases before taking its final shape:



Project algorithm is as follows:

- Checkbar development
- Defining skill set in check bar
- Creating courses set in each skill set
- Checkbar development for displaying courses
- Mapping skill set and courses
- Mapping courses and units
- Total unit count display window

Detailed features of product design

As mentioned earlier, the product is developed based on the algorithm stated. All the components involved can be described as follows:

Utilized/created attributes of python are:

- Tkinter library
- tkMessage Box
- Checkbar class
- state method

- allstates method
- Buttons (Display courses, Quit & Credits)
- allstates2 method
- credits method

Tkinter library:

The Tkinter module ("Tk interface") is the standard Python interface to the Tk GUI toolkit. Both Tk and Tkinter are available on most Unix platforms, as well as on Windows systems. (Tk itself is not part of Python; it is maintained at ActiveState.) Checkboxes & check bars are created using this library.

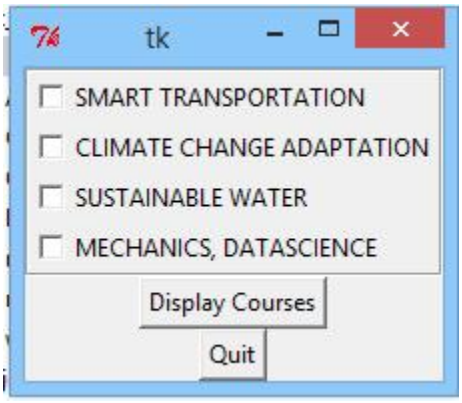
Checkboxes, also known as tickboxes or tick boxes or check boxes, are widgets that permit the user to make multiple selections from a number of different options. This is different to a radio button, where the user can make only one choice. Usually, checkboxes are shown on the screen as square boxes that can contain white spaces (for false, i.e not checked) or a tick mark or X (for true, i.e. checked). A caption describing the meaning of the checkbox is usually shown adjacent to the checkbox. The state of a checkbox is changed by clicking the mouse on the box. Alternatively it can be done by clicking on the caption, or by using a keyboard shortcut, for example, the space bar.

A Checkbox has two states: on or off. The Tkinter Checkbutton widget can contain text, but only in a single font, or images, and a button can be associated with a Python function or method. When a button is pressed, Tkinter calls the associated function or method. The text of a button can span more than one line.

Syntax for check button is as follows:

```
def __init__(self, parent=None, picks=[], side=TOP, anchor=W):
    Frame.__init__(self, parent)
    self.vars = []
    for pick in picks:
        var = IntVar()
        chk = Checkbutton(self, text=pick, variable=var)
        chk.pack(side=side, anchor=anchor, expand=YES)
        self.vars.append(var)
```

Checkboxes look as follows:



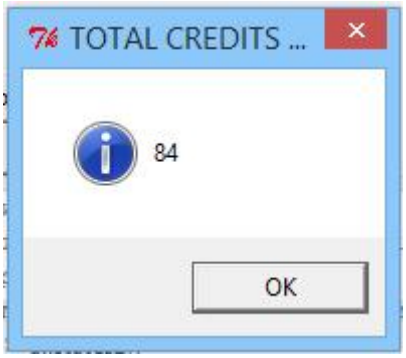
tkMessage Box:

The tkMessageBox module is used to display message boxes in the applications. This module provides a number of functions that can be used to display an appropriate message.

Syntax to create it will be as follows:

```
tkMessageBox.FunctionName(title, message [, options])
```

Following is the screenshot of message box:



Checkbar class & state method:

This class is created by us to display check bar which contains skill set and courses as check boxes. The method state is used to indicate the phase in which program is currently running.

Syntax for it is as follows:

```
class Checkbar(Frame):

    def __init__(self, parent=None, picks=[], side=TOP, anchor=W):
        Frame.__init__(self, parent)
        self.vars = []
        for pick in picks:
            var = IntVar()
            chk = Checkbutton(self, text=pick, variable=var)
            chk.pack(side=side, anchor=anchor, expand=YES)
            self.vars.append(var)

    def state(self):
        return map((lambda var: var.get()), self.vars)

if __name__ == '__main__':
    root = Tk()
    lng = Checkbar(root, ['SMART TRANSPORTATION', 'CLIMATE CHANGE ADAPTATION',
'SUSTAINABLE WATER', 'MECHANICS, DATASCIENCE'])
    lng.pack(side=TOP, fill=X)
    lng.config(relief=GROOVE, bd=2)
```

allstates method:

This part of the code is responsible for mapping skill set & courses. It has one more function of taking user's input. Three 'for loops' are used to account for the logics. The crucial part of syntax is the following logic:

```
for item in output_indices:
    output.append(mapping[item])
real_output = []
for item in output:
    for key in courses[item]:
        real_output.append(key)
real_output=list(set(real_output))
```

Credits method:

This method is used to calculate the number of units associated with the chosen skill set. It is also responsible for displaying units across each course on console screen.

Syntax is as follows:

```
credits=0
# showinfo('Credits selected are', listed,)
LBL=Label(root,text=listed).pack(side=LEFT)
print "Required Courses are "
for item in listed:
    print item, COurse_code[item], "units"
    credits += COurse_code[item]
print "Total credits are ", credits
```

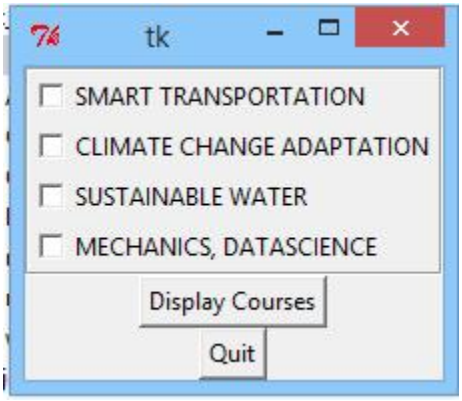
Detailed features developed

Following features have been successfully developed:

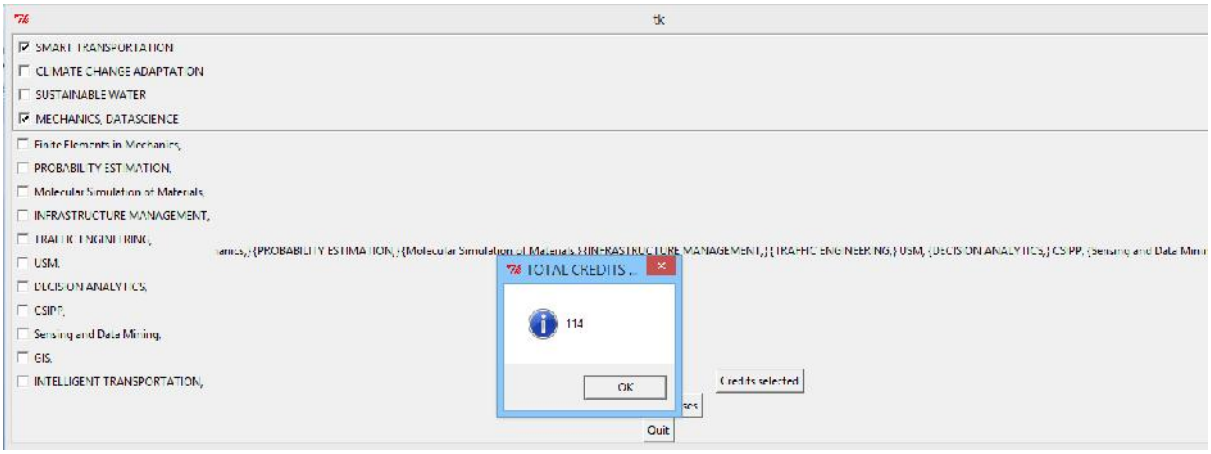
- ✓ Multi selection of skill set
- ✓ Appropriate display of courses according to skill set chosen
- ✓ Credit display as message box
- ✓ Credit display course wise on console

The application works in the following way:

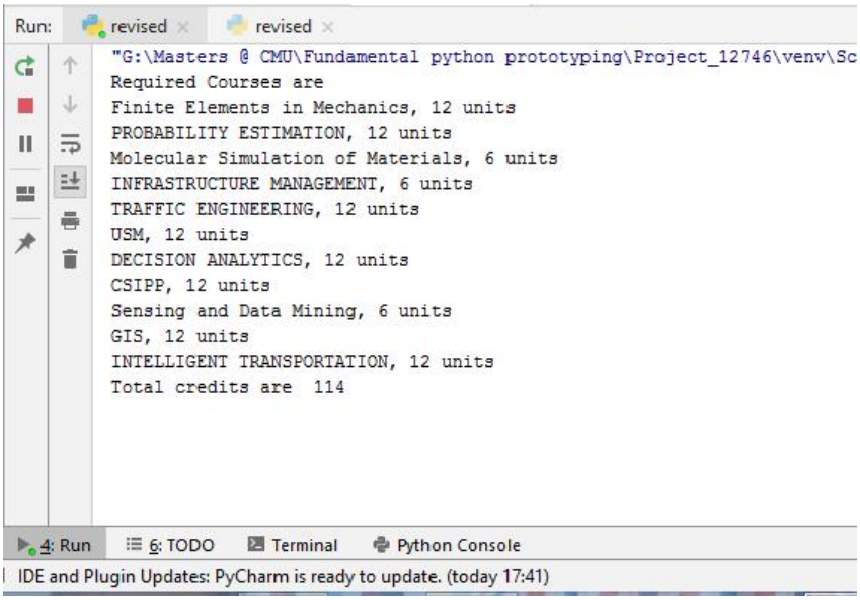
On running the code, the user will be prompted to choose the skill set that he/she is looking to opt for.



On selecting the skill set, the courses under the chosen skill set will be displayed without any overlap.



In this case we have chosen four skill sets namely, Smart Transportation, Climate change adaptation, sustainable water & Mechanics, Datascience. The courses under these sets are displayed along with total units inclusive of all courses in skill set chosen popped out. Following things are printed on console:



It can be clearly seen that all the courses are listed out along with number of units that they carry.

Assessment of approach

The approach taken by us has worked out well. It helped us in achieving the features that we wanted to develop in the application. 'Tkinter' library is used to develop the GUI for the app, TkMessenger library is used to display message and error message box. Also, the results of the application are displayed on the console to check if the code is executing as per the algorithm. The code is developed such that it will be further developed to count the number of credits selected based on the courses selected by the user from the skillset courses displayed which is based on a previous input. By the end of the project we are comfortable using for, if loops, defining methods, initialisation class, initialisation methods, defining dictionaries, lists, 'Tkinter' library for GUI development.

Scope

Project is scheduled for expansion in mini 2. In the coming course we'll be aiming at incorporating more features in to this application. We'll be running it on real time basis by collecting the user input on mySQL database. Further a little data analytics is expected to be assigned to the project while using mapping. This takes the application to next level, making it more versatile. Also app's performance can be smoothened more by working on it.

Conclusion

This application is very basic one. This has been made in accordance with what we've learnt so far in the course. All the features mentioned at the time of proposal are included. This application can be made available to students once the 2nd mini is complete, as more features are expected to come along with it. This can be a helpful tool to the students while registering for the courses and thus it minimizes the number of stake holders involved in it.

Please note that there is no reference available elsewhere for developing this code and hence it is written right from the scratch by us.