MainMenu.py

```
from tkinter import *
import tkinter.messagebox
import tkinter.simpledialog
import tkinter.colorchooser
from tkinter.filedialog import askopenfilename
import time
import threading
from PIL import Image, ImageTk, ImageDraw, ImageFont
from Users import student
from Users import professor
from Users import admin
import sys
import os
import pickle

LARGE_FONT= ("Verdana", 12)


curr_user = ""
mydict = {}

class Main():
    def __init__(self):
        window = Tk()
        window["bg"] = "white"
        menu_frame = Frame(window)
        menu_frame.pack(side = "top", fill="both", expand= True)
        menu_frame["bg"] = "white"
        menu_frame.grid_rowconfigure(0, weight=1)
        menu_frame.grid_columnconfigure(0, weight=1)

        load = Image.open("pics/iclogoweb.png")
        load = load.resize((600, 75))
        render = ImageTk.PhotoImage(load)
        label = Label(menu_frame, image = render,bg="white")
        label.image = render
        label.pack()
        # label.pack(pady=10, padx=10)

        self.img_num = 0
        self.img_total = 0
        #self.img_arr = ["pics/B.gif", "pics/D.gif", "chat.png"]
        self.img_arr = []

        self.init_img()
        self.img_label = Label(menu_frame, image=render,bg="white")
        self.img_label.image = render
        self.img_label.pack()

        self.changeImg()

        self.time_str = StringVar()
        time_label = Label(menu_frame, textvariable=self.time_str, font=LARGE_FONT)
        time_label.pack()

        login_bt = Button(menu_frame, text="Login", command=self.login, font = LARGE_FONT)
        login_bt.pack()

        self.printDateTime()

        window.mainloop()

    def printDateTime(self):
        threading.Timer(1.0, self.printDateTime).start()
        self.time_str.set((time.strftime('%d/%m/%Y %H:%M:%S')))

    def init_img(self):
        threading.Timer(5.0,self.init_img).start()
        self.img_total = 0
        self.img_arr = []
        picFile = open("pic_filenames.txt", "r")
        line = picFile.readline()
        self.img_arr.append(line[:-1])
        while line != '':
            line = picFile.readline()
            self.img_arr.append(line[:-1])
            self.img_total += 1
        picFile.close()

    def changeImg(self):
        threading.Timer(2.0, self.changeImg).start()
        # print("num:",self.img_num,"total:", self.img_total)
        if self.img_num >= self.img_total:
            self.img_num = 0
        # print(self.img_arr[self.img_num])
        # print(self.img_num)
        load = Image.open(self.img_arr[self.img_num])
        load = load.resize((600,400))
        render = ImageTk.PhotoImage(load)
        self.img_label.config(image=render)
        self.img_label.image = render
        self.img_num += 1
    def login(self):
        global curr_user
        global mydict
        pickle_in = open("dict.pickle", "rb")
        mydict = pickle.load(pickle_in)
        username = tkinter.simpledialog.askstring("Login", "Enter your username")
        password = tkinter.simpledialog.askstring("Login", "Enter your password", show="*")

        if username in mydict:
            if password == mydict[username].getPassword():
                curr_user = username
                if isinstance(mydict[curr_user],student):
                    self.userWindow()
                elif isinstance(mydict[curr_user], professor):
                    self.profWindow()
                elif isinstance(mydict[curr_user], admin):
                    self.adminWindow()
            else:
                tkinter.messagebox.showerror("Login", "Incorrect Password")
        else:
            tkinter.messagebox.showerror("Login", "Username incorrect or not found")

    def init_curr_user(self, s1, s2, s3, s4):
        curr_user = s1
        curr_pass = s2
        curr_name = s3+" "+s4[:-1]
        print("init")
        print(curr_user)
        print(curr_pass)
        print(curr_name)

    def userWindow(self):
        userwin = UserPage()

    def adminWindow(self):
        adminwin = AdminPage()

    def profWindow(self):
        profwin = ProfPage()


class UserPage():
    def __init__(self):
        self.user_win = Toplevel()

        top_frame = Frame(self.user_win)
        top_frame.pack(side="top", fill="both", expand=True)
        top_frame["bg"] = "light green"
        top_frame.pack()
        new_label = Label(top_frame, text=mydict[curr_user].get_info(), bg= "light green")
        new_label.pack()

        user_frame = Frame(self.user_win)
        user_frame.pack()

        change_pass_bt = Button(user_frame,text="New password",command = self.change_pass)
        change_pass_bt.pack(padx=3, pady=3,side=LEFT)

        add_bt = Button(user_frame,text="Add Pic", command= self.add_pic)
        add_bt.pack(padx=3, pady=3,side=LEFT)

        rm_bt = Button(user_frame, text="Remove Pic", command=self.remove_pic)
        rm_bt.pack(padx=3, pady=3,side=LEFT)

        # quit_bt = Button(self.user_win, text="Quit", command=self.user_win.destroy)
        # quit_bt.pack()

    def change_pass(self):
        new_pass = tkinter.simpledialog.askstring("New password", "Enter your new password", show="*")
        new_pass2 = tkinter.simpledialog.askstring("New password", "Enter your password again",
show="*")
        if new_pass == new_pass2:
            mydict[curr_user].setPassword(new_pass)
            pickle_out = open("dict.pickle", "wb")
            pickle.dump(mydict, pickle_out)
            pickle_out.close()
        else:
            tkinter.messagebox.showerror("New password", "passwords do not match")
    def add_pic(self):
        image_dir = askopenfilename()
        outfile = open("pic_filenames.txt", "a")
        outfile.write(image_dir + "\n")
        outfile.close()

    def remove_pic(self):
        rm = RemovePicPage()


class ProfPage(UserPage):
    def __init__(self):
        super().__init__()
        mid_frame = Frame(self.user_win)
        mid_frame.pack(side="top", fill="both", expand=True)
        mid_frame["bg"] = "pink"
        mid_frame.pack()
        Label(mid_frame,text="Professor Functions", bg = "pink").pack()
        lst_frame = Frame(self.user_win)
        lst_frame.pack()
        cancel_bt = Button(lst_frame,text="Cancel Class",command = self.cancel_class)
        cancel_bt.pack(padx=3, pady= 3, side = LEFT)

        self.subject = StringVar()
        self.subjects = mydict[curr_user].get_subjects()
        self.subject.set(self.subjects[0])
        choice_menu = OptionMenu(lst_frame, self.subject, *self.subjects)
        choice_menu.pack(padx=3, pady= 3, side = LEFT)

        meet_bt = Button(self.user_win,text="Add meeting", command = self.add_meeting)
        meet_bt.pack()

    def cancel_class(self):
        text = self.subject.get()+" class is canceled "
        fn_large = ImageFont.truetype("arial.ttf", 30)
        fn_small = ImageFont.truetype("arial.ttf", 22)

        img = Image.new('RGB', (600, 400), "white")
        draw = ImageDraw.Draw(img, 'RGBA')
        draw.text((30, 10),"Class Annoucements" ,font= fn_large, fill=(0, 0, 255, 255), )
        draw.text((30, 50), text ,font= fn_small, fill=(0, 0, 255, 255), )

        img.save("class.png")

    def add_meeting(self):
        fn_large = ImageFont.truetype("arial.ttf", 30)
        fn_small = ImageFont.truetype("arial.ttf", 22)
        text = tkinter.simpledialog.askstring("Create Meeting", "Create Meeting:")
        text = text+" "
        img = Image.new('RGB', (600, 400), "white")
        draw = ImageDraw.Draw(img, 'RGBA')
        draw.text((30, 10),"Meetings Scheduled" ,font= fn_large, fill=(0, 0, 255, 255), )
        draw.text((30, 50), text ,font= fn_small, fill=(0, 0, 255, 255), )

        img.save("meet.png")


class AdminPage(UserPage):
    def __init__(self):
        super().__init__()
        mid_frame = Frame(self.user_win)
        mid_frame.pack(side="top", fill="both", expand=True)
        mid_frame["bg"] = "pink"
        mid_frame.pack()
        Label(mid_frame, text="Admin Functions", bg="pink").pack()
        lst_frame = Frame(self.user_win)
        lst_frame.pack()

        add_user_bt = Button(lst_frame,text = "Add User", command= self.add_user)
        add_user_bt.pack(padx=3, pady=3, side=LEFT)

        self.choice = StringVar()
        self.choices = ["student", "professor"]
        self.choice.set(self.choices[0])
        choice_menu = OptionMenu(lst_frame, self.choice, *self.choices)
        choice_menu.pack(padx=3, pady=3, side=LEFT)

        rm_user_bt = Button(self.user_win,text = "Remove User", command= self.rm_user)
        rm_user_bt.pack(padx=3)

        quit_bt = Button(self.user_win, text="Quit", command=self.user_win.destroy)
        quit_bt.pack(padx=3, pady=3)

    def add_user(self):
        c = self.choice.get()
        if c == "student":
            add_user_page = AddStudPage()
        elif c == "professor":
            add_user_page = AddProfPage()
        #add_user_page = AddUserPage()

    def rm_user(self):
        user = tkinter.simpledialog.askstring("Remove User", "Enter the username of\n the user you want
removed")
        if user in mydict:
            mydict.pop(user)
            pickle_out = open("dict.pickle", "wb")
            pickle.dump(mydict, pickle_out)
            pickle_out.close()
        else:
            tkinter.messagebox.showerror("Remove User", "Username not found")


class AddUserPage():
    def __init__(self):
        self.uadd_page = Toplevel()

        self.username = StringVar()
        self.password = StringVar()
        self.fname = StringVar()
        self.lname = StringVar()
        Label(self.uadd_page, text="Username:").pack()
        Entry(self.uadd_page, textvariable=self.username, justify=RIGHT).pack()

        Label(self.uadd_page, text="Password:").pack()
        Entry(self.uadd_page, textvariable=self.password, justify=RIGHT).pack()

        Label(self.uadd_page, text="First name:").pack()
        Entry(self.uadd_page, textvariable=self.fname, justify=RIGHT).pack()

        Label(self.uadd_page, text="Last name:").pack()
        Entry(self.uadd_page, textvariable=self.lname, justify=RIGHT).pack()


class AddStudPage(AddUserPage):
    def __init__(self):
        super().__init__()

        self.major = StringVar()
        self.majors = ["SE", "ETM"]
        self.major.set(self.majors[0])
        major_menu = OptionMenu(self.uadd_page, self.major, *self.majors)
        major_menu.pack()

        add_bt = Button(self.uadd_page, text="add user", command=self.add_user)
        add_bt.pack()

    def add_user(self):
        user = self.username.get()
        password = self.password.get()
        fname = self.fname.get()
        lname = self.lname.get()
        major = self.major.get()

        new_user = student(user,password,fname,lname,major)

        mydict[user] = new_user

        pickle_out = open("dict.pickle", "wb")
        pickle.dump(mydict, pickle_out)
        pickle_out.close()

        self.uadd_page.destroy()


class AddProfPage(AddUserPage):
    def __init__(self):
        super().__init__()
        self.subjects = StringVar()
        Label(self.uadd_page, text="Subjects:").pack()
        Entry(self.uadd_page, textvariable=self.subjects, justify=RIGHT).pack()

        add_bt = Button(self.uadd_page, text="add user", command=self.add_user)
        add_bt.pack()

    def add_user(self):
        user = self.username.get()
        password = self.password.get()
        fname = self.fname.get()
        lname = self.lname.get()
        subjects = self.subjects.get()
        subjects = subjects.split(',')

        new_user = professor(user,password,fname,lname)
        for s in subjects:
            new_user.add_subject(s)

        mydict[user] = new_user

        pickle_out = open("dict.pickle", "wb")
        pickle.dump(mydict, pickle_out)
        pickle_out.close()

        self.uadd_page.destroy()


class RemovePicPage():
    def __init__(self):
        self.rm_page = Toplevel()

        self.rm_pic = StringVar()
        self.pic_arr = []
        self.init_choices()
        choice_menu = OptionMenu(self.rm_page, self.rm_pic, *self.pic_arr)
        choice_menu.pack()

        confirm_rm_bt = Button(self.rm_page,text="remove",command = self.rmpage_remove)
        confirm_rm_bt.pack()

    def init_choices(self):
        self.pic_arr = []
        picFile = open("pic_filenames.txt", "r")
        line = picFile.readline()
        self.pic_arr.append(line[:-1])
        while line != '':
            line = picFile.readline()
            self.pic_arr.append(line[:-1])
        picFile.close()
        self.rm_pic.set(self.pic_arr[0])


    def rmpage_remove(self):
        target = self.rm_pic.get()+"\n"
        print(self.rm_pic.get())
        print(self.pic_arr)

        picFile = open("pic_filenames.txt", "r")
        s = picFile.read()
        print(s)
        s= s.replace(target, '')
        print(s)
        picFile.close()

        outFile = open("pic_filenames.txt", "w")
        outFile.write(s)
        outFile.close()

        self.rm_page.destroy()


Main()
```

User.py

```
import abc

class User(metaclass=abc.ABCMeta):

    def __init__(self, username, password, firstName, lastName):
        self.username = username
        self.password = password
        self.firstName = firstName
        self.lastName = lastName

    def getPassword(self):
        return self.password

    def setPassword(self,password):
        self.password = password

    @abc.abstractmethod
    def get_info(self):
        pass
```

```
class student(User):

    def __init__(self,username, password, firstName, lastName,major):
        super().__init__(username, password, firstName, lastName)
        self.major = major

    def get_info(self):
        return "Student: "+self.firstName+" "+self.lastName+" "+self.major

    def set_major(self,major):
        self.major = major

class professor(User):

    def __init__(self,username, password, firstName, lastName):
        super().__init__(username, password, firstName, lastName)
        self.subjects = []

    def get_info(self):
        return "Professor: "+self.firstName+" "+self.lastName

    def add_subject(self,subject):
        self.subjects.append(subject)

    def get_subjects(self):
        return self.subjects


class admin(User):
    def __init__(self, username, password, firstName, lastName):
        super().__init__(username, password, firstName, lastName)

    def get_info(self):
        return "ADMIN: "+self.firstName+" "+self.lastName
```