# Introduction of this project

This project is to develop a vacation vaccine app which customers can book appointments to gain the vaccines required before they travel. This app attempts to let customers,

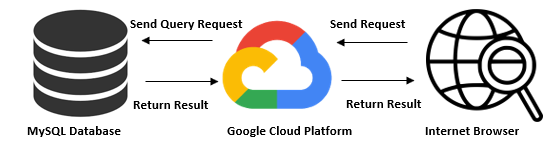
* To view the vaccine required the country where the customers travel to,
* To **C**reate(book) a new appointment,
* To **U**pdate(change) an existing appointment,
* To **R**etrieve and view the appointment history,
* To **D**elete an existing appointment
* To **C**reate, **U**pdate, **R**etrieve and **D**elete user account.

# Goal of this project:

To create a web app based on MySQL server, Python and Flask that can accept the users’ input/request to create, retrieve, update and delete records in database.

# Structure of the project

The structure of this project is shown in the next figure.



# Image result for database symbol Image result for google cloud platform symbols Image result for browsersymbols

**Internet Browser**

**Google Cloud Platform**

**MySQL Database**

**Send Request**

**Return Result**

**Send Query Request**

**Return Result**

# Interface of web app

Web interface includes the following 6 HTML files,

* Login page: This page is for user to login. Users have to provide their register email address and password.

**For a new user**

* Signup page: for new users, they can register and create their accounts in this page.
* Update page: Users can update their accounts details in this page.

**For a return user, redirected to actions**

* Actions page: Once the users login successfully, they will be directed to actions page which enables them to,
  + Update their register account details
  + Make a vaccine appointment
  + Logout
* Booking page: The page is the key component of this project. In this page, users can
  + **C**reate a new appointment.
  + **R**etrieve their appointment records.
  + **U**pdate an existing appointment. Users will be redirected to another URL to view the selected appointment details, change the contents, and update the record. and
  + **D**elete an existing appointment.
* Updatebooking: This page allows user to
  + View the selected appointment details, change the appointment and update the appointment. Or
  + If they change their minds to go back booking page.

Each URL corresponds to one route with the same name of the html file.

The workflow of above can be depicted by the next figure,

**login**

**signup**

**updatebooking**

**booking**

**actions**

**update**

**New User**

**Return User**

# Project folder structure

# Route function

|  |  |  |
| --- | --- | --- |
| Function | What it does | What it cannot do |
| Login | Login:   * Validate if email address is the correct format * Validate if user is a returned or new user   Create New Account:   * Redirect user to signup page and auto fill email field if user input an valid email address in login page | * If a returned user input his/her email address and click create new account, the user will be redirected to signup page. But the user won’t be able to create a new account using an existing email. * Change passwords if user forget their passwords. |
| Actions | This page has three buttons. Each button will redirect users to different pages.   * Update Your Account * Manage Your Appointments * Log out   This page also displays all the appointment booked by the users if any. | NA |
| Booking | This page allows users to management their appointments by   * Create an Appointment: To create a new appointment, users select the country here they travel to, then click Retrieve Vaccine to populate the vaccine drop-down list with vaccine associated to the selected country. Then select the vaccine. Then select the date. The selected date must be a working day (Monday to Friday) and later than today. Then select time slot. If any of them is incorrectly selected. There will be an error message to remind users to reselect.   To delete or update a previous booked appointment, users select one of the existing appointments from Appointments drop-down list, then click   * Delete selected Appointment: or * Update selected Appointment which redirect users to another web page   If no appointment is selected, an error message will show.  All the appointments will be to by the users before and after users’ options and reflect updated result accordingly. | Ideally, the appointments should be seen in different status, such as expired, executed and to be fulfilled. But at this moment all the appointments will be treated as same in respect of database point of view.  Further, the users should be able to click the appointment they want to see or change by clicking the appointment directly. But at this moment, there are some undergone process to follow.  In addition, when the country is selected, the vaccine drop-down menu should automatically populate. Now it require to retrieve vaccine button to trigger the change. |
| Updatebooking | This is similar to the process to Create an Appointment in Booking. The difference the country, vaccine, date and time slot is set based on the previous booked record. Users can change what they to change and click   * Update this Appointment to update the appointment or * Cancel, if they change their mind.   Both buttons will redirect them back to booking page. A message in booking page will also show how the appointment has been updated. |  |
| signup | This page is for the new users to create their account before they can make a vaccine appointment. All the field are required input (i.e cannot be null). The page also validates email format, length of the password (6 to 25 characters) and if the password is same as the re-type password.  If the email is already exits in the database, the users will be redirect back to login page. | It cannot check and advise if the password is strong or weak. |
| Update | This page allows users to   * Update your Account, change the all the account information except password. * Delete your Account, users can only delete their accounts if they don’t have appointments. If they do, they cannot delete their accounts. and * Make an Appointment: redirect to booking page. |  |

# Deploy process (CI??)

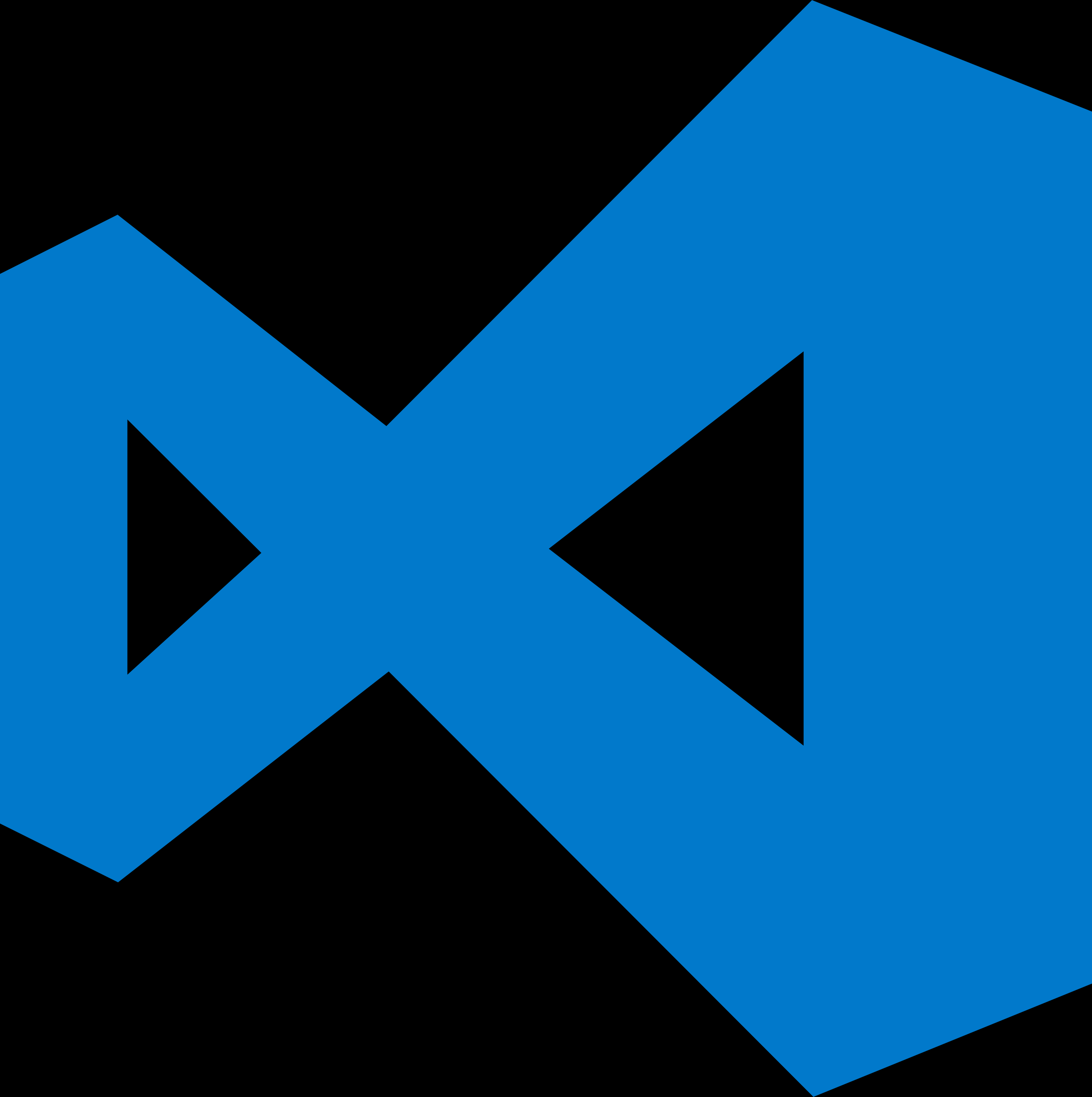
The code is created, developed, and tested in a local git repo.

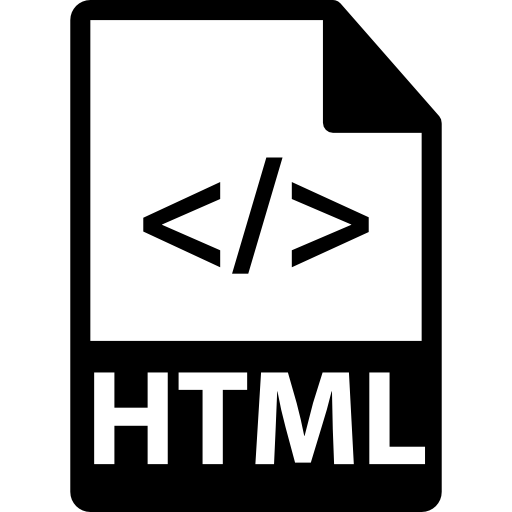
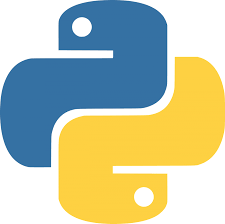
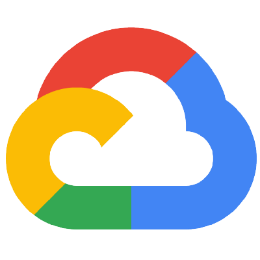
* Git is used to control the revision of the codes.
* Working environment: UBUNTU environment (20.04) by WSL(Window Subsystem Linux).
* MySQL server is installed.
* Python 3.6

Each commit of the codes was pushed to git hub.

The project in Git Hub will fetch to GCP instance to test if the app can be accessed from Internet.



# Developing process

* The project starts from a simplified table which has only three attributes. Use this table to learn how to use HTML form, FLASK and SQLAlchemy. Learn how to use form to receive the client’s request and data and use Flask to perform the corresponding actions to the request and display the execution result
* After I have a better understanding of this framework, more complicated relational tables.
* Develop codes to response to all the requests.
* Test the code.

# Challenges

There are several challenges while developing this project,

* Environment: There are setup issues in many occasions. For example, UBUNTU revisions are different ion the local and instance. Different versions of UBUNTU have different issues with the other setup. It would be great if there is recommended setup at the very beginning of the training.
* Rational database with foreign key. It took me some time to figure out the naming of backreef in db.Relational function.
* There are variables passed between different routes. In the beginning I use session[‘variable’] to pass the variables. They work but also result in some problem, especially in performing pytest. Thanks for Luck’s suggestion to use dynamic URL.
* DateTime comparison. As mention above when the users make an appointment, they need to select a future working day. To make this work, the selected date is compared to another day (today). Due to the date format, the comparison is not that straightforward. Eventually, the dates were converted to an integer for the comparison purpose.
* Another datetime issue is caused when I run Pytest. Change the code to

bookings.date= datetime.now()

* Pytest. As I modified my codes at the last minute, still try to figure out how it works.
  + When I use session[‘var’] to pass the variable between routes, pytest returns error, something like KeyError: id. I suspect that is caused by session variable.
  + Luke suggested me to use Dynamic URL. Then I pass the variable by using ‘url/<email>/<id>’. I use the following code to perform test,

self.client.post(

url\_for('signup'),

data = dict(email ="John.doe@company.com"),

follow\_redirects=True

)

However, I receive the following error:

BuildError: Could not **build url for endpoint 'signup'**. Did you forget to specify values ['email']

* + I did assign email variable. I change the code to

self.client.post(

url\_for('signup', email ="John.doe@company.com"),

follow\_redirects=True

)

This code works.