

User Registration with Flask

Revision...



```
from flask import Flask

app = Flask(__name__)


@app.route("/")
def hello():
    return "<h1>Hello</h1>"
```

This was the first flask application we built that has a single page showing Hello on the screen.

HTML Templates

Flask applications usually return messages and it is most of the times in HTML format. So, it is possible to return a whole HTML page from a flask function decorated with route.

Flask projects render html files using `render_template` function that brings HTML file from templates folder and sends it to the client.

- 
- project-folder
 - run.py
 - templates
 - register.html
 - login.html
 - ...

Inside of register.html

```
<!DOCTYPE html>
<html>
  <head>
    <title>Register Now!</title>
  </head>
  <body>
    <form action="/register" method="POST">
      <label for="username">Username:</label> <br />
      <input type="text" id="username" name="username" /> <br />
      <label for="email">Email:</label> <br />
      <input type="email" id="email" name="email" /><br />
      <label for="password">Password:</label> <br />
      <input type="password" id="password" name="password" /><br />

      <button type="submit">Register</button>
    </form>
  </body>
</html>
```

Inside of register.html

```
<!DOCTYPE html>
<html>
  <head>
    <title>Register Now!</title>
  </head>
  <body>
    <form action="/register" method="POST">
      <label for="username">Username:</label> <br />
      <input type="text" id="fname" name="fname" /> <br />
      <label for="email">Email:</label> <br />
      <input type="email" id="email" name="email" /> <br />
      <label for="password">Password:</label> <br />
      <input type="password" id="password" name="password" /> <br />
      <button type="submit">Register</button>
    </form>
  </body>
</html>
```

MUST INCLUDE!!

`action` attribute tells where to send the form data after submit. `method` tells which HTTP method to follow when sending the data. Finally, button type should be `submit` to trigger the form action.

Flask Application

```
from flask import Flask, render_template

app = Flask(__name__)

@app.route('/')
def index():
    return render_template("register.html")

if __name__ == '__main__':
    app.run(debug=True, port=3000)
```

In the base url / you will now see the register.html result.

Result...

Username:

Email:

Password:

Register

Quite simple looking registration page...


Right now, if you click on the Register button you will get a Not Found page error. Because the view for /register is not yet defined.

/register URL View

```
@app.route('/register', methods=['POST', 'GET'])
def register():
    if request.method == 'POST':
        # Code to take care of registration
        pass
    return render_template('register.html')
```

1. `register()` can handle both GET and POST type requests when `/register` URL is accessed.
2. `request` object has `method` property that tells us what kind of request you must handle. So, you can use that to choose what to do when POST request comes and when GET request comes.

Design DB Model



```
from flask_sqlalchemy import SQLAlchemy
from sqlalchemy import Column, Integer, String

db = SQLAlchemy()

class User(db.Model):
    __tablename__ = 'users'

    id = Column(Integer, primary_key=True)
    username = Column(String(20), nullable=False)
    email = Column(String(20), nullable=False, unique=True)
    password = Column(String(200), nullable=False)
```

Design DB Model (model.py)

```
from flask_sqlalchemy import SQLAlchemy
from sqlalchemy import Column, Integer, String

db = SQLAlchemy()

class User(db.Model):
    __tablename__ = 'users'

    id = Column(Integer, primary_key=True)
    username = Column(String(20), nullable=False)
    email = Column(String(20), nullable=False, unique=True)
    password = Column(String(200), nullable=False)
```

Initialize the SQLAlchemy and make an instance that will be used to build the models.

User Model:
Tablename: users
Id: INTEGER PRIMARY KEY
username: VARCHAR(20) NOT NULL
email: VARCHAR(20) UNIQUE NOT NULL
password: VARCHAR(200) NOT NULL

Flask Application Code

```
from model import db, User
```

```
app.config['SQLALCHEMY_DATABASE_URI'] = \n'mysql://username:password@host/db_name'
```

```
db.init_app(app)
```

```
with app.app_context():\n    db.create_all()
```

Import the db and User model from the model file you just created.

Application comes with a config object that holds all necessary variables needed to start the application. You can set the SQLAlchemy database URI and point it to your database.

Finally Initialize the application with the database and create tables and databases.

Registration



```
@app.route('/register', methods=['POST', 'GET'])
def register():
    if request.method == 'POST':
        # Code to take care of registration
        user = User(username=request.form['username'],
                    email=request.form['email'],
                    password=generate_password_hash(request.form['password']))
        db.session.add(user)
        try:
            db.session.commit()
        except:
            return "Something went wrong...", 404

    return render_template('register.html')
```

1. Create `User` object with the form data coming to the request object.
2. Add user to the database using `db.session.add()`.
3. Commit the changes to the databases to modify the existing database.
4. Keep it inside `try-except` block so that when any database error happens due to the type of input, it will show it to the user.

CAUTION!!

SHOULD BE SAME!!!

```
@app.route('/register', methods=['POST', 'GET'])
def register():
    if request.method == 'POST':
        # Code to take care of registration
        user = User(username=request.form['username'],
                    email=request.form['email'],
                    password=generate_password_hash(request.form['password']))
        db.session.add(user)
    try:
        db.session.commit()
    except:
        return "Something went wrong...", 404

    return render_template('register.html')
```

```
<!DOCTYPE html>
<html>
<head>
    <title>Register Now!</title>
</head>
<body>
    <form action="/register" method="POST">
        <label for="username">Username:</label> <br />
        <input type="text" id="username" name="username" /> <br />
        <label for="email">Email:</label> <br />
        <input type="email" id="email" name="email" /> <br />
        <label for="password">Password:</label> <br />
        <input type="password" id="password" name="password" /> <br />

        <button type="submit">Register</button>
    </form>
</body>
</html>
```