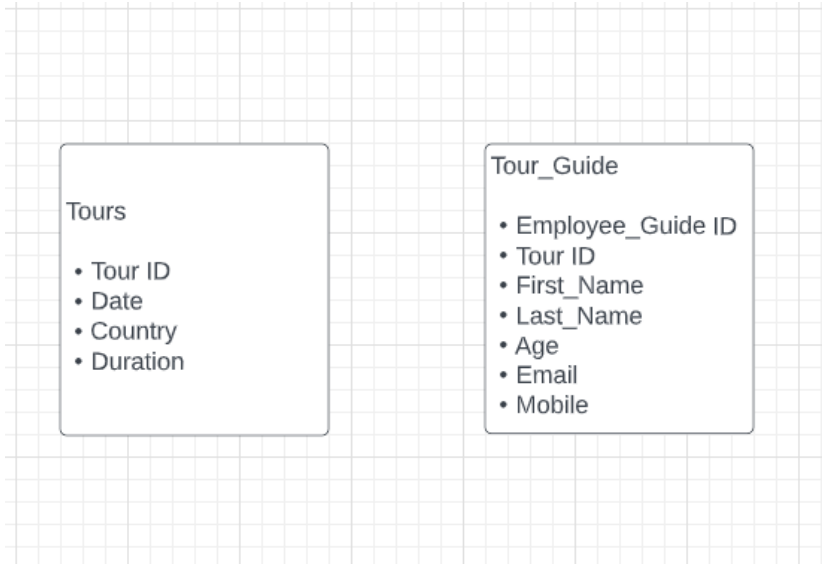


# DFE Final Project

Created a web application based database via a linux based virtual machine. The create a pipeline using Jenkins and Github.

ER Diagram:



The ER disgrace shows a one-to- many relation as there is one tour guide per tour however a tour guide can have many tours allocated to them.

DataBase:

```
tour_app.py x SQLTools Settings mydata.db
tour_app.py > ...
1 from flask import Flask,render_template, redirect, url_for, request
2 from flask_sqlalchemy import SQLAlchemy
3
4 app = Flask(__name__)
5 app.config['SQLALCHEMY_DATABASE_URI'] = 'sqlite:///mydata.db'
6 db = SQLAlchemy(app)
7
8 class TourGuide(db.Model):
9     id = db.Column(db.Integer, primary_key=True)
10    first_name = db.Column(db.String)
11    last_name = db.Column(db.String)
12    age = db.Column(db.Integer)
13    email = db.Column(db.String)
14    mobile = db.Column(db.String)
15    tour = db.relationship('Tour', backref='all_tour')
16
17
18 class Tour(db.Model):
19     id = db.Column(db.Integer, primary_key=True)
20     date = db.Column(db.String)
21     duration = db.Column(db.String)
22     city = db.Column(db.String)
23     country = db.Column(db.String)
24     tourguide = db.Column(db.ForeignKey('tour_guide.tourguide_id'))
25
26
27 db.create_all()
28 g1 = TourGuide(id=1, first_name='Jessica', last_name= 'Sanchez', age= 25, email= 'jessica@eurotours.com', mobile='07963784932'])
29 g2 = TourGuide(id=2, first_name='Abby', last_name= 'Longhart', age= 30, email= 'abby@eurotours.com', mobile='07986536374')
30 g3 = TourGuide(id=3, first_name='Jamie', last_name= 'Bennett', age= 34, email= 'jamie@eurotours.com', mobile='07738564783')
31 g4 = TourGuide(id=4, first_name='Tomas', last_name= 'Smith', age= 28, email= 'tomas@eurotours.com', mobile='07456819282')
32 db.session.add(g1)
33 db.session.add(g2)
34 db.session.add(g3)
35 db.session.add(g4)
36 db.session.commit()
37
38 t1 = Tour(id=1, date='12-10-2022', duration='5 days', city='Spain', country='Barcelona', tourguide= 1)
39 t2 = Tour(id=2, date='01-12-2022', duration='7 days', city='Budapest', country='Hungary', tourguide= 2)
40 t3 = Tour(id=3, date='24-03-2023', duration='14 days', city='Geneva', country='Switzerland', tourguide= 3)
41 t4 = Tour(id=4, date='20-05-2023', duration='3 days', city='Berlin', country='Germany', tourguide= 4)
42 t5 = Tour(id=5, date='20-05-2023', duration='2 days', city='Paris', country='France', tourguide= 5)
43 t6 = Tour(id=6, date='15-07-2023', duration='8 days', city='Venice', country='Italy', tourguide= 6)
44 db.session.add(t1)
45 db.session.add(t2)
46 db.session.add(t3)
47 db.session.add(t4)
48 db.session.add(t5)
49 db.session.add(t6)
50 db.session.commit()
```

I created a database in visual code studio using my sqlite. I unfortunately wasn't able to connect the database to my virtual machine.

This database is suppose to produce two tables one for the tours where the foreign key is the tour-guide and another table for the tour guide. I have also created two separate classes with variables that link them together.

I then created a simple database using the the code I have presented a print screen of on the left.

## Pipeline:

I created a Jenkins pipeline via a Linux virtual machine. Below is a screen shot of the installation process.

```
jenkins_install
jenny@TourAppDfE:~/jenkins_install$ vi jenkins_install
jenny@TourAppDfE:~/jenkins_install$ sudo chmod 700 jenkins_install
jenny@TourAppDfE:~/jenkins_install$ ./jenkins_install
./jenkins_install: line 4: [: too many arguments
updating and installing dependencies
Hit:1 http://azure.archive.ubuntu.com/ubuntu focal InRelease
Hit:2 http://azure.archive.ubuntu.com/ubuntu focal-updates InRelease
Hit:3 http://azure.archive.ubuntu.com/ubuntu focal-backports InRelease
Hit:4 http://azure.archive.ubuntu.com/ubuntu focal-security InRelease
Hit:5 https://download.docker.com/linux/ubuntu focal InRelease
Reading package lists... Done
Building dependency tree
Reading state information... Done
14 packages can be upgraded. Run 'apt list --upgradable' to see them.

WARNING: apt does not have a stable CLI interface. Use with caution in scripts.

Extracting templates from packages: 100%
configuring jenkins user
downloading latest jenkins WAR
% Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
             Dload  Upload  Total   Spent    Left     Speed
100 245 100 245    0     0 1376      0 --:--:-- --:--:-- --:--:-- 1376
100 228 100 228    0     0 1036      0 --:--:-- --:--:-- --:--:-- 1036
  0  0  0  0    0     0      0      0 --:--:-- --:--:-- --:--:--    0
100 89.2M 100 89.2M    0     0 24.1M      0 0:00:03 0:00:03 --:--:-- 31.9M
setting up jenkins service
Created symlink /etc/systemd/system/multi-user.target.wants/jenkins.service → /etc/systemd/system/jenkins.service.
waiting for initial admin password
waiting for initial admin password
waiting for initial admin password
waiting for initial admin password
waiting for initial admin password
waiting for initial admin password
waiting for initial admin password
initial admin password: 21c7847525114095b7c88f186df595b4
jenny@TourAppDfE:~/jenkins_install$
```

```
57 lines (53 sloc) 782 Bytes
1 pipeline
2 {
3     agent none
4     stages
5     {
6         stage('Build Stage')
7         {
8             agent any
9             steps
10            {
11                echo 'This is Build part'
12                sh 'chmod 777 build.sh'
13            }
14            sh './build.sh'
15        }
16    }
```

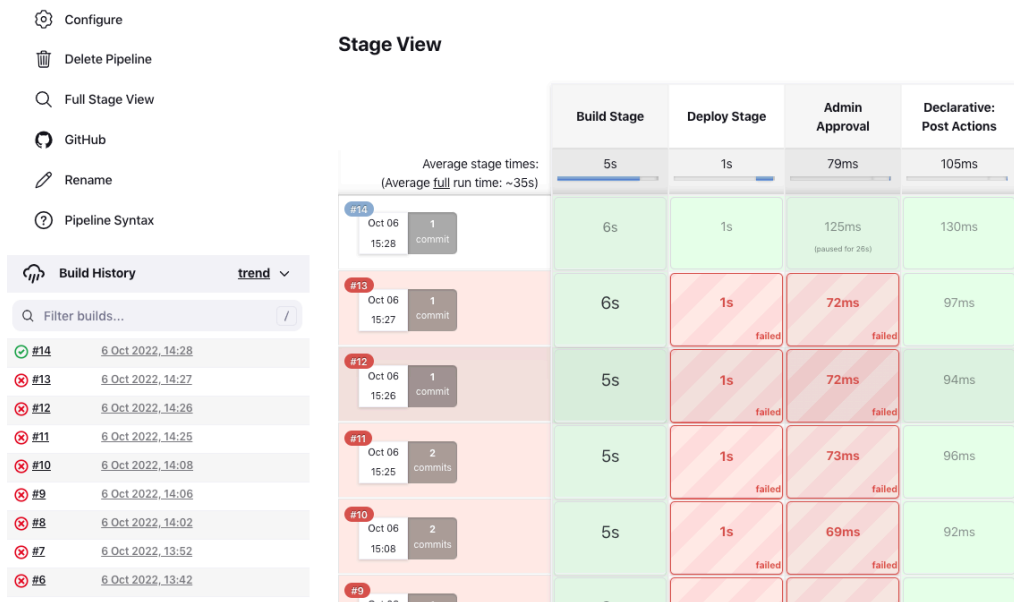
```
16    }
17 }
18 stage('Deploy Stage')
19 {
20     agent any
21     steps
22     {
23         echo 'This is Deploy part'
24         sh 'chmod 777 run.sh'
25         sh './run.sh'
26     }
27 }
28 }
```

```
    }
    }
    stage('Admin Approval')
    {
        steps
        {
            input "Does the staging environment look ok?"
        }
    }
}
```

In this code we have established that this is the build stage of the pipeline. The echo command simply prints out 'This is Build part' when building in Jenkins. The sh 'chmod 777 build.sh' allows full access permissions to the file build.sh and the last command sh './build.sh' runs the file. The build.sh file contains a docker build command which builds Docker images from a Dockerfile which contains the App.py file.

Very Similar to the build stage we have the Deploy stage of the pipeline. The commands are also very similar and executed the same way except this time we have a run.sh file instead and this contains a docker run command dictating which port number the app will run on.

The final step of the pipeline is simply approving the completion of the pipeline.



This screen shot illustrates a working Jenkins Pipeline