

**SYNOPSIS: *Intelligent Emergency-Auto-Adjustments for Faculty***

***GROUP 17 –***

***GITHUBlink-***

***https://github.com/anmolpradhan/INT404-Project***

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Roll No | Registration No | Section |
| Anmol Pradhan | 65 | 11813478 | K18XC |
| Ramanadham Yuva Ganesh | 64 | 11813447 | K18XC |
| Merugu Venkatesh | 63 | 11813398 | K18XC |
| Vutla Ajay Kumar | 62 | 11813398 | K18XC |

1. INTRODUCTION

As the name infers, Intelligent Emergency auto modifications for Faculty Helps

the University to discover the staff on recreation and alter them with the

workforce on crisis leave that causes understudy to go with their classes not

surprisingly.

How would you recognize Faculty on leave and change them with workforce on

relaxation?

1. A University has numerous instructors for the equivalent subject/course.

Identify Entities. The substances right now are Department, Teachers

accessible.

2. alteration. We developed timetable grids for each hold educators and normal

teachers

3. we likewise have the executive’s mode with the goal that one can without

much of a stretch include more resources by which TensorFlow execution

would be conceivable

4. Instructor Priorities, save workforce who are accessible for task can be

organized and can be picked the best one for the activity.

2. Background

We use simple code not advance to enforce the mission

“Intelligent Emergency auto adjustment for faculty”.

In this project just like query management it is ask the information and

implement your adjustment with it. The history section should talk your

findings in a chronological way to accentuate the development in the discipline

and the missing points that need to be addressed. The historical past must be

written as a precis of your interpretation of previous studies and what

your look at proposes to accomplish.

3. Motivation

Unlike maximum tangible project management

functions, motivation is not certain by the project supervisor to a group

member, alternatively motivation is inner to every team member and derived

from a crew member's preference to attain a goal, accomplish a task, or work

in the direction of expectations.

4. Goals and objective

our primary and important Goal here turned into to reap this problem of assigning schools to the magnificence that was abandoned by way of their

Professor to reap maximum utilisation of the time of students ans nicely of

the colleges and reserve colleges.

We made it our number one objective to make this program easy to use and

be scalable and be effortlessly diagnosable in case of any malfunctioning.

Description

this code is of faculty management device and uses AI concepts to assign

reserve professors in threat one lively professor applies leave.

5. Technologies and frameworks used

Python:

we selected python as our choice due to its ease of

implementation and simplicity of use and the really nicely integration of the

A.I. Libraries in python.

5. Code

*import* numpy *as* np

*import* datetime

*import* calendar

*import* pandas *as* pd

def login\_screen(teacherDB):

print("Auto Adjustment For Faculty")

print(" Teacher 1: 1021 \n Teacher 2: 1022 \n Teacher 3: 1023 \n Teacher 4: 1024 \n")

teacherID = int(input("please enter your Teacher ID >> "))

*if*(teacherID in teacherDB.keys()):

teacherID = teacherDB[teacherID]

welcome\_screen(teacherID)

*else*:

print("your record doesnot exist")

def show\_free\_slots(teacherID, day):

*for* time\_Slot in range(5):

*if* (profTT[teacherID][day][time\_Slot] == 0):

print(trans\_dict\_time[time\_Slot], " is free")

def show\_taken\_slots(teacherID, day):

*for* time\_Slot in range(5):

*if* (profTT[teacherID][day][time\_Slot] == 1):

print(trans\_dict\_time[time\_Slot], " is occupied")

def show\_makeup\_slots(teacherID, day):

*for* time\_Slot in range(5):

*if* (profTT[teacherID][day][time\_Slot] == 2):

print(trans\_dict\_time[time\_Slot], " is make up adjustment")

def welcome\_screen(teacherID):

print("Menu")

print("1> View schedule ")

print("2>Appoint a make-up class ")

print("3>Apply emergency leave ")

print("4>Exit ")

choice = int(input("Choose > "))

*if*(choice == 1):

day = input("Enter the day >> ")

print("\nYOUR SCHEDULE \n")

day = trans\_dict\_day[day]

show\_taken\_slots(teacherID, day)

show\_free\_slots(teacherID, day)

show\_makeup\_slots(teacherID, day)

choice = input("would you like to return to home page? >> ").lower()

*if*(choice == "yes"):

print("redirecting you to home page \n\n")

welcome\_screen(teacherID)

*elif*(choice == 2):

day = input("please enter the day you wish to apply for a make up >> ")

day = trans\_dict\_day[day]

print("please choose time slot for the leave")

print("10:00AM - 11:00AM >> 0")

print("11:00AM - 12:0PAM >> 1")

print("1:00PM - 2:00PM >> 2")

print("2:00PM - 3:00PM >> 3")

print("3:00PM - 4:00PM >> 4")

time\_slot = int(input("option >> "))

MakeUpApp(0, teacherID, day, time\_slot)

*elif*(choice == 3):

day = input("please enter the day you wish to apply for a leave >> ")

day = trans\_dict\_day[day]

print("please choose time slot for the leave")

print("10:00AM - 11:00AM >> 0")

print("11:00AM - 12:0PAM >> 1")

print("1:00PM - 2:00PM >> 2")

print("2:00PM - 3:00PM >> 3")

print("3:00PM - 4:00PM >> 4")

time\_slot = int(input("option >> "))

leaveApp(teacherID, day, time\_slot)

*elif*(choice == 4):

print("\nexiting!")

def leaveApp(teacherID, day, time\_slot):

print("\nupdating time schedule")

profTT[teacherID][day][time\_slot] = 0

MakeUpApp(1, teacherID, day, time\_slot)

choice = input("would you like to return to home page? >> ").lower()

*if*(choice == "yes"):

print("redirecting you to home page \n\n")

welcome\_screen(teacherID)

def MakeUpApp(flag, teacherID, day, time\_slot):

*if*(flag == 0):

profTT[teacherID][day][time\_slot] = 2

print("\nsucessfully updated\n")

choice = input("would you like to return to home page? >> ").lower()

*if*(choice == "yes"):

print("redirecting you to home page \n\n")

welcome\_screen(teacherID)

*else*:

ResProfChoices = []

*for* ResProf in range(len(resprofTT)):

*if*(resprofTT[ResProf][day][time\_slot] == 0):

ResProfChoices.append(ResProf)

*if*(ResProfChoices.count == 1):

resprofTT[ResProfChoices[0]][day][time\_slot] = 2

print("\n\nassigning reserve professor",

ResProfChoices[0]+1, "for the time slot\n\n")

*else*:

*# print(ResProfChoices)*

bestChoice = GetBestChoice(ResProfChoices, day, time\_slot)

print("\n\nassigning reserve professor",

bestChoice+1, "for the time slot\n\n")

def GetBestChoice(choices, day, time\_slot):

priorityTable = {}

*for* i in choices:

priorityTable[i] = 0

*if*(time\_slot > 4 and time\_slot < 0):

*if* (resprofTT[i][day][time\_slot-1] == 1 or resprofTT[i][day][time\_slot+1] == 1):

priorityTable[i] += 1

*elif* (resprofTT[i][day][time\_slot-1] == 1 and resprofTT[i][day][time\_slot+1] == 1):

priorityTable[i] += 2

*# print(priorityTable)*

*for* key, value in priorityTable.items():

*if*(value == max(priorityTable.values())):

*# print(key)*

*return* key

defTT = [[0, 0, 0, 0, 0], [0, 0, 0, 0, 0], [

0, 0, 0, 0, 0], [0, 0, 0, 0, 0], [0, 0, 0, 0, 0]]

profTT = [[[0, 1, 0, 1, 0], [1, 1, 0, 0, 1], [1, 1, 0, 1, 1], [0, 1, 1, 1, 0], [1, 1, 0, 1, 1]],

[[1, 0, 0, 1, 1], [0, 1, 0, 1, 0], [0, 1, 1, 0, 0],

[1, 0, 1, 0, 0], [1, 0, 1, 0, 1]],

[[0, 1, 1, 0, 1], [1, 0, 1, 0, 0], [1, 0, 0, 1, 0],

[1, 0, 1, 0, 1], [1, 1, 1, 0, 0]],

[[1, 1, 1, 0, 0], [0, 0, 1, 1, 1], [0, 0, 1, 0, 1], [0, 1, 0, 1, 1], [0, 0, 1, 1, 1]]]

resprofTT = [[[1, 0, 0, 1, 1], [0, 1, 0, 1, 1], [1, 1, 0, 0, 1], [1, 0, 0, 1, 1], [0, 1, 0, 1, 1]],

[[0, 1, 1, 1, 0], [1, 1, 0, 0, 0], [0, 1, 0, 1, 0],

[0, 1, 1, 0, 0], [1, 1, 0, 0, 0]],

[[1, 1, 0, 0, 0], [1, 0, 1, 0, 0], [1, 0, 0, 1, 1],

[1, 1, 0, 1, 1], [1, 0, 1, 0, 0]],

[[0, 0, 1, 1, 1], [0, 0, 1, 1, 1], [1, 0, 1, 0, 0], [0, 0, 1, 0, 1], [0, 0, 1, 1, 1]]]

trans\_dict\_day = {"monday": 0, "tuesday": 1,

"wednesday": 2, "thursday": 3, "friday": 4}

trans\_dict\_day\_opp = {0: "monday", 1: "tuesday",

2: "wednesday", 3: "thursday", 4: "friday"}

trans\_dict\_time = {0: "10:00AM - 11:00AM", 1: "11:00AM - 12:00PM ",

2: "1:00PM - 2:00PM", 3: "2:00PM - 3:00 PM ", 4: "3:00PM - 4:00PM"}

teacherDB = {1: 0, 1021: 0, 1022: 1, 1023: 2, 1024: 3}

login\_screen(teacherDB)

6. Analysis

Strengths:

● Unique interface which is simple to apprehend and use

● Can be scaled up nationally and globally.

● may be connected effortlessly to database and then shop the data for

authentication

● Relatively clean to diagnose the issues if occured

Weaknesses:

● Not anybody can use this service as it is available to offline database

handiest for now

● tideous paintings of adding in colleges

● very sluggish if a variety of faculties time schedules are present

Opportunities:

● Expand excessive the usage of TensorFLow

● can be used by colleges faculties and all educational institutions so we can find customers without problems

Threats:

● feasible authentication failures.

● no fail safe methods added