Documentation about URL Shorten Website using Flask

In this project, I build a *URL shortener*, a service that takes any URL and generates a shorter, more readable version like **bit.ly**. For this purpose, we're going to use Flask. The application will allow users to enter a URL and an optional custom short id and generate a shorter version.

Step 1 — Setting Up Dependencies:

In this step, I activate our Python virtual environment and install Flask using the pip package installer. To manage the virtual environment, I going to use *Pipenv*. If you haven't installed it, you can use the below command to install it:

```
$pip install pipenv
```

After Pipenv is installed successfully, we can create and activate our virtual environment using the command:

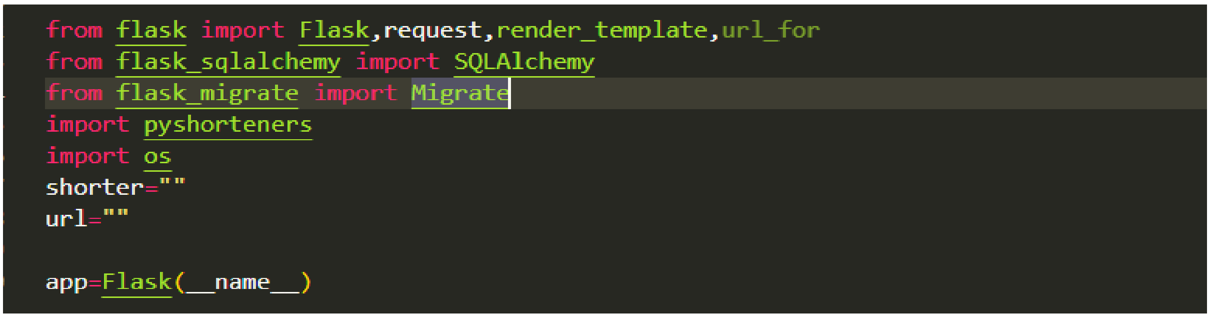
```
$ pipenv shell
```

Once our virtual environment is created, we can install the required dependencies using the command:

```
$ pipenv install Flask Flask-Migrate Flask-SQLAlchemy
```

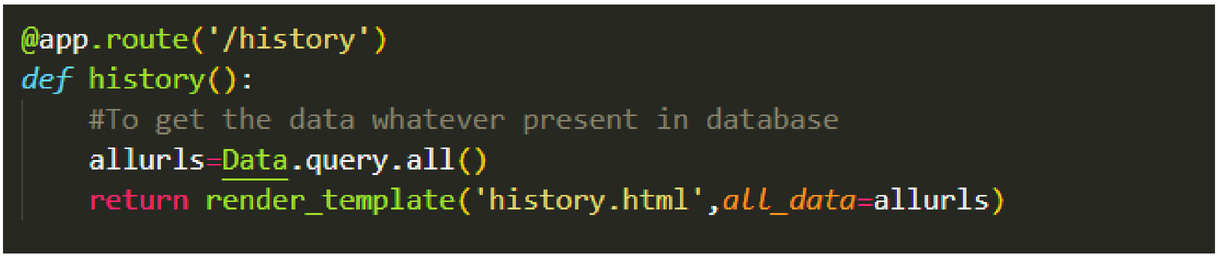
Step 2 — Setting Up Project:

This project will be existing in a package. In this step first a python file called **app.py**, in this file(app.py) contains the all python code for this project. At first, I create a flask object name app.



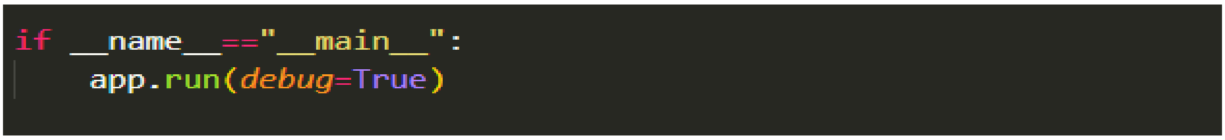
```
from flask import Flask,request,render_template,url_for
from flask_sqlalchemy import SQLAlchemy
from flask_migrate import Migrate
import pyshorteners
import os
shorter=""
url=""

app=Flask(__name__)
```

After creating the app object I create some routes for go to a different page in this application.



```
@app.route('/history')
def history():
    #To get the data whatever present in database
    allurls=Data.query.all()
    return render_template('history.html',all_data=allurls)
```

After creating app and route It is time to run our application. To run this application run the following code.

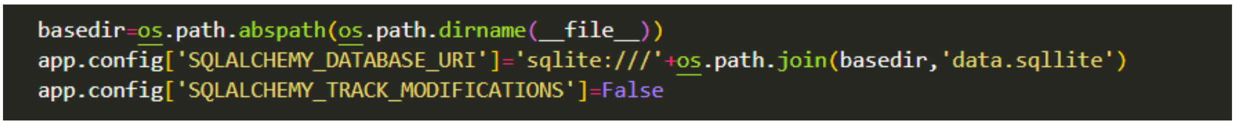


```
if __name__=="__main__":
    app.run(debug=True)
```

This application running on default port no i.e 5000. The created URL for this application is **http://127.0.0.1:5000** or **http://localhost:5000**.

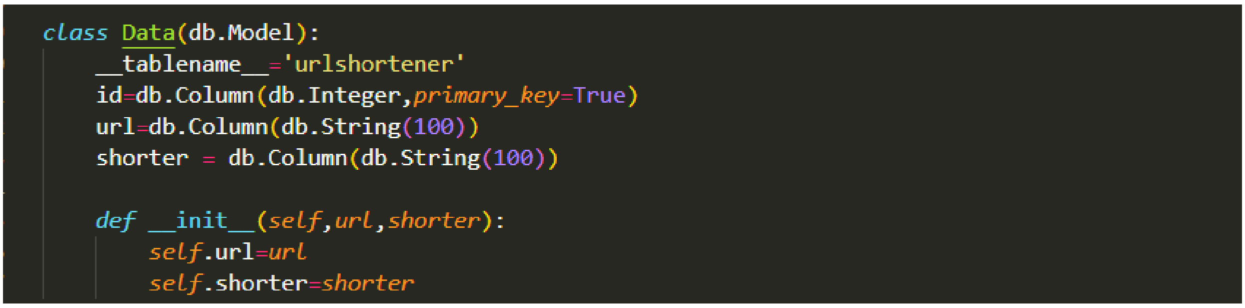
Step 3 — Setting Up Database:

In this step, at first, config our project by running following the code.



```
basedir=os.path.abspath(os.path.dirname(__file__))
app.config['SQLALCHEMY_DATABASE_URI']='sqlite:///'+os.path.join(basedir,'data.sqllite')
app.config['SQLALCHEMY_TRACK_MODIFICATIONS']=False
```

After config ,we create a database for our application by-



```
class Data(db.Model):
    __tablename__='urlshortener'
    id=db.Column(db.Integer,primary_key=True)
    url=db.Column(db.String(100))
    shorter = db.Column(db.String(100))

    def __init__(self,url,shorter):
        self.url=url
        self.shorter=shorter
```

We can then make use of Flask-Migrate commands to migrate the database with the new tables. The commands to be used are :

flask db init — to initialize the database at the beginning.

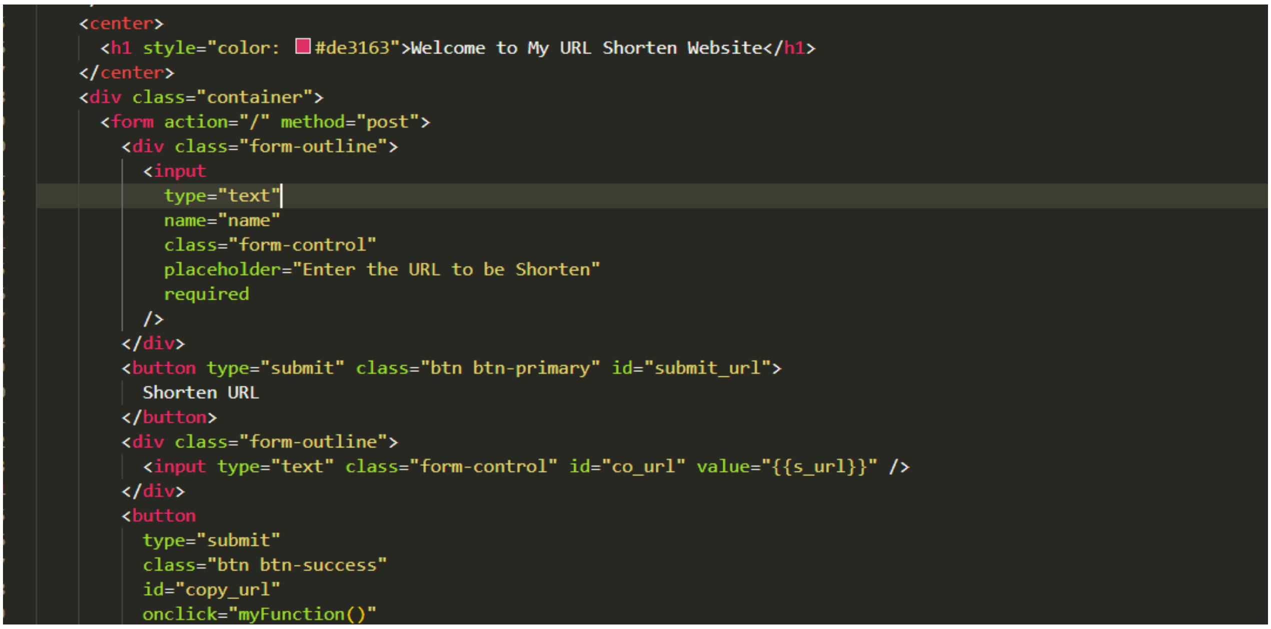
flask db migrate — to migrate the new changes to the database.

flask db upgrade — to upgrade our database with the new changes.

Step 4— Creating the Index Page for Shortening URLs:

In this step, we will create a Flask route for the index page, which will allow users to enter a URL that we then save into the database. This route will use the custom short id provided by the user or generate one on its own, construct the short URL, and then render it as a result and a user can copy his/her own created shorten URL link.

Now, we need to create a template for the index page that will be served by the index route. This template will have a simple form where user can input the original URL and custom short id(optional) and submit it. But we’ll not create index.html directly. We can make use of the Template Inheritance concept in Jinja2.

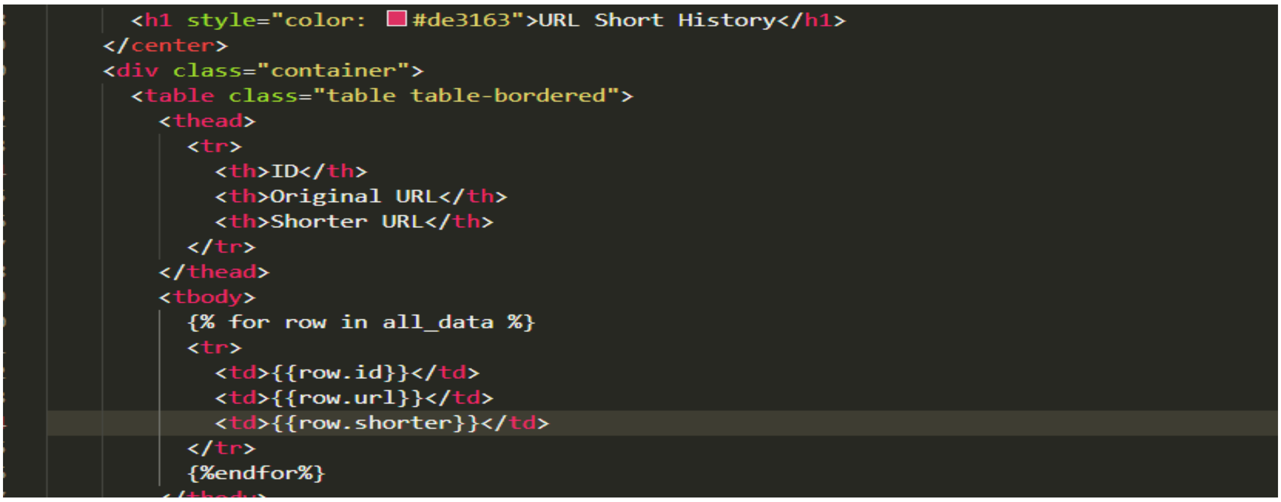


```
<center>
  <h1 style="color: ■#de3163">Welcome to My URL Shorten Website</h1>
</center>
<div class="container">
  <form action="/" method="post">
    <div class="form-outline">
      <input
        type="text"
        name="name"
        class="form-control"
        placeholder="Enter the URL to be Shorten"
        required
      />
    </div>
    <button type="submit" class="btn btn-primary" id="submit_url">
      Shorten URL
    </button>
    <div class="form-outline">
      <input type="text" class="form-control" id="co_url" value="{{s_url}}" />
    </div>
    <button
      type="submit"
      class="btn btn-success"
      id="copy_url"
      onclick="myFunction()"
```

For full code please visit at- https://bit.ly/3AjKoQb

Step 5— Creating the History Page to show all Shortening URLs and Original URLs.

In this step, we will create a Flask route for the history page, which will allow users to show all the shorten URLs, and original URLs.

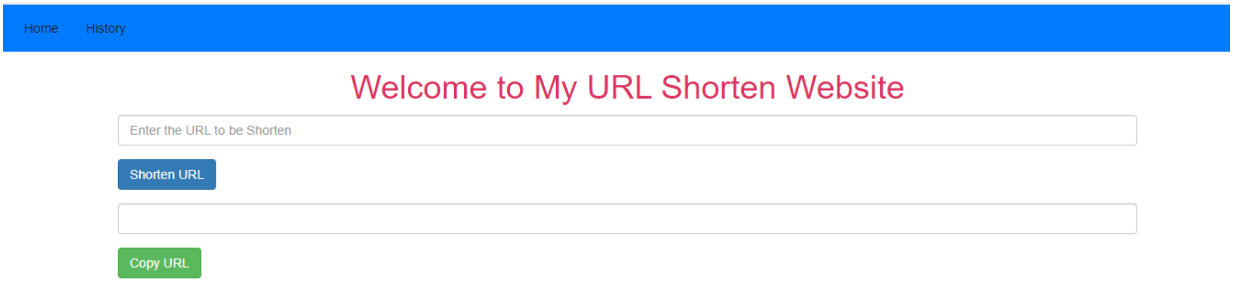


```
  <h1 style="color: ■#de3163">URL Short History</h1>
</center>
<div class="container">
  <table class="table table-bordered">
    <thead>
      <tr>
        <th>ID</th>
        <th>Original URL</th>
        <th>Shorter URL</th>
      </tr>
    </thead>
    <tbody>
      {% for row in all_data %}
      <tr>
        <td>{{row.id}}</td>
        <td>{{row.url}}</td>
        <td>{{row.shorter}}</td>
      </tr>
      {%endfor%}
```

For full code please visit at- https://tinyurl.com/2ljk9smn

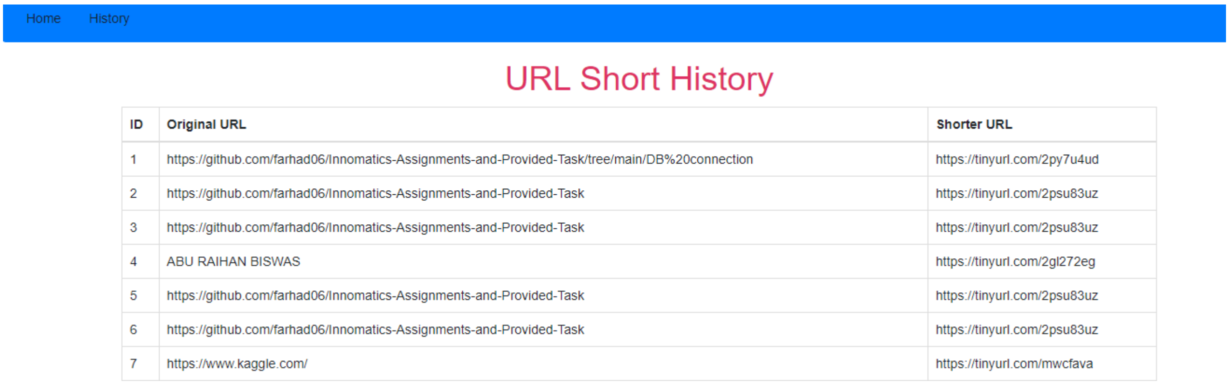
Some Screen Shot of my application:

Index Page:



Welcome to My URL Shorten Website

History Page:



URL Short History

ID	Original URL	Shorter URL
1	https://github.com/farhad06/Innomatics-Assignments-and-Provided-Task/tree/main/DB%20connection	https://tinyurl.com/2py7u4ud
2	https://github.com/farhad06/Innomatics-Assignments-and-Provided-Task	https://tinyurl.com/2psu83uz
3	https://github.com/farhad06/Innomatics-Assignments-and-Provided-Task	https://tinyurl.com/2psu83uz
4	ABU RAIHAN BISWAS	https://tinyurl.com/2gl272eg
5	https://github.com/farhad06/Innomatics-Assignments-and-Provided-Task	https://tinyurl.com/2psu83uz
6	https://github.com/farhad06/Innomatics-Assignments-and-Provided-Task	https://tinyurl.com/2psu83uz
7	https://www.kaggle.com/	https://tinyurl.com/mwcfava

Conclusion

We have created a Flask application that allows users to enter a long URL and generate a shorter version, copy the short URL and user can show the history of all short URLs. We can add more features to this application such as User Authentication, Shortened URLs Statistics, etc.

Full Project available at: https://bit.ly/3C5tQwA