Program 1 - Google Form

18|04|2022

Demonstrate the working of google form to develop event feedback system.

Requirements : Internet Connection , Google account

Theory :

Google form is a tool that allows collecting information from user through a personalized survey or quiz format. The information collected is ~~then~~ automatically connected to a spreadsheet. The service has undergone several updates and are not limited to menu search, shuffle of questions for recognized order, limiting response to once per person, shorter URLs, custom themes automatically generating answer, suggesting when creating forms and an upload file option too. The app allows users to create and edit surveys online while collaborating with other users in real time.

Steps Involved :

1. Go to google apps. Click on Google forms
2. Create a feedback form where we can select fields for response like short answer, paragraph, multiple choice, checkbox, dropbox menu.
3. To add another question, click on the plus (+) symbol
4. To give multiple choice to user, click on add options
5. To give dropdown to user, click on add option
6. To give date option to user, click on add option
7. To give time option to user, click on add option
8. To send the form to user select the send option. Give all the information about receiver's mail id and other information and click on 'send', or we can copy the link and send it via other means.
9. We can know the response of the sent form by clicking

on the Responses Option.

10. In Responses options, 'Summary' option gives the response of each question along with pie chart and bar graph for the suitable questions as response.

In the 'Question' option, we get a dropdown option of questions and we can select a question and view the response for that question only.

In the 'Individual' option, we get the response for each person individually and a slide through arrows to switch between different responses.

11. The responses can be extracted and viewed in the form of a table in excel sheets, through download responses (.csv) option. We can also point all response and delete all responses.

Google Sheets

25|04|2022

Requirements : Internet Connection , Google account

Theory :

Google sheets is a spreadsheet program included as part of the free, web-based Google Docs Editor Suite offered by Google. It is an online spreadsheet app that lets you to create and format spread sheets. People can work simultaneously , we can see people's changes as they make them and every change is saved automatically.

Steps Involved :

1. Go to google apps . Click on 'Google Sheets'.
2. Click on 'Blank' to get a new untitled spreadsheet and rename the file.
3. To create the 'COURSE OUTCOME CALCULATION' through spreadsheet , insert in the cells , college name, branch name, semester, subject name , subject code and title.
4. The ~~rows~~ columns in the sheet should have serial number, USN of the student , Student Name , CIE 1 marks along with the course outcome (CO1 , CO2 or CO3) the particular questions (1a , 1b, 2a, 2b, 2c) belongs to and similarly CIE II marks , then Assignment I marks along with the course outcome which it belongs to and similarly Assignment II marks.
5. The next three columns are CO 1 , CO 2 , CO3 which are the sum of marks obtained in each course outcome through CIE and Assignment marks. This is calculated using the formula :

   ⇒ SUM (E16, F16, G16 , Q16)

   (E16 , F16 , G16 , Q16 are particular cell numbers)

6. We can apply this formula for all the rows by cli
on the bottom right corner of the cell and dragging it
to all rows, needed. Formulas are typed in that particular ce

7. The average of each course outcome is calculated as
by using the formula :

⇒ 0.75 * AVERAGE (T16 : T30)

(T16 : T30 are ~~total~~ number of rows to which the formula is applied in the T column of the sheet)
The calculated value will appear in the cell.

8. The target level is set as LEVEL 1 (< 40 %) , LEVEL 2 (> 40 % and < 75 %) and LEVEL 3 (> 75 %).

9. They total students for each course outcome (CO1, CO2, CO3) is ~~written~~. inserted.

10. Students above target for each course outcome is calculated using the formula :

⇒ COUNTIF (T16 : T30 , ">=" & T13)

(To check if each individual student has scored above the average calculated in step 7 or not)
The calculated value will appear in the cell.

11. To calculate CO Assessment (%) , we use the formula :

⇒ (E43 / E42 * 100)

(or) ('STUDENTS ABOVE TARGET' / TOTAL STUDENTS' * 100)
The calculated value will appear in the cell.

12. The 'CO ATTAINMENT LEVEL' is calculated using the formula :

⇒ IF (E44 > 75 , "3" , IF (AND (E44 > 40, E44 < 75),
"2" , IF (E44 < 40 , "1" , "0")))

where ,

E44 ⟶ CO ASSESSMENT (%)

13. We can calculate the above formulas for one course outcome
Suppose CO1 and then click on the bottom right

corner of the cell and drag it to the column CO2 and CO3 for each formula.

14. All the changes made are saved automatically.

Programs → Code anywhere

Requirements : Internet connection, Code anywhere access

Theory :

Cloud anywhere is a Cloud IDE. It saves time by deployment deploying a development environment to secure enabling you to code, learn, build and collaborate on your projects. It is a fully featured web-based code editor. One can team up with their fellow developers and collaborate on their projects. It has pre-installed language runtime and tools for all popular programming languages.

Python Development Stack with pyenv, Python and pip preinstalled.

pyenv is a command-line tool to manage multiple python versions.

The codeanywhere Container comes with:

- 10GB of Disc Storage
- 2048 MB RAM (+512 MB swap)
- Sudo access
- SSH access ~~to~~ on host23.codeanyhost.com -p 52320
- Access to all HTTP and Websocket ports.

Program - 5 (File handling 1)

Steps :

1. Create an account in codeanywhere by providing your email-id and password through Sign Up option.

2. Verify your account by clicking on the link sent to the given email-id.

3. Create a new container in the Dashboard by clicking on 'New Container'.

4. Select Python and click on Create

5. Once the container is created, an IDE workspace will open.

6. Create new project by clicking on new folder option and name it 'Project'

7. ~~Open~~ Create a new file named '1.py' by clicking on new file option.

8. Type the following code in the file :

```
def program5():
with open("/home/cabox/workspace/merge.txt","r")
 as f1:
      data = f1.readlines()
 cnt_lines = 0
 cnt_A = 0
 cnt_B = 0
 cnt_C = 0
 for lines in data :
      cnt_lines +=1
      if lines[0] == 'A' :
           cnt_A += 1
      if lines[1] == 'B' :
           cnt_B += 1
      if lines[2] == 'C' :
           cnt_C += 1
```

```
print (" Total number of lines are : ", cnt_lines)
print (" Total number of lines starting with A are :
  cnt_A)
print (€" Total number of lines starting with B are :
  cnt_B)
print(" Total number of lines starting with C are :
  cnt_C)
program5()
```

9. Save the file '1.py'

10. Create a file named 'Merge.txt' using new file option.

11. In 'merge.txt' file, insert the following data :

```
Apples are fruits
Apricots are tasty.
Zebra is an Animal
arrangements are done.
Bob likes pets.
Coconut Water is good for health.
Banana is also a fruit.
Bombay is now Called Mumbai.
```

12. Save the file 'merge.txt'. Copy the path of the file and paste it in the code's open()function.

13. In the console to run the ~~command~~ code ;
  type : python 1.py

14. The output will be displayed in the console.

OUTPUT :

```
Total number of lines are : 8
Total number of lines starting with A are : 2
Total number of lines starting with B are : 3
Total number of lines starting with C are : 1
```

Program - 6 : (File-handling 2)

Steps :

1. Create an account in Codeanywhere by providing your email-id and password through Sign Up option

2. Verify your account by clicking on the link sent to the given email-id.

3. Create a new container in the Dashboard by clicking on 'New Container'

4. Select Python and click on create

5. Once the container is created, an IDE workspace will open.

6. Create new project by clicking on new folder option and name it project

7. Create a new file named '2.py' by clicking on new file option :

8. Type the following code in the file :

```
def program6():
    cnt = 0
    word_search = input (" Enter the words to search : ")
    with open ("/home/cabox/workspace/merge.txt", "r")
    as f1 :
        for data in f1 :
            words = data.split ()
            for word in words :
                if (word == word-search):
                    cnt + = 1
    print(word_search, "found", cnt, "times from the
    file")
    return cnt
cnt1 = program6 ()
with open ("/home/cabox/workspace/ result.txt","w")
as f3:
    f3.write ( str (cnt1))
```

9. Save the file '2.py'

10. Create a file named 'merge.txt' using new file option.

11. In 'merge.txt' file, insert the following data:

    Welcome to Codeanywhere platform.
    We welcome our developers.
    Welcome to your personal python container.

12. Save the file 'merge.txt'. Copy the path of this file and paste it in the open() function of the code.

13. ~~In the console~~ Also paste the path of the 'result.txt' file in the second open() function of the code similar to 'merge.txt' file.

14. In the (terminal/) console to run the code:

    type : Python 2.py

15. The output will be displayed in the console. and a new file result.txt is created inside which the output will be written.

OUTPUT 1 :

```
Enter the words to search : Welcome
Welcome found 2 times from the file
```

OUTPUT 2 :

```
Enter the word to search : welcome
welcome found 1 times from the file.
```

Database program : (Code Anywhere)

Steps :

1. Create an account in Code anywhere by providing your email-id and password through sign up option.

2. Verify your account by clicking on the link sent to the given email id.

3. Create a new container in the dashboard by clicking on 'New Container'.

4. Select Python and click on create.

5. Once the container is created, an IDE workspace will open.

6. Create new project by clicking on new folder given option and name it 'Project'.

7. Create a new file named 'dbprogram.py' by clicking on new file option.

8. Open 'new tab' and go to official website of Codeanywhere Then click on 'Dashboard' and go to 'Advanced Topics' Now click on 'Installing MySQL'.

   To install MySQL in your container, you have to run the following commands in the IDE terminal:

```
Sudo apt-get update
Sudo apt-get install mysql-server
Sudo service mysql status
```

9. Once the Installation is done, type the following command

```
mysql --user = root -- password = root
```

10. Now create a new database named 'COURSEMATCH' using the following command :

    ~~Create~~ ~~COURSEMATCH~~;

```
create database COURSEMATCH;
use COURSEMATCH;
```

11. Now create new table in the database using the following command :

```
create table course_details ( c_id varchar(10),
 stream varchar(5), c_name varchar(10));
```

12. Insert values into the table using the following command :

```
insert into course_details values ('01', 'ISE', 'CC')
```

13. To display the table , use the following command :

```
Select * from course_details;
```

14. To describe the table , use the following command :

```
desc course_details;
```

15. Prerequisites to run the program :

→ Download mysql - connector - python module

```
pip install ~~mysql~~ mysql-connector-python
```

→ Download pandas module

```
pip install pandas.
```

16. Now , in the file 'dbprogram.py' , write the following code :

```
import os
import platform
import mysql.connector
import pandas as pd
mydb = mysql.connector.connect (host = "localhost",\
                                user = "root", \
                                passwd = "root",\
                                datbase = "COURSEMATCH
```

```
mycursor = mydb. cursor()
def  course_ Insert ( ):
     L = ( )
     c-id = int ( ignput (" Enter the course id : "))
     L.append ( c_id )
     Stream = input ("Enter stream name : ")
     L. append ( e~~name~~ stream )
     c-name = input(" Enter courses in this stream : ")
     L.append ( c_name )
     course = ( L )
     sql = " insert into course_details (c_id , stream,
     c_name) values (·/·s,·/·s,·/·s) "
     mysql.execute (sql , cOurse)
     mydb. commit ( )
def  cView ( ):
     print (" Select the Search criteria : ")
     print (" 1. c_id ")
     print (" 2. Stream ")
     print (" 3. All ")
     ch = int (input (" Enter the choice : "))
     if ch = = 1 :
          s = int(input ("c_id : "))
          c = (s, )
          sql = "select * from course_details where
          c_id = ·/·s "
          ~~mycoursor~~ mycursor. execute (sql, c)
     elif ch = = 2 :
          s = input (" Enter Stream Name : ")
          n = (S, )
          sql = " select * from course_details where
          Stream = ·/·s "
          mycursor.execute (sql,n)
```

```
    elif  ch==3 :
        sql = " select * from course_details "
        mycursor.execute (sql)
    res = mycursor.fetchall ()
    print (" The course details are as follows :")
    print (" Course_id , Stream_Name, Course_Oppertunities"
    for x in res :
        print(x)
def removeCourse () :
    c_id = int(input(" Enter the course_id of the course
    to be deleted : "))
    ci = (c_id , )
    sql = " Deleted from course_details where c_id=%s"
    mycursor.execute(sql, ci)
    mydb.commit()
def MenuSet () :
    print (" Enter 1 : To Add course ")
    print (" Enter 2 : To view course")
    print ( " Enter 3 : To Remove course ")
    try :
        userInput = int (input (" Please select an above
        option : "))
    except : ValueError :
        exit ("\n Hey ! That's not a number! ")
    else :
        print("\n")
        if (userInput ==1) :
            Course_Insert ()
        elif (userInput ==2) :
            cView()
        elif (userInput ==3) :
            removeCourse ()
    else :  print ( " Enter correct choice : . . " )
MenuSet ( )
def  runAgain ( ) :
    runAgn = input ( "\n want to run again Y/n : ")
    while (runAgn.lower ( ) = = 'y' ) :
        if ( platform.system ( ) = = "Windows " ) :
            print ( os.system ( 'cls' ))
        else :
            print ( os.system ( 'clear' ))
        MenuSet ( )
        runAgn = input ( "\n want to run again Y/n : ")
runAgain ( )
```

17. In the console/terminal to run the code (type : python dbprogram.py

18. The output will be displayed.

16/5/2022

Git hub

```
git init
git clone ________ (copy from github)
git config user.name ReshmaRaoCH
git config user.email 1da19is031.cs@drait.edu.in
git add CCLAB
git status -s
git log
git commit -m "git"
```

```
touch .gitignore
git checkout -b B
git push origin B.
```

Program - Github

Steps

1. Create an account on Github by providing your email-id and password, and give a username.

2. Verify your account by entering the code sent to your email id.

3. To create a repository on Github, click on Creat Repository option. Enter the repository name and click on create repository.

4. Open new tab and go to 'git-scm.com', to download Git for Windows.

5. Create a folder on your PC with few files and directories.

6. Right click on the folder created and click on 'Git Bash Here' option.

7. Type the command: git init, to create an empty Git repository in specified directory (folder created)

8. To clone the repository created on ~~the~~ Github onto the local machine, first copy the url from

Git hub and then type the following command on the Git Console,

```
git clone <repo>
```

Here, <repo> is the copied url (via HTTP or SSH)

9. Type the commands:

```
git config user.name <name>
git config user.email <email>
```

, To define author name and email to be used for all commits in the current repository.

10. To add a file or directory to your repository before next commit, type the command:

```
git add <file>
          or
        <directory>
```

11. To list which files are staged, unstaged and untracked, type the command

```
git status -s
```

12. To commit the files or directory, to save changes permanently, type the command

```
git commit -m "<message>"
```

13. To get the commit history, type,

```
git log
```

14. To push the files stored in the folder on the local machine to remote github repository, type the command,

```
git push <repo> <branch>
```

repo → url (via HTTP or SSH)

It creates a named branch in the remote repository if it doesn't exist.

Once, you click enter, we get a dialog box asking to sign in with a code, click on it.

A url with code will be displayed.

Copy the code and click on the url.

The url takes you to a device activation page and paste the code there. Then click on 'Authorize GitCredentialManager'

Now the file is pushed onto the Github remote repository and can be accessed anywhere with the ~~logr~~ internet and login credentials.

15. To create a git ignore files, type the command

```
touch .gitignore
```

16. Type the filename along with extension inside the gitignore file created, and these files will be ignored when pushed to the repository.

17. To check the branch in the Console, type:

```
git checkout -b B
```

Now the branch is B.

18.

```
git push origin B
```

A new branch named 'B' is created in the github repository.

Virtual Box

Virtual box is a general-purpose full virtualizer for x86 hardware, targeted at server, desktop and embedded use.

Steps

1) Go to Oracle virtual box on you web browser. Click on 'downloads' option.

2) Unda VirtualBox 6.1.34 platform packages, Click on 'Windows hosts', VM is downloaded

3) A dialog window appears, click on Next and then 'Finish'.

4) After installation, click on New on Oracle VM and Name your file.

5) Set the Memory size as required.

6) Set hard disk by clicking on 'create a virtual hard disk now'

7) Click on Settings, Go to storage,
Click on empty.
Click on the disc symbol at the right corner of the ~~to~~ window and click on 'Choose disk file'.
~~8)~~ Select the folder containing the operating system to be virtually used.

8) Double click on the ~~file~~ machine you created in step 4 to run the machine (power on)

9) To create a directory, use the command

```
mkdir CC
```

10) To list all files use the command

```
ls
```

11) To change directory, use the command

```
cd CC
```

12) To create a new file,

```
cat > 'file_name'
This is my first file
```

13) To remove a file,

```
rm 'file_name'
```

14)
```
# wc 2        // 2 → filename
```

o/p :- 2 6 28 2

This command tells how many lines, words and characters are there in a file.

15) Grep command :-

To look for strings in the file

```
# grep -i "this" 2
```

o/p :- This is Unix command

16) pwd (present working directory

This command tells you where you currently are (directory)

17) To view file contents,

```
cat 3        // 3 → file_name
```

o/p :- Cloud Computing

18) man

'man' command → online reference manuals

19)
```
mv 3 4
```

This command moves/renames file named 3 to 4

20) du o/p : 4

This command displays number of bytes occupied by the contents of the directory.

21) cal

Displays the calender

22) date

Displays current date and time

o/p :- Mon May 30 15:16:11 UTC 2022

23) rmdir CC

To remove the directory