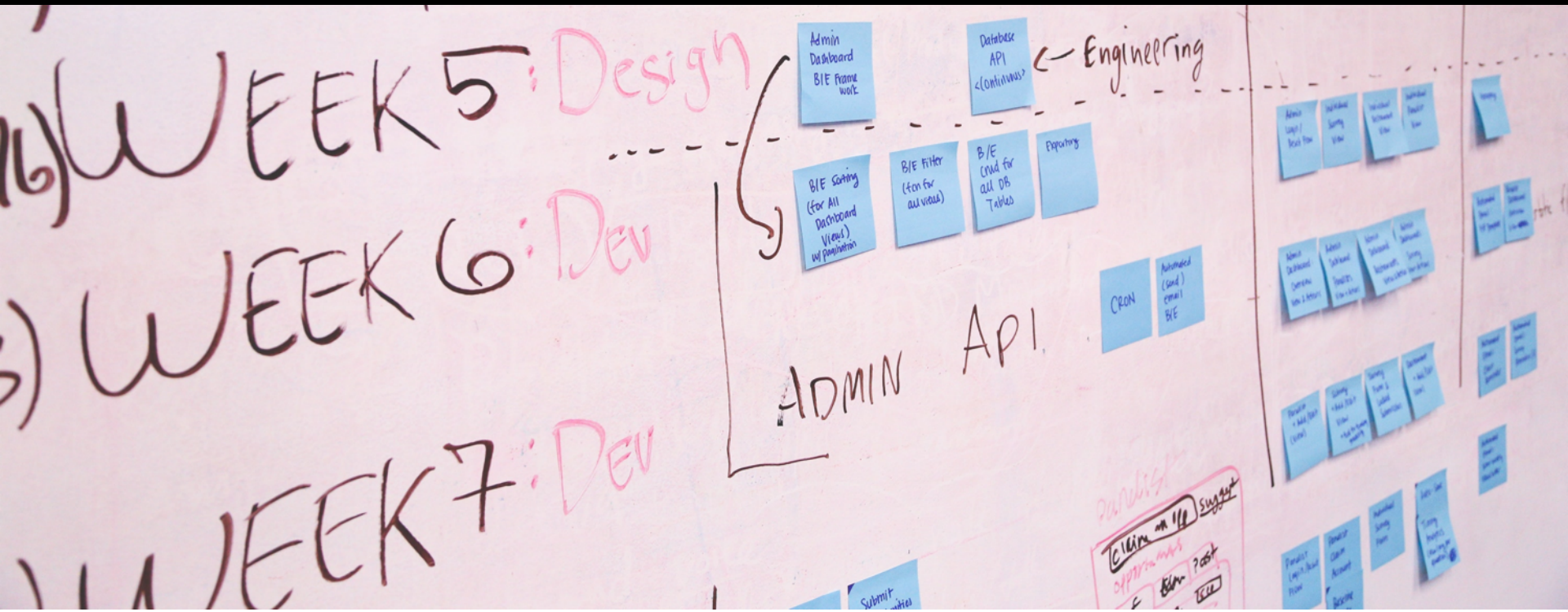


COM5961 DATA DRIVEN PRODUCTS & SERVICES DESIGN:
LESSON 7 - PYTHON WEB PROGRAMMING IN FLASK II

Bernard Suen
Center for Entrepreneurship
Chinese University of Hong Kong

Today's agenda.

1. Group Python modules into a Python Package for integrated app development.
2. Perform DML and DQL database operations (CRUD) in Flask using SQLAlchemy and SQLite.
3. Use of Bootstrap form in a Flask application.
4. Create a login and registration forms with Flask-Login and WTForms modules.
5. Encrypting passwords for improving security.
6. Email notification for authenticating registration signup and password reset using SMTP or API services (To be covered in later lesson).

Quick recap of assignment #4.

Be specific. What you cannot define you cannot control.

Be specific. What you cannot define "concretely", you cannot control.

1. Data sources (with name, url, address, date, etc.)
2. Shortcomings (robots.txt, missing data, garbage characters, type format, etc.)
3. Remedy strategies (integrating alternative sources, cleaning, imputation, UGC tools, interviews, etc.)
4. Additional persona identification and study (supply side persona, additional questions from existing persona)

Goal: timeline for prioritising and scheduling remaining jobs to be done to complete project.

Persona of One

Full-stack Front-end/Back-end Web Development

Browser/ Mobile

Web Server

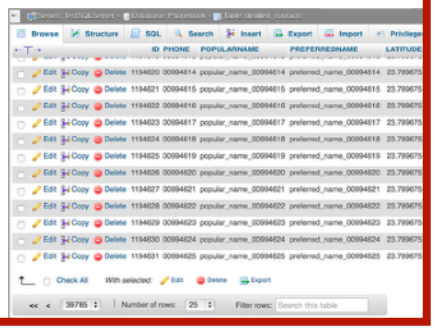
Apache/Nginx

App Server

Python
Custom Coded App

Database Server

SQL Database



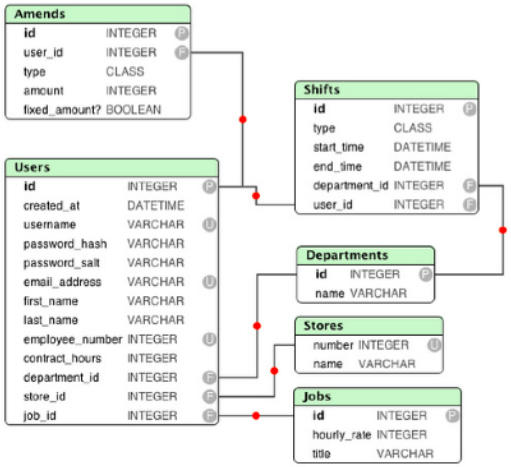
Frontend

Backend

The MVC Pattern for Structuring Application Development

Source: commons.wikimedia.org

Source: Caius Durling (Flickr)



M(odel)
(SQL/NoSQL data models and processing functions)

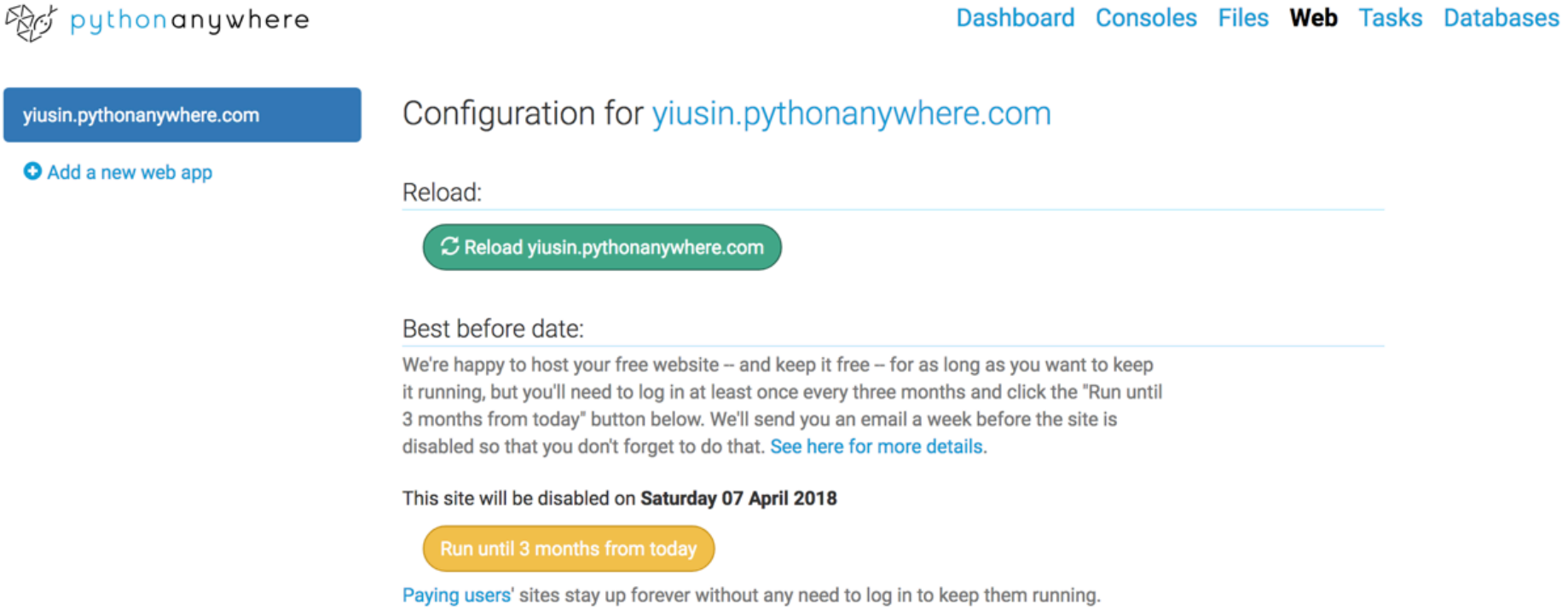
V(iew)
(Flask templates)

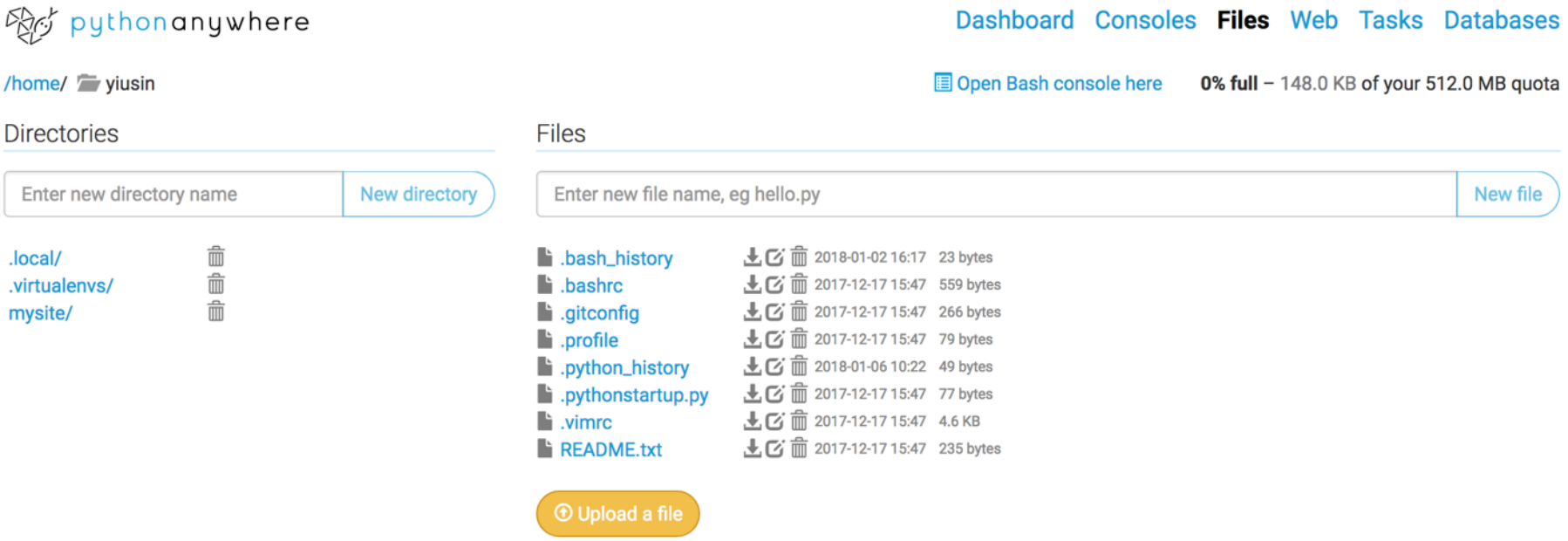
Routing control to pre-defined data and functions

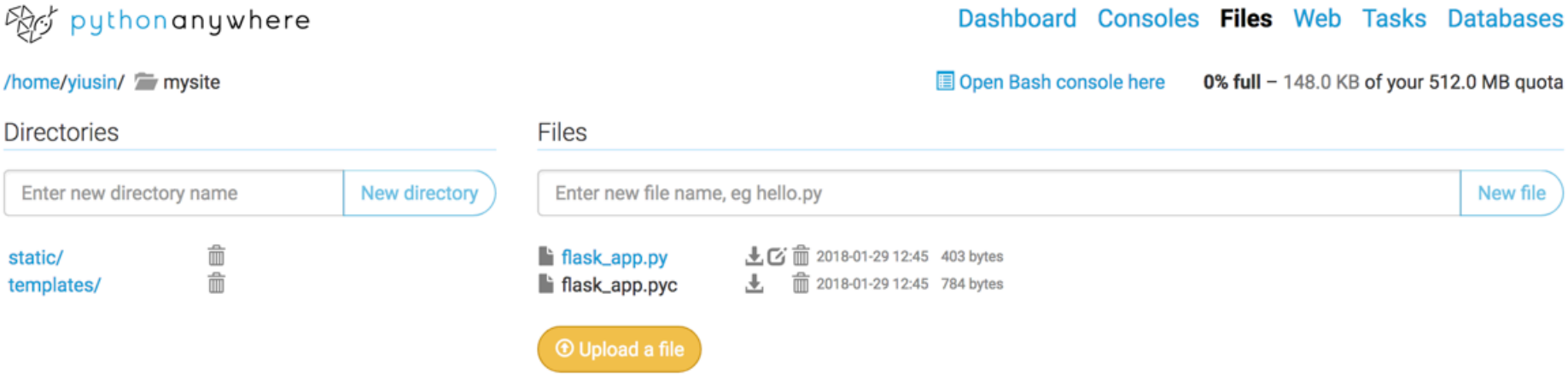
Source: pexels.com

C(ontroller)
(e.g. Routes, rights and authentication)

Python Web Development in Flask: From Jupyter Notebook to PythonAnywhere


pythonanywhere
Dashboard Consoles Files Web Tasks Databases
yiusin.pythonanywhere.com
Add a new web app
Configuration for yiusin.pythonanywhere.com
Reload:
Reload yiusin.pythonanywhere.com
Best before date:
We're happy to host your free website -- and keep it free -- for as long as you want to keep it running, but you'll need to log in at least once every three months and click the "Run until 3 months from today" button below. We'll send you an email a week before the site is disabled so that you don't forget to do that. See here for more details.
This site will be disabled on Saturday 07 April 2018
Run until 3 months from today
Paying users' sites stay up forever without any need to log in to keep them running.


pythonanywhere
Dashboard Consoles Files Web Tasks Databases
/home/ yiusin
Open Bash console here
0% full – 148.0 KB of your 512.0 MB quota
Directories
Enter new directory name
New directory
.local/
.virtualenvs/
mysite/
Files
Enter new file name, eg hello.py
New file
.bash_history 2018-01-02 16:17 23 bytes
.bashrc 2017-12-17 15:47 559 bytes
.gitconfig 2017-12-17 15:47 266 bytes
.profile 2017-12-17 15:47 79 bytes
.python_history 2018-01-06 10:22 49 bytes
.pythonstartup.py 2017-12-17 15:47 77 bytes
.vimrc 2017-12-17 15:47 4.6 KB
README.txt 2017-12-17 15:47 235 bytes
Upload a file


pythonanywhere
Dashboard Consoles Files Web Tasks Databases
/home/yiusin/ mysite
Open Bash console here
0% full – 148.0 KB of your 512.0 MB quota
Directories
Enter new directory name
New directory
static/
templates/
Files
Enter new file name, eg hello.py
New file
flask_app.py
2018-01-29 12:45 403 bytes
flask_app.pyc
2018-01-29 12:45 784 bytes
Upload a file

```
from flask import Flask, render_template

app = Flask(__name__)
@app.route("/")
def hello():
    return "Hello World!"

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

The Controller Component

Use Python Flask Framework for Controlling Access to Web Resources

Flask Development in Jupiter Notebook

```
from flask import Flask

app = Flask(__name__)
@app.route("/")
def hello():
    return "Hello World!"

if __name__ == '__main__':
    app.run('localhost', port=9000)
```

Linking a route to a function.

Python Function

```
def name_of_function(parameter1, parameter2…):
    [ Python variable assignment statements,
      mathematical and logical operations ]
    return result
```

Example

```
def calc_avg(x,y):
    result = x + y
    return result
print("Result = ", calc_avg(10,20)
```

Output

Result=30

Example in Flask

```
@app.route("/blogs")
def blogs():
    list =
    [{'Student': ‘D. Chan', 'Jan': 90, 'Feb': 85, 'Mar': 88},
            {'Student': ‘P. Lee',  'Jan': 72, 'Feb': 75, 'Mar':
    68},
            {'Student': ‘J. Lui',  'Jan': 60,  'Feb': 80,  'Mar':
    50}]
    return render_template('blogs.html', entries = list)
```

Python Module

What is a module?

A module is an external Python file which contains the functions that can be imported into the current Python program to be used.

In [7]:

```
1  from convert_temp import *
2  # Move the following functions into convert_temp module file.
3
4  total_temp = 0
5  total_temp = get_daily_celcius()
6  display_avg(total_temp)
```

```
Enter Sun's temperature in celcius:23
Sun's temperature in Fahrenheit:73.4
Enter Mon's temperature in celcius:24
Mon's temperature in Fahrenheit:75.2
Enter Tue's temperature in celcius:28
Tue's temperature in Fahrenheit:82.4
Enter Wed's temperature in celcius:19
Wed's temperature in Fahrenheit:66.2
Enter Thu's temperature in celcius:17
Thu's temperature in Fahrenheit:62.6
Enter Fri's temperature in celcius:21
Fri's temperature in Fahrenheit:69.8
Enter Sat's temperature in celcius:25
Sat's temperature in Fahrenheit:77.0

Average temperature for the week:72.37
```

jupyter convert_temp.py ✔ 19/04/2021

File Edit View Language

```
1  def convert_temp(temp):
2      f_value = 0
3      f_value = (float(temp) * 9/5) + 32
4      return f_value
5
6  def compute_total_temp(temp, total_temp):
7      total_temp = total_temp + float(temp)
8      return total_temp
9
10 def get_daily_celcius():
11     # Declare major variables used
12     total_temp = 0
13     week_of_day =['Sun','Mon','Tue','Wed','Thu','Fri','Sat']
14
15     for i in week_of_day:
16         temp = input("Enter " + i + "'s temperature in celcius:")
17         f_temp = convert_temp(temp)
18         print(i + "'s temperature in Fahrenheit:" + str(f_temp))
19         total_temp = compute_total_temp(f_temp, total_temp)
20     return total_temp
21
22 def display_avg(total_temp):
23     avg_temp = round((total_temp/7),2)
24     print("\n" + "Average temperature for the week:" + str(avg_temp))
25     return
26
```

Python Package

- A package is a folder with a "__init__.py" file and other Python module files.
- The "__init__.py" can be empty.
- Its presence permits us to use the folder name as a reference to selectively import needed modules and functions.
- With the use of packages, we can organise a large Python Flask project into a hierarchy of folders, combining 3rd parties' and self-developed (customised) modules and functions.

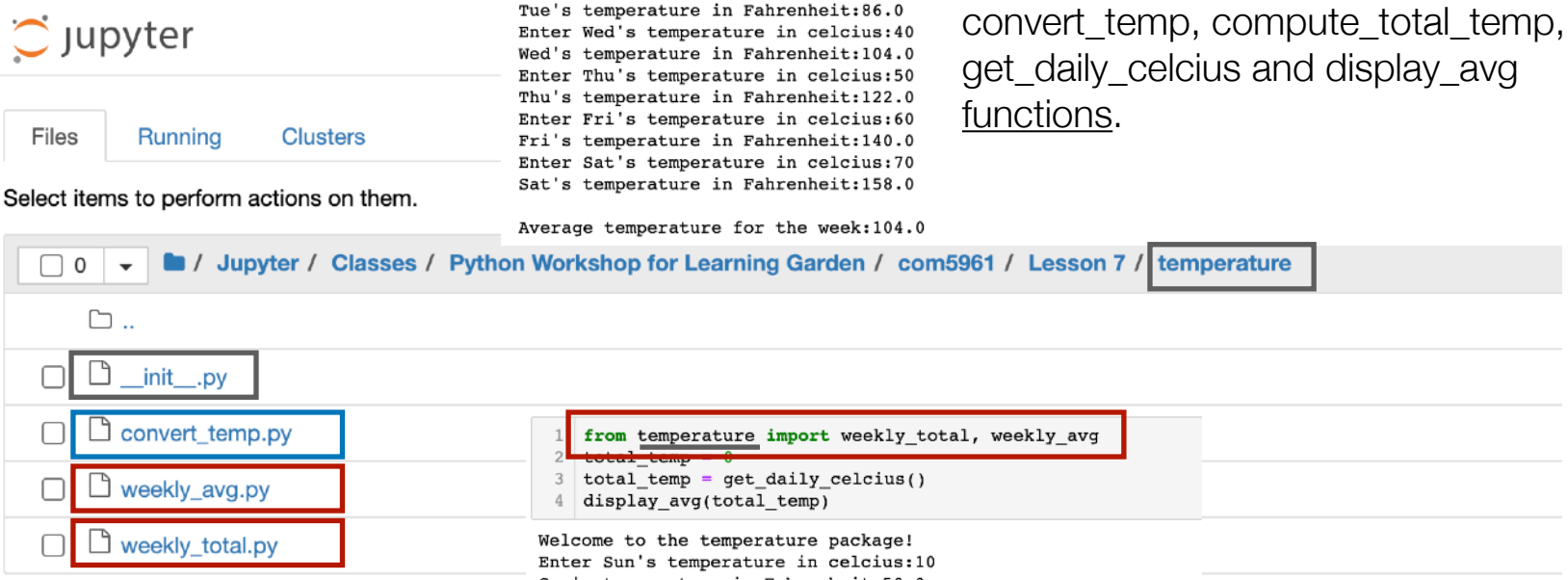

0
/ Jupyter / Classes / Python Workshop for Learning Garden / com5961 / Lesson 7
..
flask_demo
temperature
package.ipynb
convert_temp.py
weekly_hours.csv

```
from temperature.convert_temp import convert_temp, compute_total_temp, get_daily_celcius, display_avg
total_temp = 0
total_temp = get_daily_celcius()
display_avg(total_temp)
```

```
Welcome to the temperature package!
Enter Sun's temperature in celcius:10
Sun's temperature in Fahrenheit:50.0
Enter Mon's temperature in celcius:20
Mon's temperature in Fahrenheit:68.0
Enter Tue's temperature in celcius:30
Tue's temperature in Fahrenheit:86.0
Enter Wed's temperature in celcius:40
Wed's temperature in Fahrenheit:104.0
Enter Thu's temperature in celcius:50
Thu's temperature in Fahrenheit:122.0
Enter Fri's temperature in celcius:60
Fri's temperature in Fahrenheit:140.0
Enter Sat's temperature in celcius:70
Sat's temperature in Fahrenheit:158.0

Average temperature for the week:104.0
```

From the convert_temp module inside the **temperature package** import the convert_temp, compute_total_temp, get_daily_celcius and display_avg functions.




```
from temperature import weekly_total, weekly_avg
total_temp = 0
total_temp = get_daily_celcius()
display_avg(total_temp)
```

```
Welcome to the temperature package!
Enter Sun's temperature in celcius:10
Sun's temperature in Fahrenheit:50.0
Enter Mon's temperature in celcius:20
Mon's temperature in Fahrenheit:68.0
Enter Tue's temperature in celcius:30
Tue's temperature in Fahrenheit:86.0
Enter Wed's temperature in celcius:40
Wed's temperature in Fahrenheit:104.0
Enter Thu's temperature in celcius:50
Thu's temperature in Fahrenheit:122.0
Enter Fri's temperature in celcius:60
Fri's temperature in Fahrenheit:140.0
Enter Sat's temperature in celcius:70
Sat's temperature in Fahrenheit:158.0

Average temperature for the week:104.0
```

From the **temperature package** import the weekly_total and weekly_avg modules.

jupyter convert_temp.py an hour ago

File Edit View Language

```
def convert_temp(temp):
    f_value = 0
    f_value = (float(temp) * 9/5) + 32
    return f_value

def compute_total_temp(temp, total_temp):
    total_temp = total_temp + float(temp)
    return total_temp

def get_daily_celcius():
    # Declare major variables used
    total_temp = 0
    week_of_day =['Sun','Mon','Tue','Wed','Thu','Fri','Sat']
    print("Welcome to the temperature package!")
    for i in week_of_day:
        temp = input("Enter " + i + "'s temperature in celcius:")
        f_temp = convert_temp(temp)
        print(i + "'s temperature in Fahrenheit:" + str(f_temp))
        total_temp = compute_total_temp(f_temp, total_temp)
    return total_temp

def display_avg(total_temp):
    avg_temp = round((total_temp/7),2)
    print("\n" + "Average temperature for the week:" + str(avg_temp))
    return
```

jupyter weekly_total.py an hour ago

File Edit View Language

```
def convert_temp(temp):
    f_value = 0
    f_value = (float(temp) * 9/5) + 32
    return f_value

def compute_total_temp(temp, total_temp):
    total_temp = total_temp + float(temp)
    return total_temp

def get_daily_celcius():
    # Declare major variables used
    total_temp = 0
    week_of_day =['Sun','Mon','Tue','Wed','Thu','Fri','Sat']
    print("Welcome to the temperature package!")
    for i in week_of_day:
        temp = input("Enter " + i + "'s temperature in celcius:")
        f_temp = convert_temp(temp)
        print(i + "'s temperature in Fahrenheit:" + str(f_temp))
        total_temp = compute_total_temp(f_temp, total_temp)
    return total_temp
```

from temperature.convert_temp import convert_temp, compute_total_temp, get_daily_celcius, display_avg

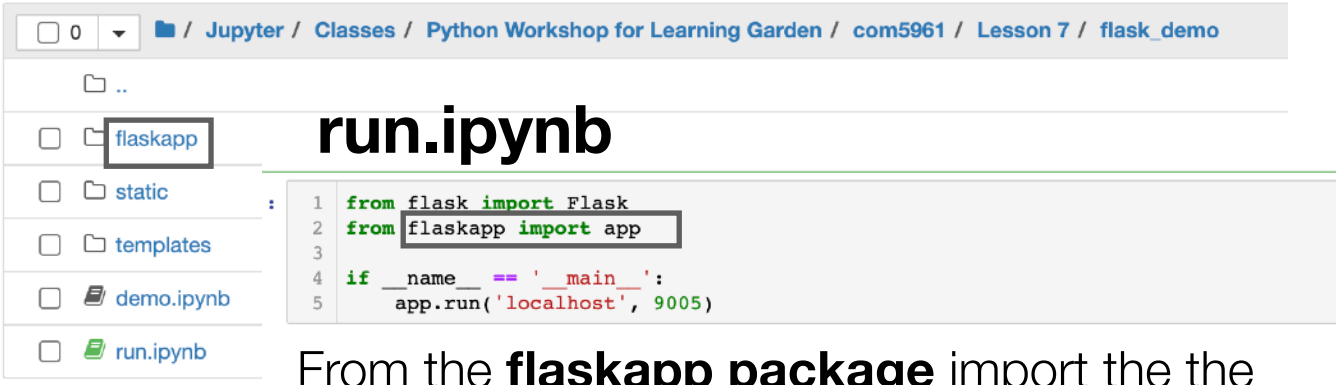
from temperature import weekly_total, weekly_avg

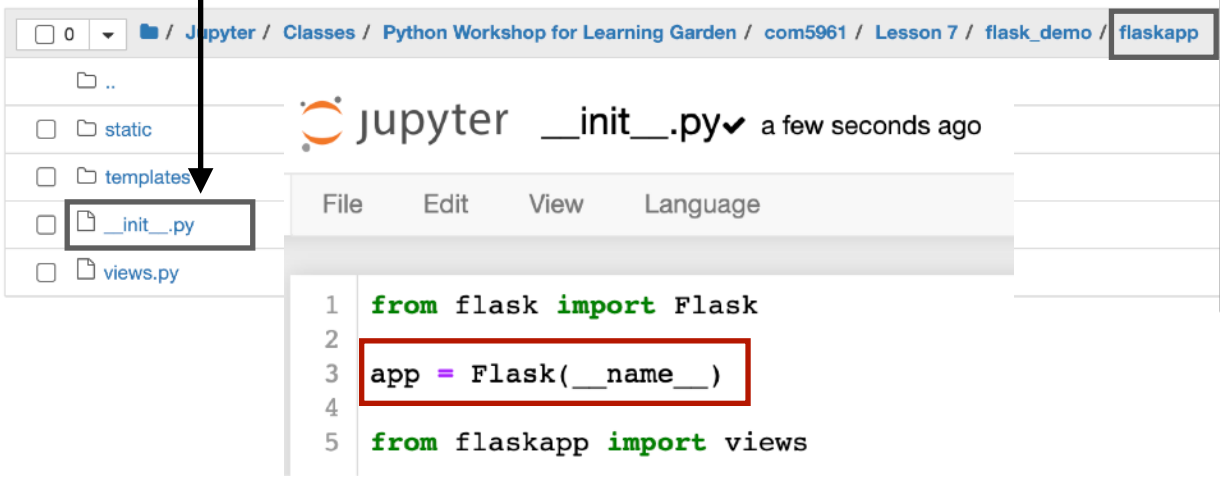
jupyter weekly_avg.py an hour ago

File Edit View Language

```
def display_avg(total_temp):
    avg_temp = round((total_temp/7),2)
    print("\n" + "Average temperature for the week:" + str(avg_temp))
    return
```

Building a Flask package.





Flask类只有一个必须指定的参数，即程序主模块或者包的名字，__name__是系统变量，该变量指的是本py文件的文件名"""
Source: https://www.cnblogs.com/chaojiyingxiong/p/9549987.html

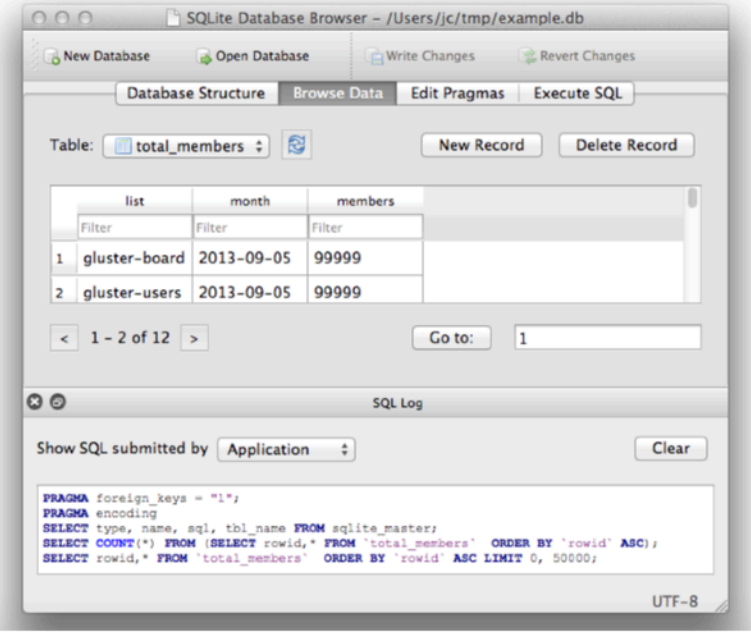
The Database Component

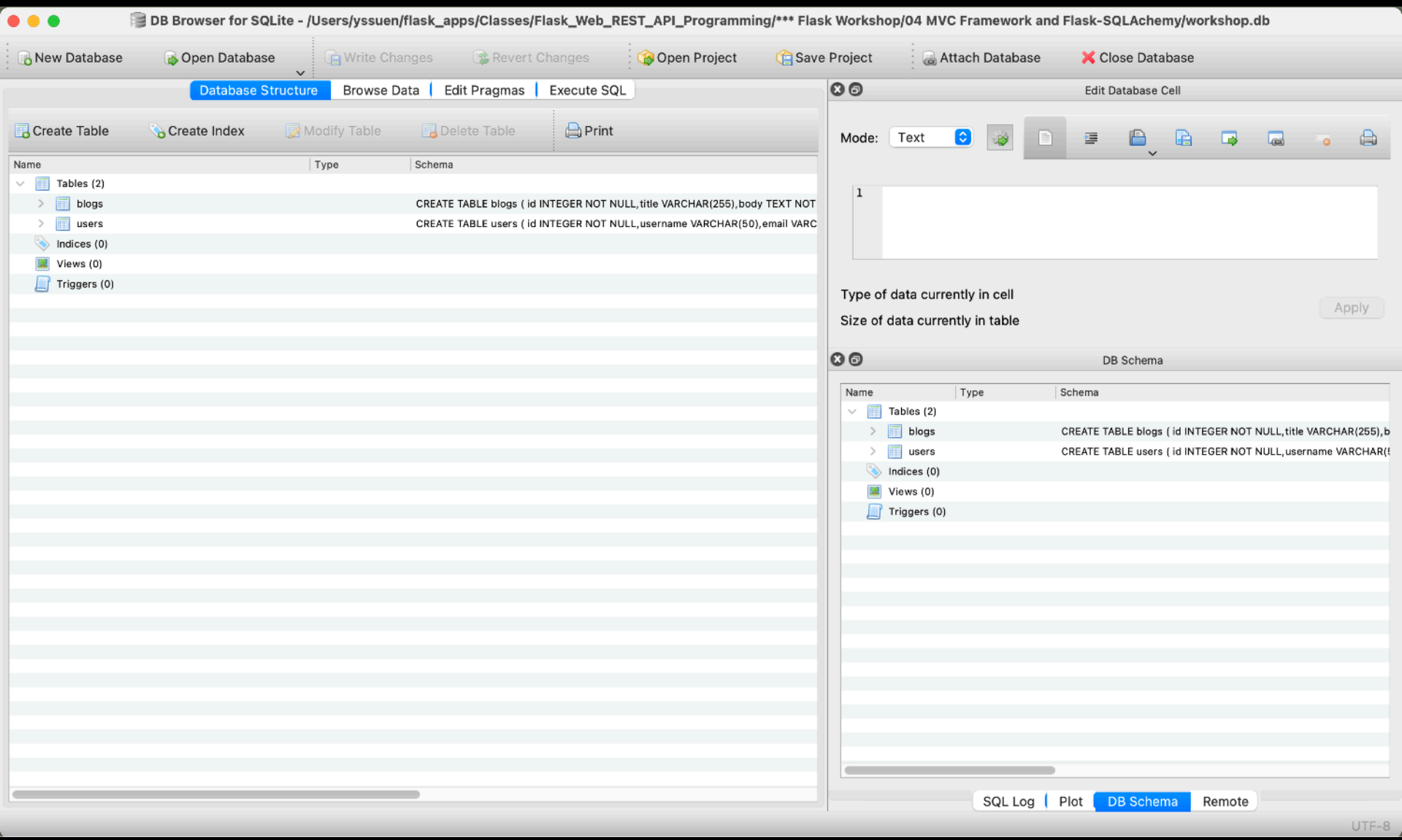
SQLite for Quick Prototyping

DB Browser for SQLite

The Official home of the DB Browser for SQLite

Screenshot




DB Browser for SQLite - /Users/yssuen/flask_apps/Classes/Flask_Web_REST_API_Programming/*** Flask Workshop/04 MVC Framework and Flask-SQLAchemy/workshop.db
New Database
Open Database
Write Changes
Revert Changes
Open Project
Save Project
Attach Database
Close Database
Database Structure
Browse Data
Edit Pragmas
Execute SQL
Create Table
Create Index
Modify Table
Delete Table
Print
Name
Type
Schema
Tables (2)
blogs
CREATE TABLE blogs (id INTEGER NOT NULL,title VARCHAR(255),body TEXT NOT
users
CREATE TABLE users (id INTEGER NOT NULL,username VARCHAR(50),email VARC
Indices (0)
Views (0)
Triggers (0)
Edit Database Cell
Mode:
Text
1
Type of data currently in cell
Size of data currently in table
Apply
DB Schema
Name
Type
Schema
Tables (2)
blogs
CREATE TABLE blogs (id INTEGER NOT NULL,title VARCHAR(255),b
users
CREATE TABLE users (id INTEGER NOT NULL,username VARCHAR(
Indices (0)
Views (0)
Triggers (0)
SQL Log
Plot
DB Schema
Remote
UTF-8

Introduction to SQLAlchemy

Why SQLAlchemy?

- Most popular database package for Flask with extensive supports and examples.
- Support many popular SQL databases such as SQLite, MySQL, Postgresql, MS-SQL and Oracle.
- Support both Object Relational Mapping (ORM) and Structured Query Language (SQL) based data manipulation.
- Support paging of data records, making it ideal for e-commerce catalog production.
- Support one-to-many and many-to-many relationships (https://flask-sqlalchemy.palletsprojects.com/en/2.x/models/)

Data Definition Language (DDL)

Set Up Database (Schema)

CREATE DATABASE database name

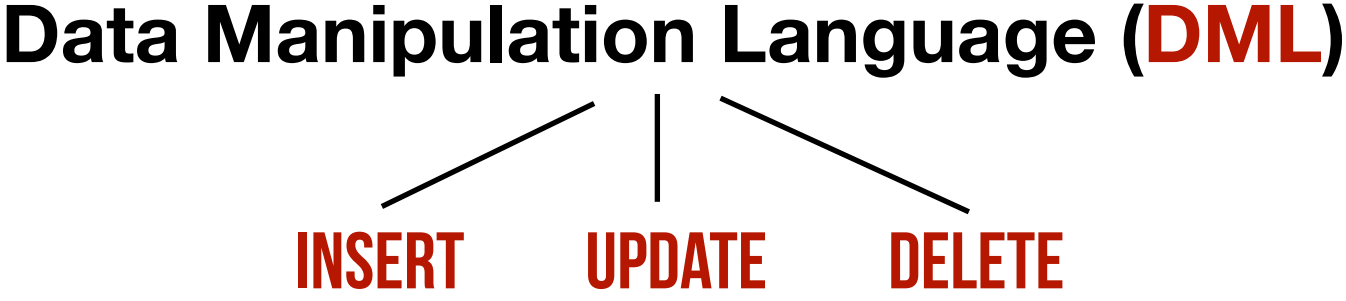
DROP DATABASE database name

SHOW DATABASES

Data Definition Language (DDL)

Set Up Table (Schema)

CREATE TABLE IF NOT EXISTS table name
SHOW TABLES

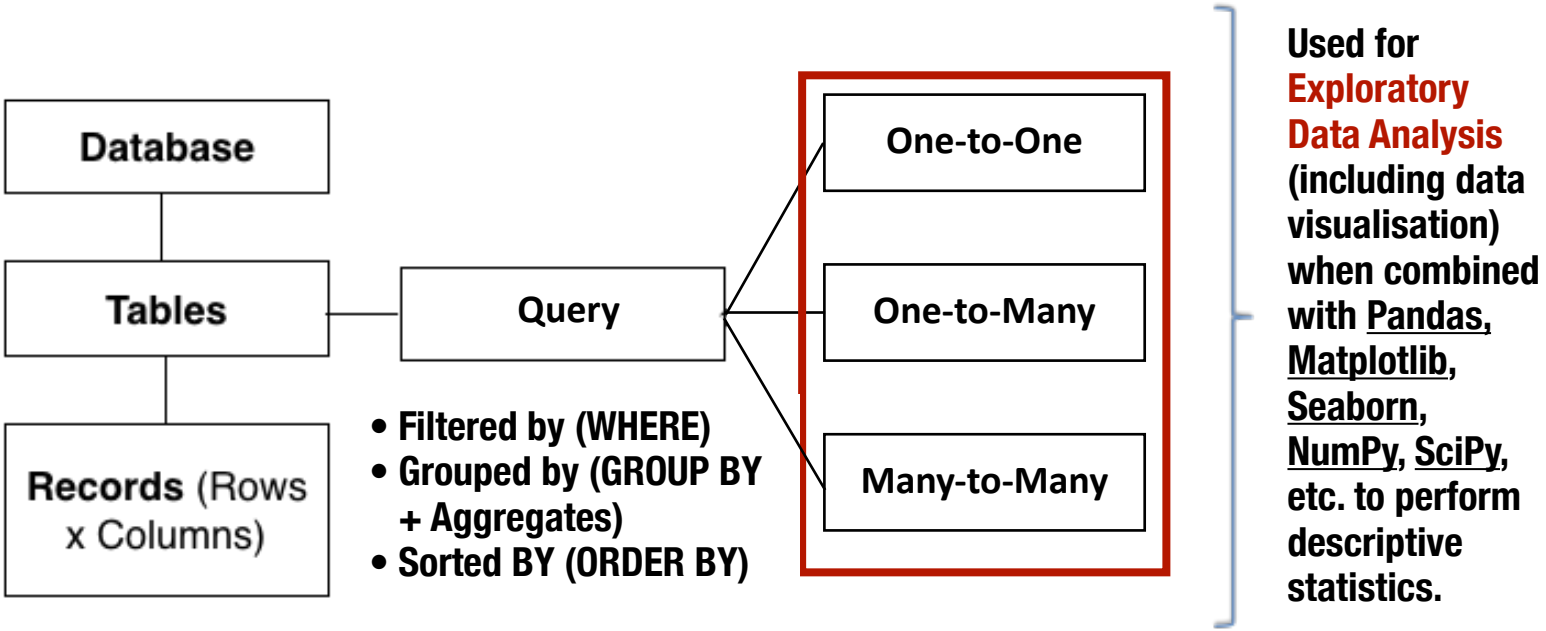

Data Manipulation Language (DML)
INSERT
UPDATE
DELETE

Data Query Language (DQL)***

*** DQL used most frequently in analytics.

Retrieve Record

SELECT FROM table name **WHERE** condition
GROUP BY field name
ORDER BY field name **ASC | DESC**


Database
Tables
Records (Rows x Columns)
Query
• Filtered by (WHERE)
• Grouped by (GROUP BY + Aggregates)
• Sorted BY (ORDER BY)
One-to-One
One-to-Many
Many-to-Many
Used for Exploratory Data Analysis (including data visualisation) when combined with Pandas, Matplotlib, Seaborn, NumPy, SciPy, etc. to perform descriptive statistics.

CRUD Operations in SQL (DML + DQL)

See Examples in Jupyter Notebook.

The View Component

B Home Documentation Examples Themes Expo Blog v4.3 Download

Bootstrap

Build responsive, mobile-first projects on the web with the world's most popular front-end component library.

Bootstrap is an open source toolkit for developing with HTML, CSS, and JS. Quickly prototype your ideas or build your entire app with our Sass variables and mixins, responsive grid system, extensive prebuilt components, and powerful plugins built on jQuery.

Get started Download

Currently v4.3.1

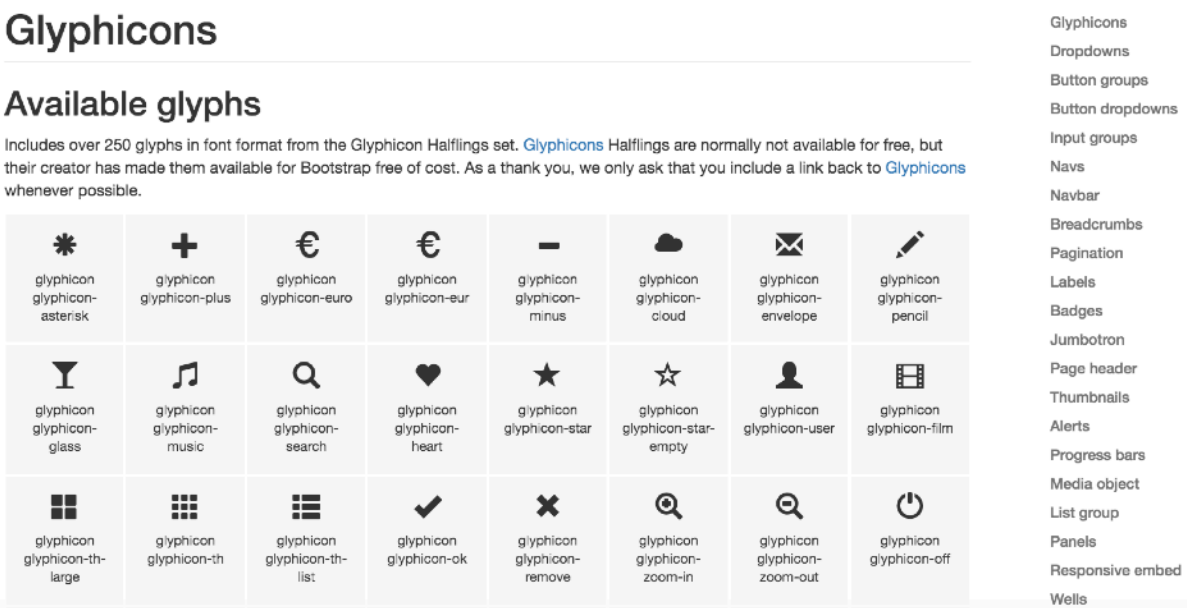
2

https://getbootstrap.com/

Bootstrap Components for Responsive Web Page Layout and Composition.

Components

Over a dozen reusable components built to provide iconography, dropdowns, input groups, navigation, alerts, and much more.



Glyphicons

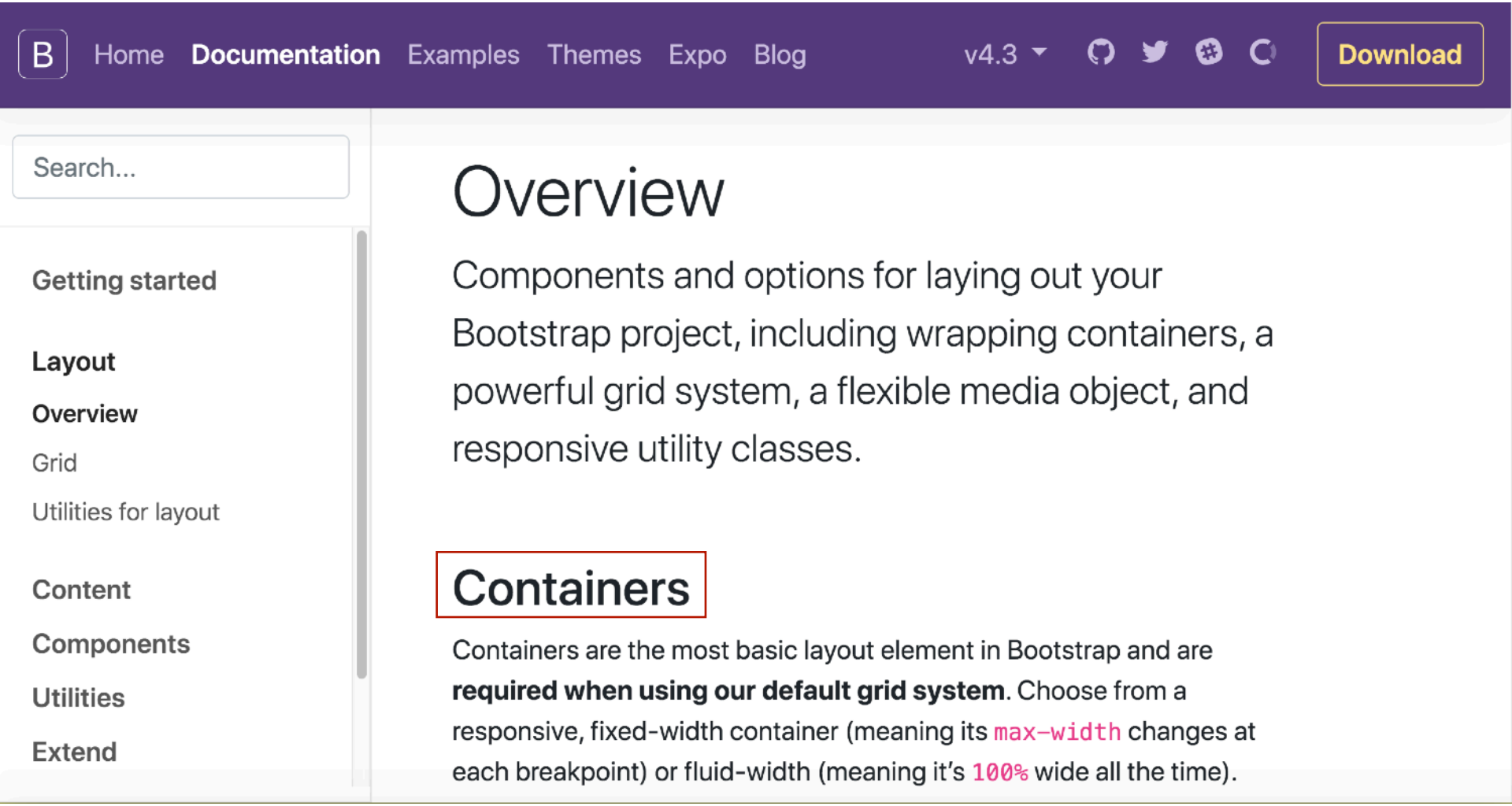
Available glyphs

Includes over 250 glyphs in font format from the Glyphicon Halflings set. Glyphicons Halflings are normally not available for free, but their creator has made them available for Bootstrap free of cost. As a thank you, we only ask that you include a link back to Glyphicons whenever possible.

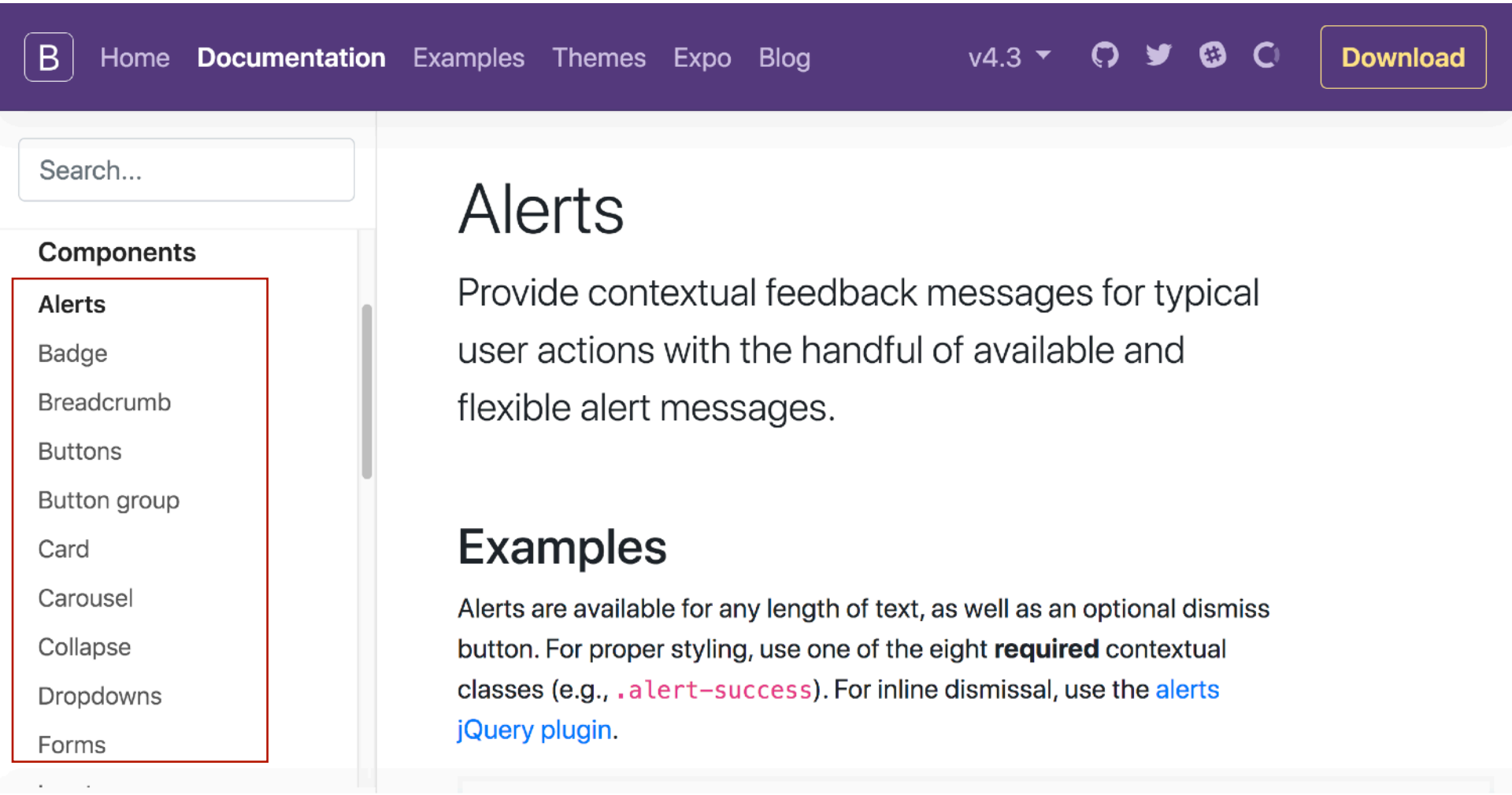
glyphicon glyphicon-asterisk
glyphicon glyphicon-plus
glyphicon glyphicon-euro
glyphicon glyphicon-eur
glyphicon glyphicon-minus
glyphicon glyphicon-cloud
glyphicon glyphicon-envelope
glyphicon glyphicon-pencil
glyphicon glyphicon-glass
glyphicon glyphicon-music
glyphicon glyphicon-search
glyphicon glyphicon-heart
glyphicon glyphicon-star
glyphicon glyphicon-star-empty
glyphicon glyphicon-user
glyphicon glyphicon-film
glyphicon glyphicon-th-large
glyphicon glyphicon-th
glyphicon glyphicon-th-list
glyphicon glyphicon-ok
glyphicon glyphicon-remove
glyphicon glyphicon-zoom-in
glyphicon glyphicon-zoom-out
glyphicon glyphicon-off

Glyphicons
Dropdowns
Button groups
Button dropdowns
Input groups
Navs
Navbar
Breadcrumbs
Pagination
Labels
Badges
Jumbotron
Page header
Thumbnails
Alerts
Progress bars
Media object
List group
Panels
Responsive embed
Wells

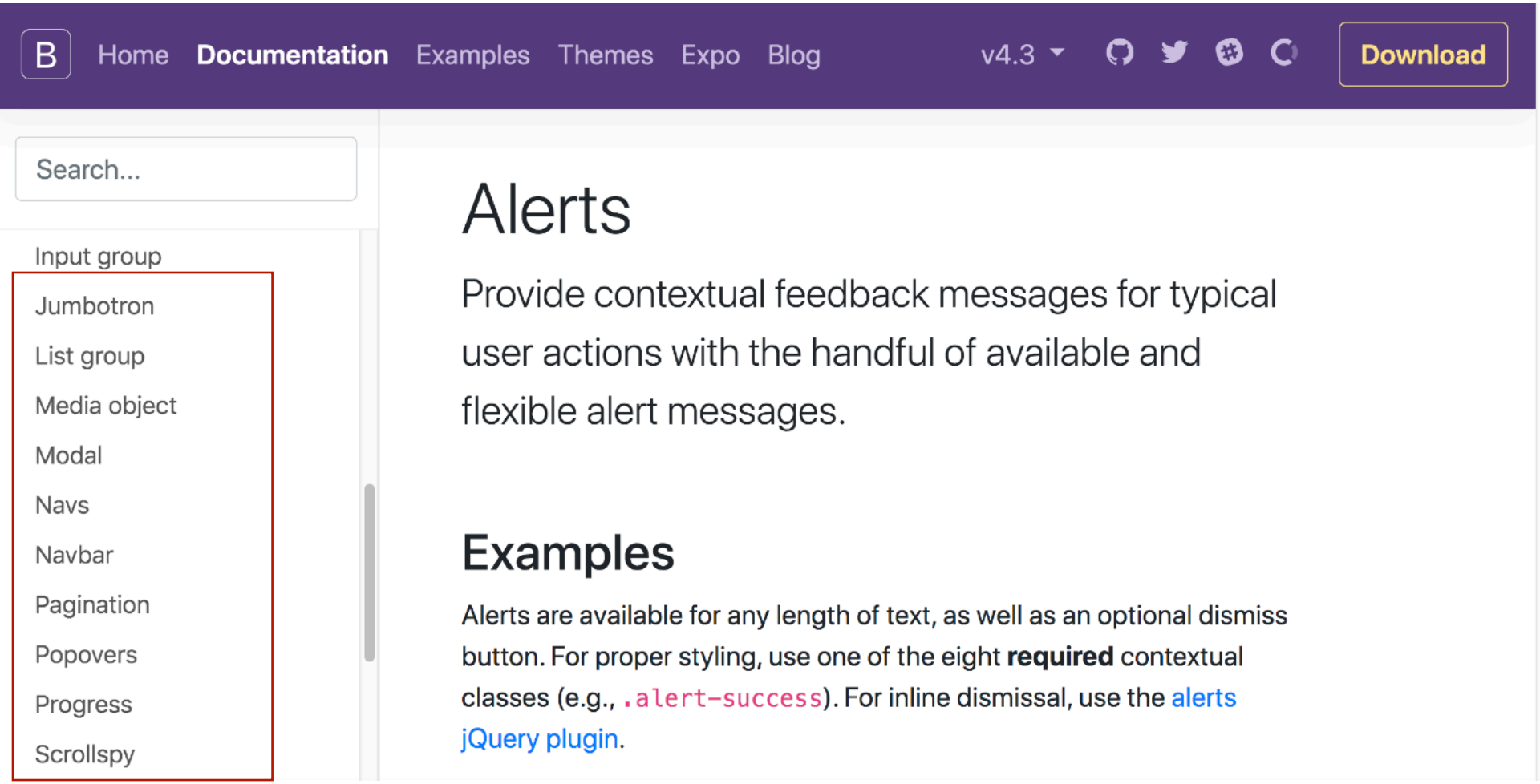
https://getbootstrap.com/docs/3.3/components/#nav



https://getbootstrap.com/docs/4.3/layout/overview/



https://getbootstrap.com/docs/4.3/components/alerts/


B
Home
Documentation
Examples
Themes
Expo
Blog
v4.3
Download
Search...
Input group
Jumbotron
List group
Media object
Modal
Navs
Navbar
Pagination
Popovers
Progress
Scrollspy
Alerts
Provide contextual feedback messages for typical user actions with the handful of available and flexible alert messages.
Examples
Alerts are available for any length of text, as well as an optional dismiss button. For proper styling, use one of the eight required contextual classes (e.g., .alert-success). For inline dismissal, use the alerts jQuery plugin.

Search...

Popovers
Progress
Scrollspy
Spinners
Toasts
Tooltips

Utilities
Extend
Migration
About

Bootstrap "spinners" can be used to show the loading state in your projects. They're built only with HTML and CSS, meaning you don't need any JavaScript to create them. You will, however, need some custom JavaScript to toggle their visibility. Their appearance, alignment, and sizing can be easily customized with our amazing utility classes.

For accessibility purposes, each loader here includes `role="status"` and a nested `<span class="sr-only">Loading...</span>`.

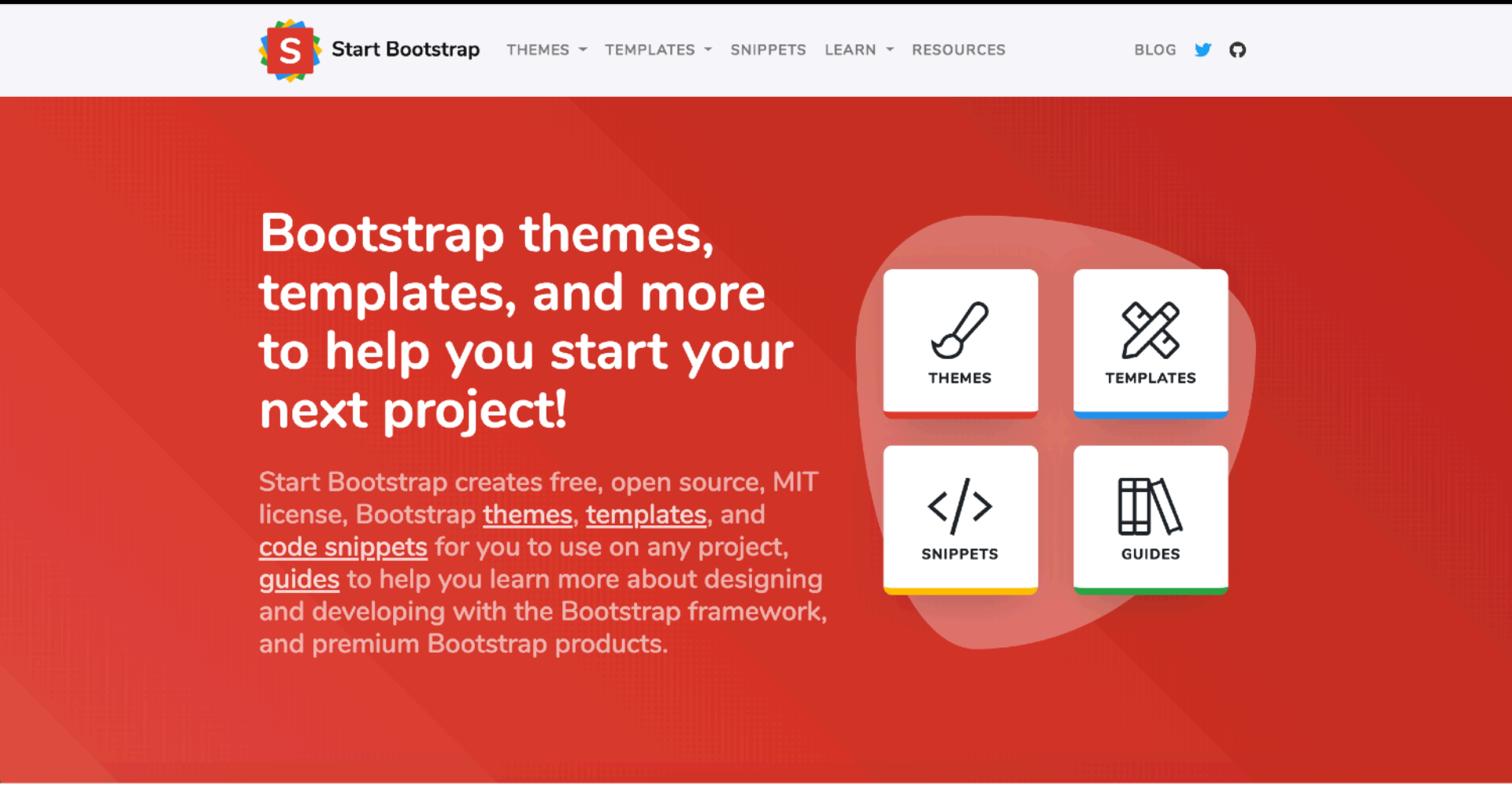
Border spinner

Use the border spinners for a lightweight loading indicator.

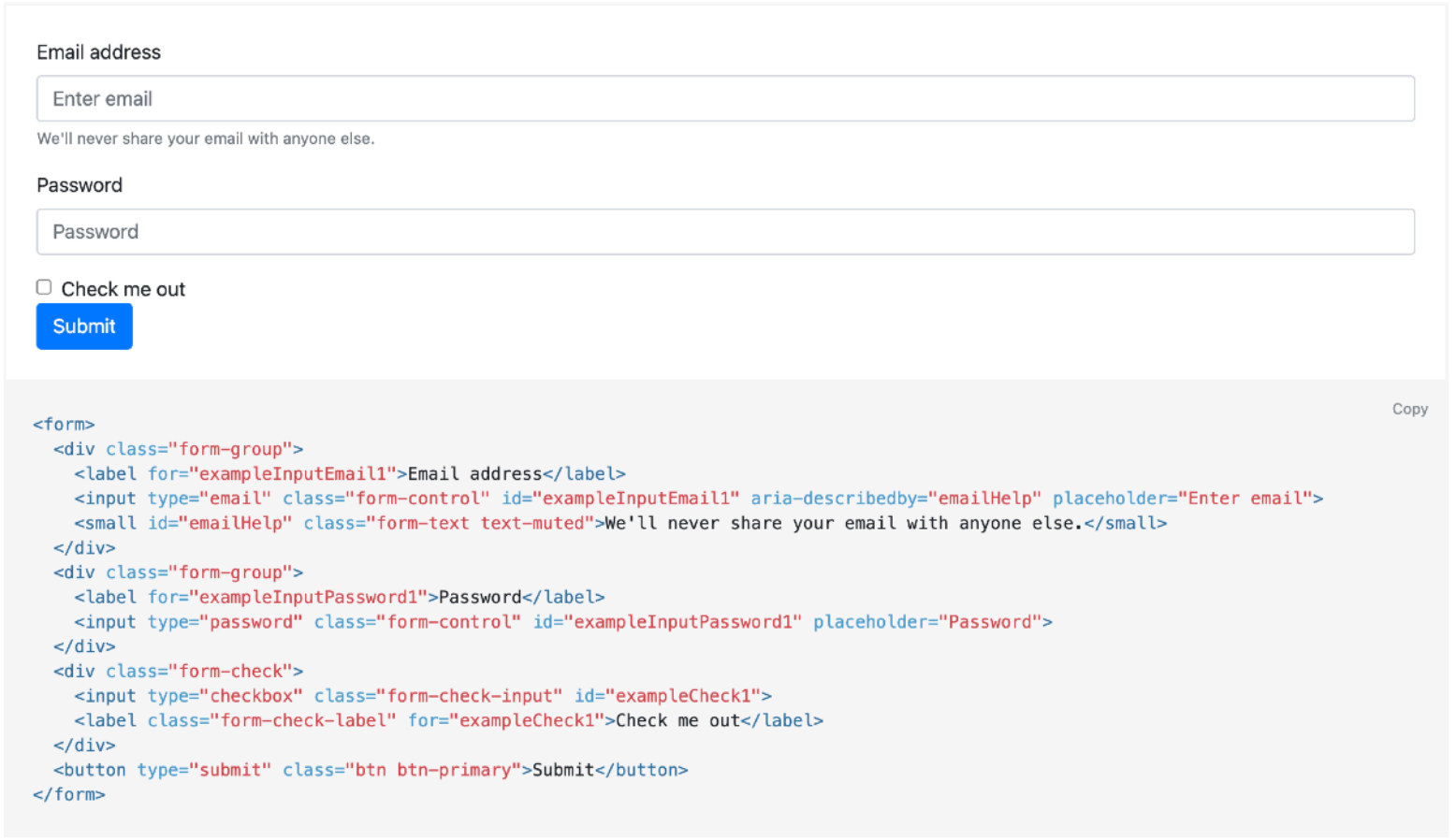
Popular Components

a. Container
b. Navbar
c. Glphicons
d. Grid/Row/Column
e. Thumbnail
f. Button
g. Form
h. Panel/Wells/Jumbotron
i. Carousel/Collapse
j. Tabs
k. Table
l. Pagination
m. Badges/List Group/Cards
n. Modal/Tooltip/Popover

https://www.w3schools.com/bootstrap4/default.asp



https://startbootstrap.com/



Source: https://getbootstrap.com/docs/4.0/components/forms/

jupyter sign_up.html a minute ago

File Edit View Language

```
1  {% extends "base.html" %}
2  {% block header %}
3  <!-- Page Header-->
4          <header class="masthead">
5              <div class="overlay"></div>
6              <div class="container">
7                  <div class="row" style="max-height: 330px; margin-top: -5%;">
8                      <div class="col-lg-8 col-md-10 mx-auto">
9                          <div class="site-heading">
10                             <h3>Dashboard</h3>
11                         </div>
12                     </div>
13                 </div>
14             </div>
15         </header>
16 {% endblock %}
17 {% block content %}
18         <!-- Main Content-->
19         <div class="container">
20
21         </div>
22         <hr />
23 {% endblock %}
```

```
<form>
  <div class="form-group">
    <label for="exampleInputEmail1">Email address</label>
    <input type="email" class="form-control" id="exampleInputEmail1" aria-describedby="emailHelp" placeholder="Enter email">
    <small id="emailHelp" class="form-text text-muted">We'll never share your email with anyone else.</small>
  </div>
  <div class="form-group">
    <label for="exampleInputPassword1">Password</label>
    <input type="password" class="form-control" id="exampleInputPassword1" placeholder="Password">
  </div>
  <div class="form-check">
    <input type="checkbox" class="form-check-input" id="exampleCheck1">
    <label class="form-check-label" for="exampleCheck1">Check me out</label>
  </div>
  <button type="submit" class="btn btn-primary">Submit</button>
</form>
```

Insert the Bootstrap form codes into the **div** element with **container** class.

Buttons

Use Bootstrap's custom button styles for actions in forms, dialogs, and more with support for multiple sizes, states, and more.

Examples

https://getbootstrap.com/docs/4.0/components/buttons/

Bootstrap includes several predefined button styles, each serving its own semantic purpose, with a few extras thrown in for more control.

Primary Secondary Success Danger Warning Info Light Dark Link

```
<button type="button" class="btn btn-primary">Primary</button>
<button type="button" class="btn btn-secondary">Secondary</button>
<button type="button" class="btn btn-success">Success</button>
<button type="button" class="btn btn-danger">Danger</button>
<button type="button" class="btn btn-warning">Warning</button>
<button type="button" class="btn btn-info">Info</button>
<button type="button" class="btn btn-light">Light</button>
<button type="button" class="btn btn-dark">Dark</button>

<button type="button" class="btn btn-link">Link</button>
```

Hands-on Workshop

CRUD Operations without Authentication

Task Master

Task	Added	Actions
Publish next class online material	2021-10-18	Delete Update
Answer discussion forum	2021-10-18	Delete Update
Respond to email	2021-10-18	Delete Update
Prepare next class powerpoint	2021-10-18	Delete Update
Correct assignments and give feedbacks	2021-10-18	Delete Update

Add Task

YouTube Tutorial: https://www.youtube.com/watch?v=Z1RJmh_OqeA&t=1004s

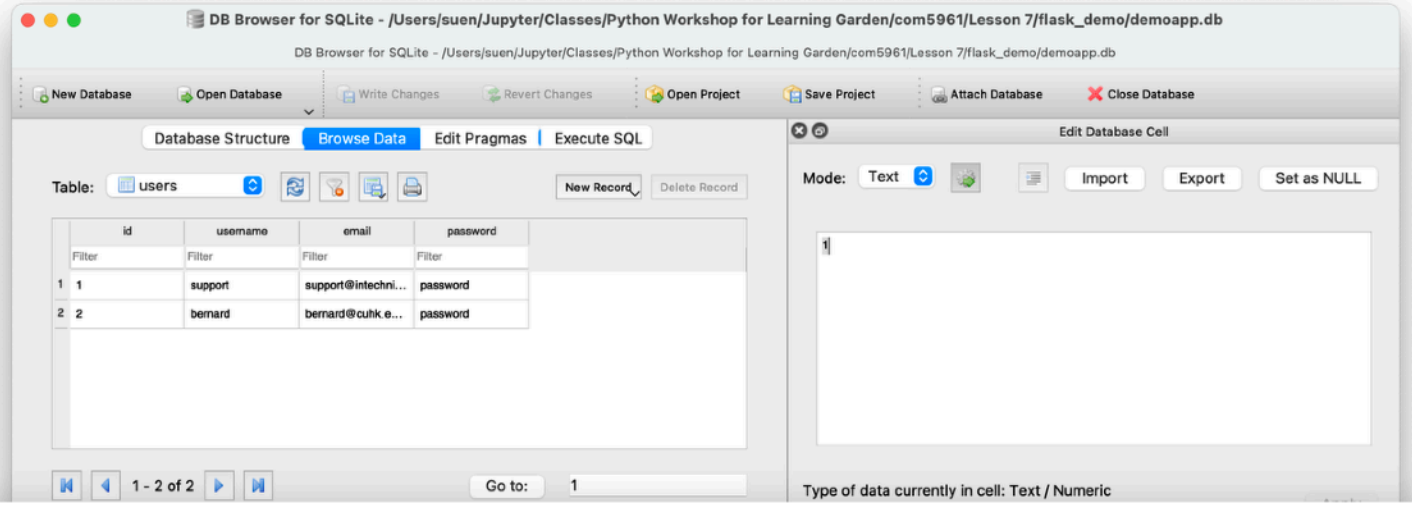
Github source codes: https://github.com/jakerieger/FlaskIntroduction

Authentication without Encryption

Start Bootstrap HOME ABOUT BLOGS SAMPLE POST CONTACT DASHBOARD USERS SIGN UP

Users

ID	User Name	Email
1	support	support@intechnigence.com
2	bernard	bernard@cuhk.edu.hk



Encryption with BCrypt

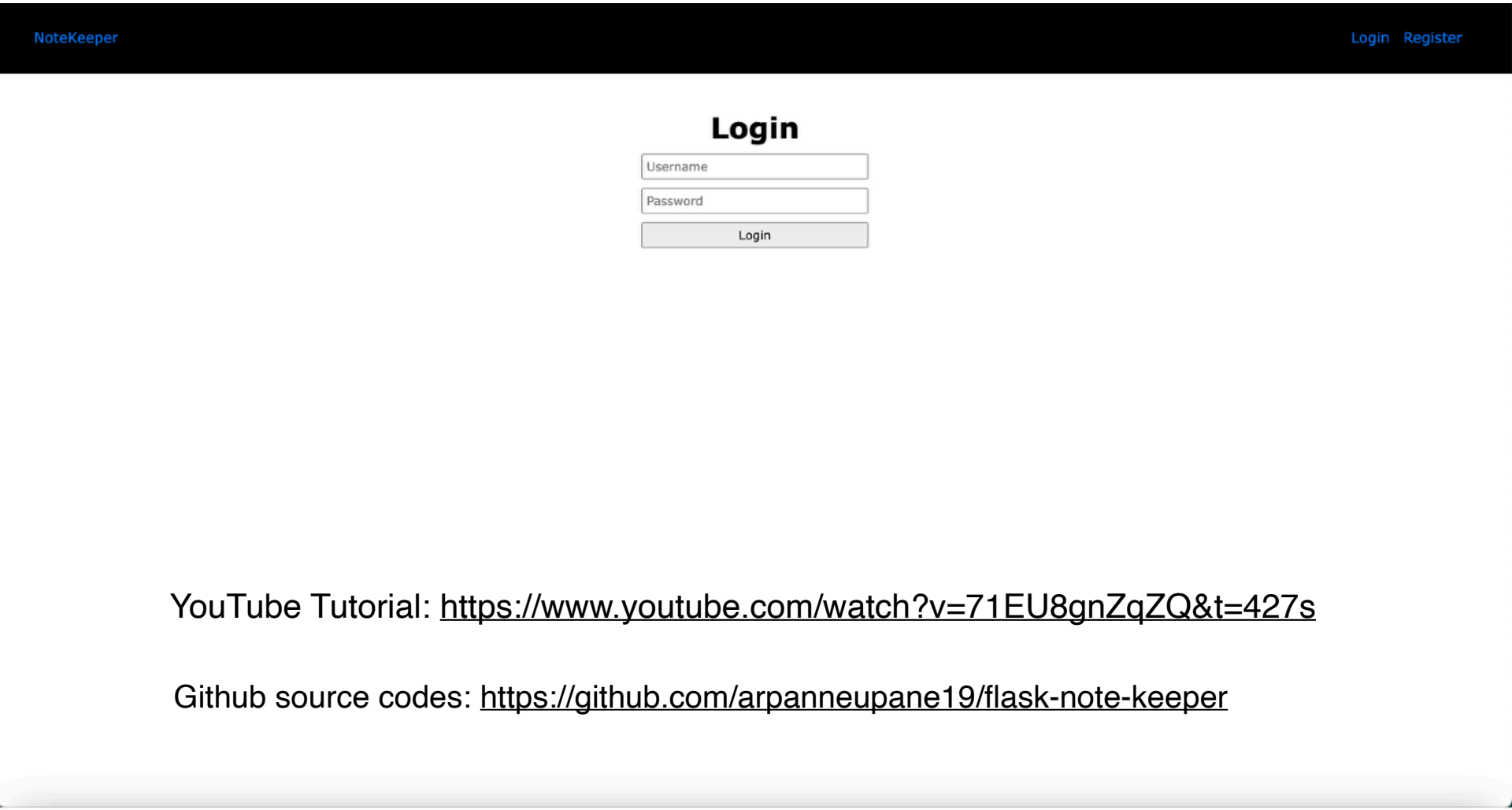
In [19]:

```python
import bcrypt
from getpass import getpass
password2 = getpass("Enter password:")
print('Password entered:',password2)
password2 = password2.encode('UTF-8')
print("Encode as UTF-8 :",password2)
# Hash a password for the first time, with a randomly-generated salt
hashed = bcrypt.hashpw(password2, bcrypt.gensalt())
print("Encrypted password:",hashed)
print("Decode UTF-8:",hashed.decode('UTF-8'))
# Check that an unhashed password matches one that has previously been
# hashed
password = getpass("Confirm password:")
print("Confirmed password entered:",password)
password = password.encode('UTF-8')
print("Confirmed password as UTF-8:",password)
if bcrypt.checkpw(password, hashed): #Compare passwords
    print("It Matches!")
else:
    print("It Does not Match :(")

```

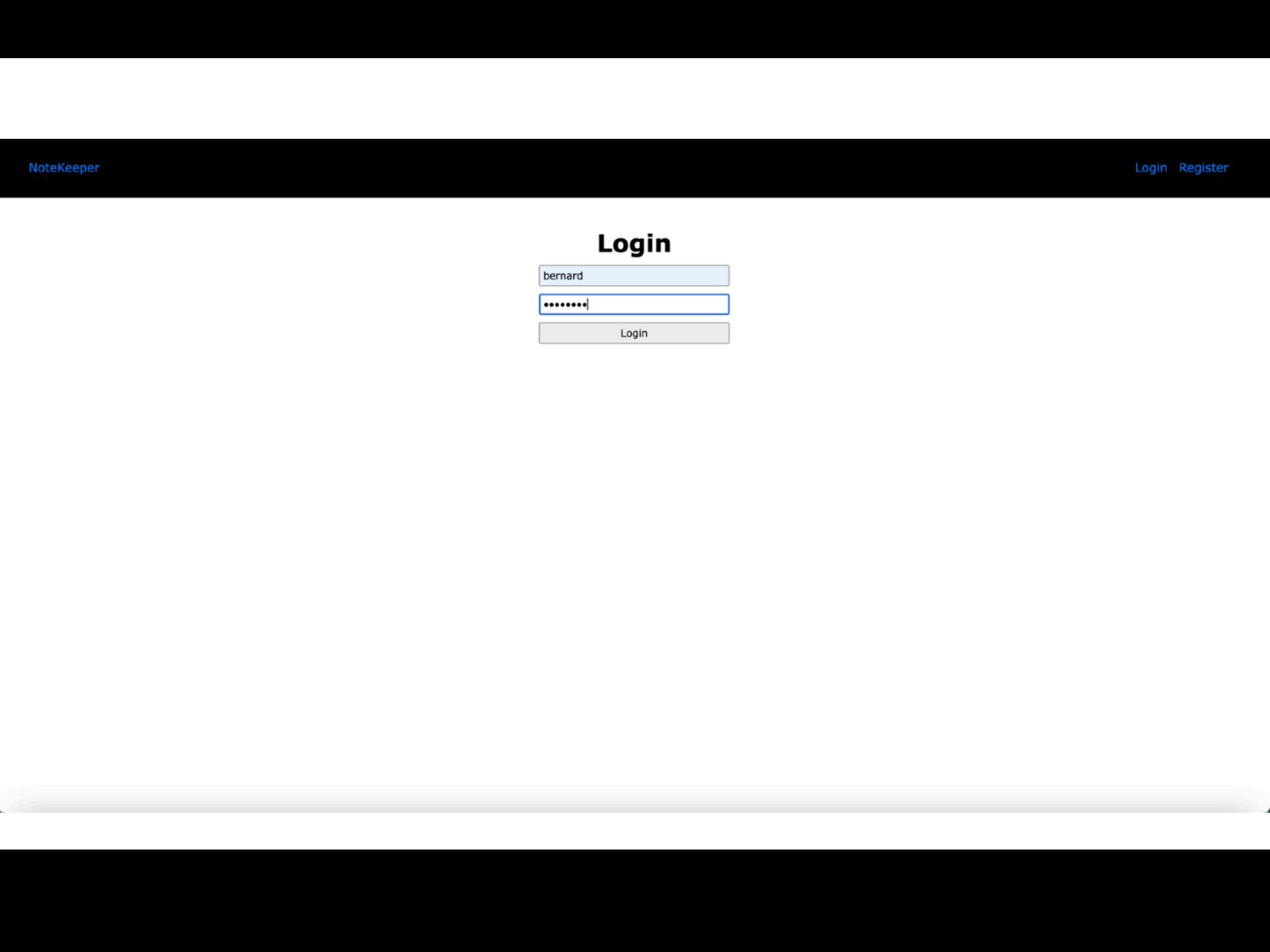
```
Enter password:••••••••
Password entered: peace
Encode as UTF-8 : b'peace'
Encrypted password: b'$2b$12$NgPrB/S9fRXn3XFyQViAyuPWb7LvllKP8pIiTS5un1hKDd4Raz0ku'
Decode UTF-8: $2b$12$NgPrB/S9fRXn3XFyQViAyuPWb7LvllKP8pIiTS5un1hKDd4Raz0ku
Confirm password:••••••••
Confirmed password entered: hello
Confirmed password as UTF-8: b'hello'
It Does not Match :(
```

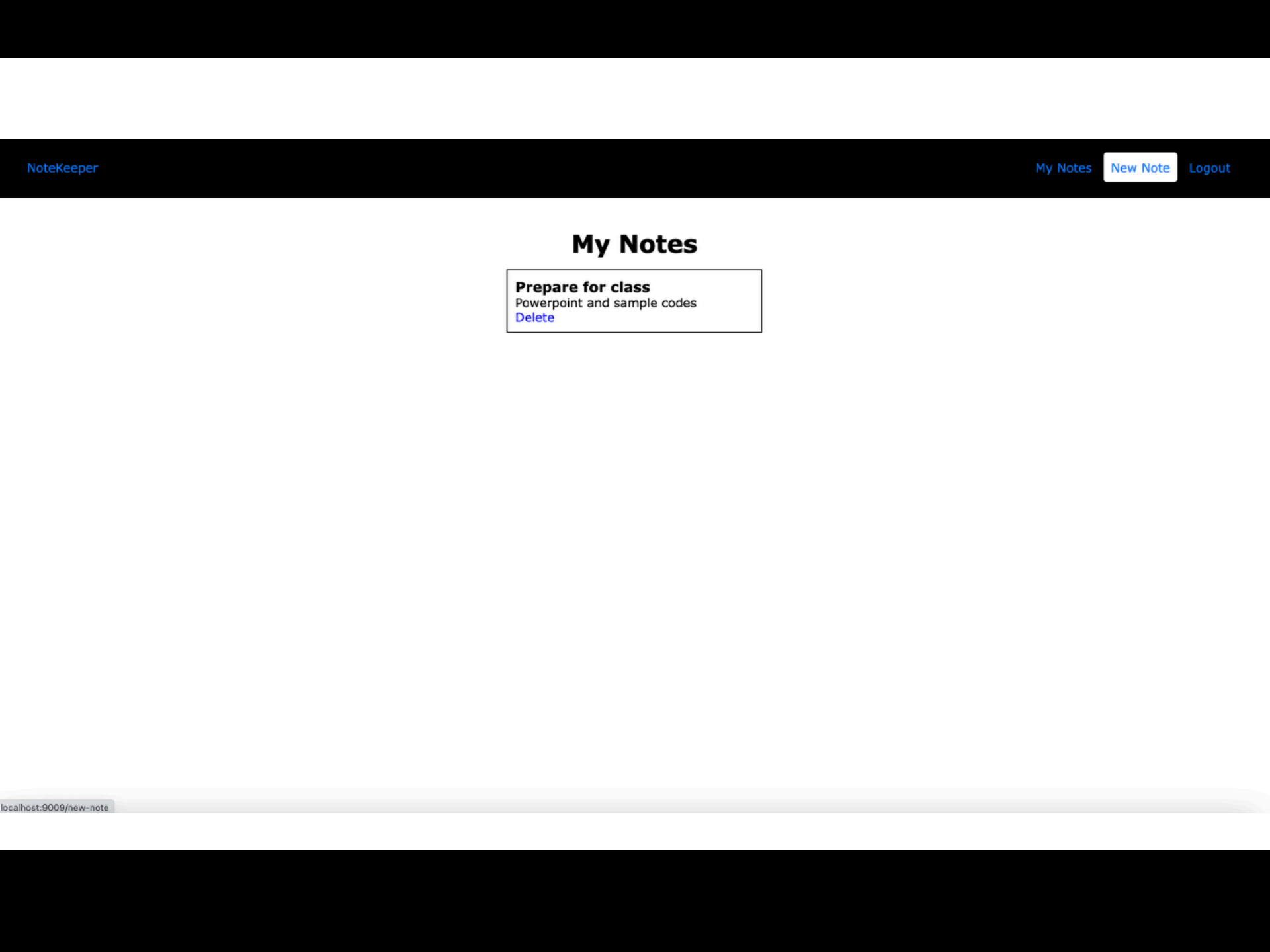
CRUD Operations with Authentication

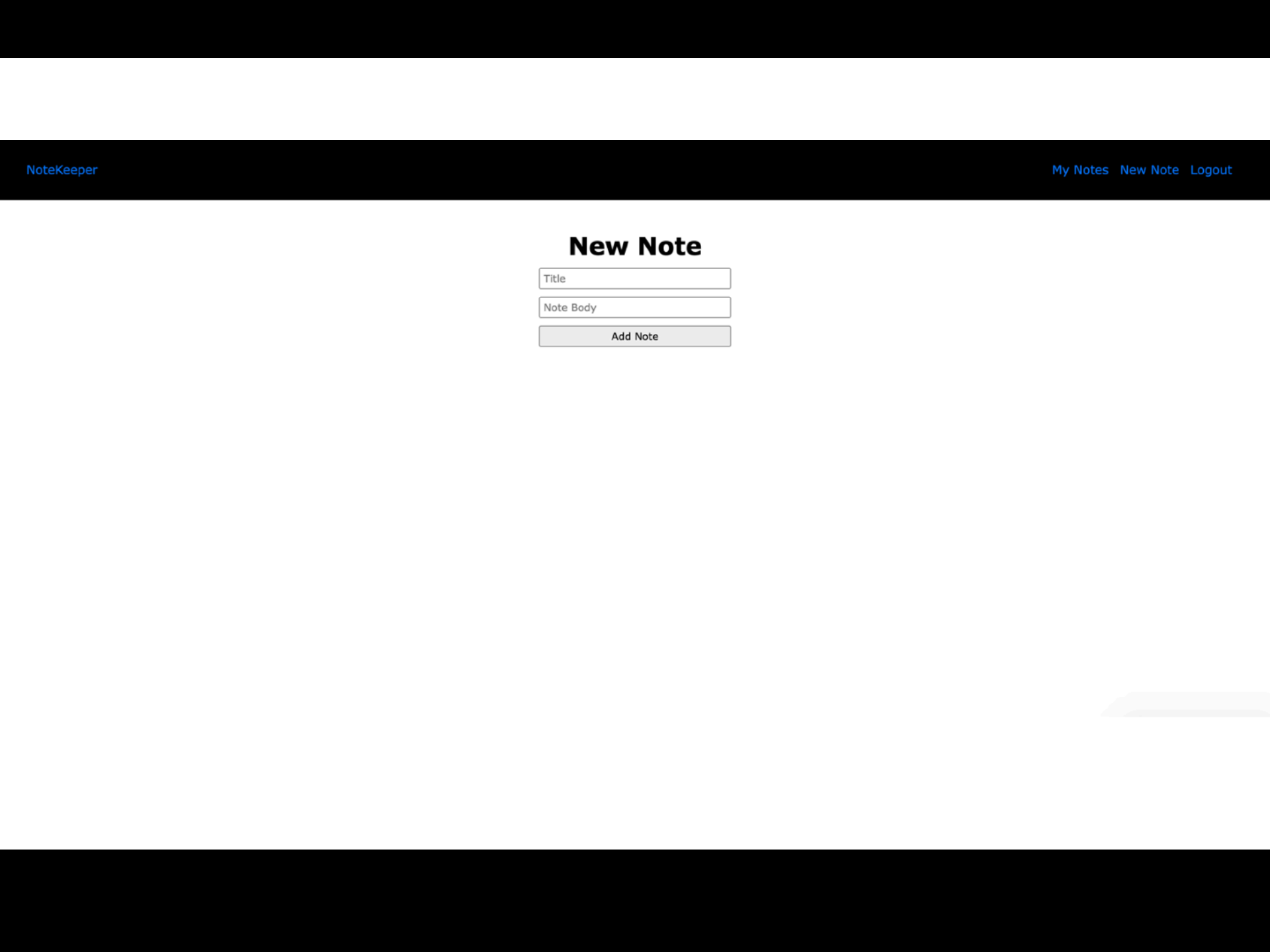


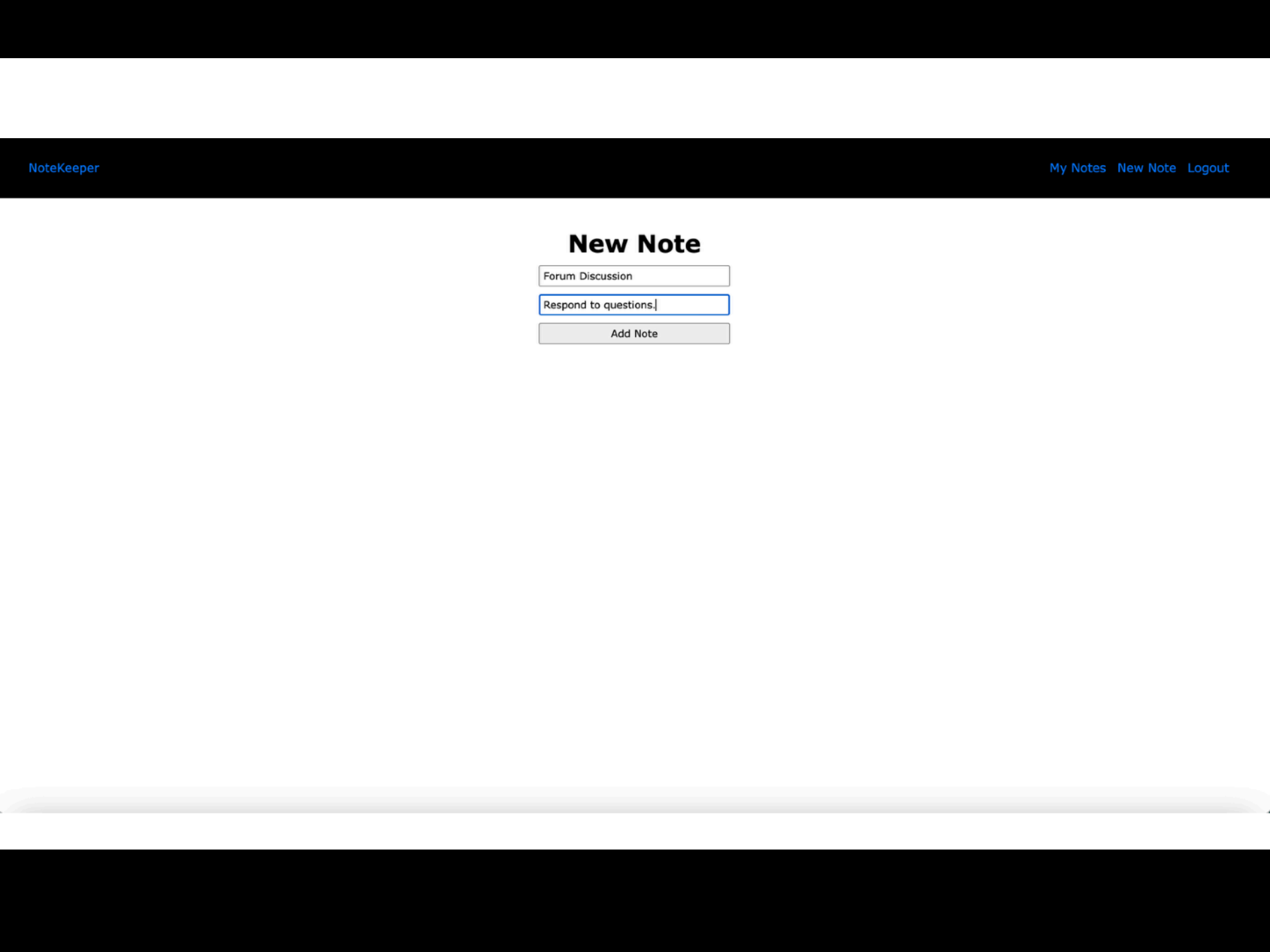
YouTube Tutorial: https://www.youtube.com/watch?v=71EU8gnZqZQ&t=427s

Github source codes: https://github.com/arpanneupane19/flask-note-keeper


NoteKeeper
Login
Register
Login
bernard
Login


NoteKeeper
My Notes
New Note
Logout
My Notes
Prepare for class
Powerpoint and sample codes
Delete
localhost:9009/new-note


NoteKeeper
My Notes
New Note
Logout
New Note
Title
Note Body
Add Note


NoteKeeper
My Notes
New Note
Logout
New Note
Forum Discussion
Respond to questions.
Add Note

NoteKeeper

My Notes New Note Logout

My Notes

Prepare for class
Powerpoint and sample codes
Delete

Forum Discussion
Respond to questions.
Delete

Man must explore, and this is exploration at its greatest

Problems look mighty small from 150 miles up

Posted by Start Bootstrap on September 24, 2021

I believe every human has a finite number of heartbeats. I don't intend to waste any of mine.

Posted by Start Bootstrap on September 18, 2021

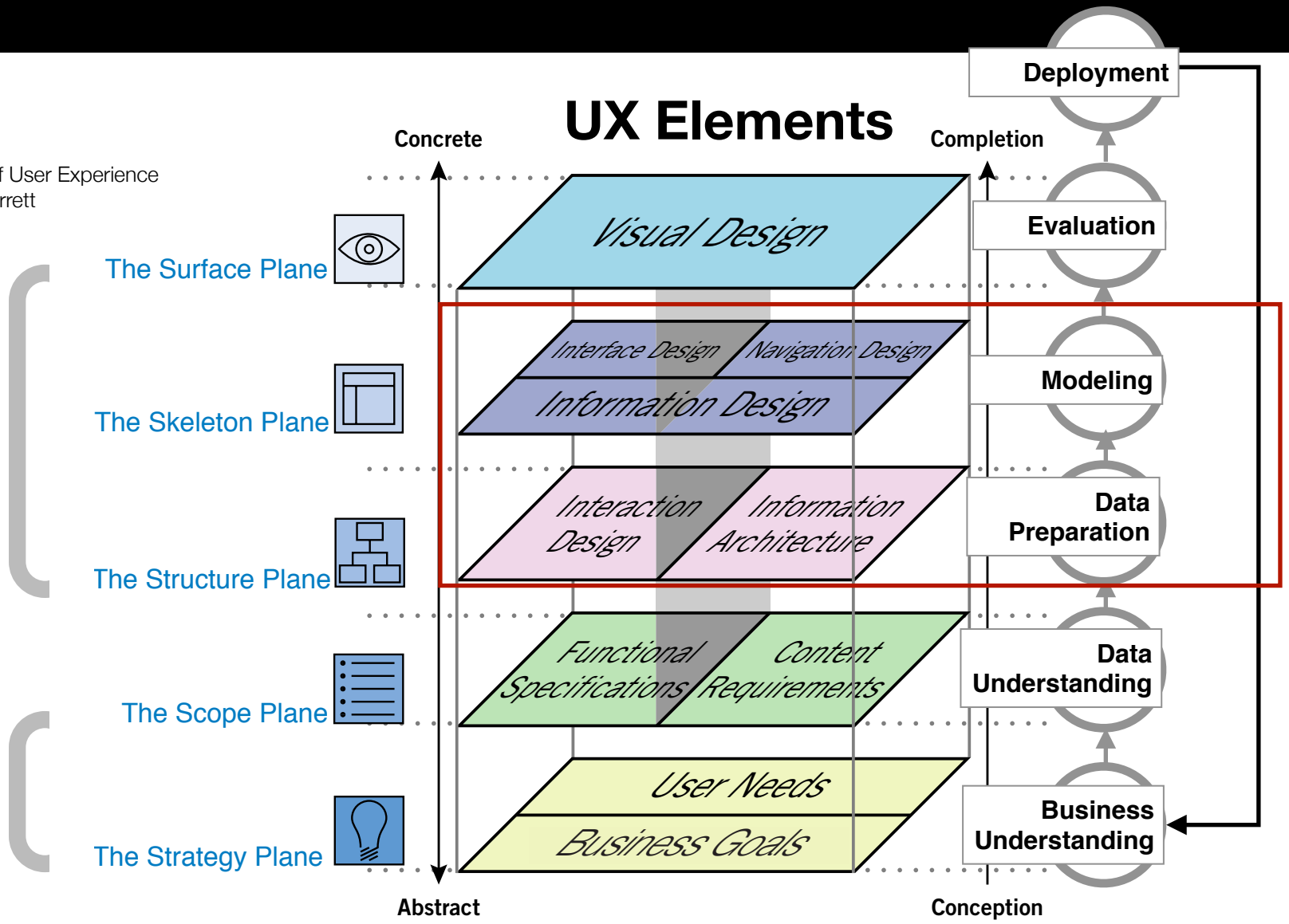
Dashboard

ID	Title	Author	Date		
1	MAKING REMOTE WORKS COLLABORATIVE AND ACCOUNTABLE	info	2021-05-16 14:00:45.036667	EDIT	DELETE
2	LESSON FROM THE TIKTOK DEAL: NAVIGATING THE PARALLEL INTERNET	info	2021-05-16 14:01:18.393540	EDIT	DELETE
3	COVID-19: THE GREAT ACCELERATOR OF WORK AND LEARNING	info	2021-05-16 14:01:54.522165	EDIT	DELETE
4	THINK MVA WHILE DEVELOPING MVP	bernard	2021-05-16 16:32:42.238190	EDIT	DELETE
5	THE IOT TECHNOLOGY STACK MAY BE FALLING APART	bernard	2021-05-16 16:33:08.147485	EDIT	DELETE

Problem Set #6

UX Elements

Source: Elements of User Experience
by Jesse James Garrett



Solution Space
how and how much

Problem Space
who, what, and why

Business Goals

1. Pretending that you have completed your design works on the structure and skeleton planes based on your requirements, you're now ready to implement your prototype with Python Flask.
2. Create a registration and login page to add and authenticate users to your website.
3. Allow only admin user to add to-do tasks with expected completion date to the database (For your project update).
4. Integrate the added features to your Flask application with new and updated Bootstrap templates.
5. Upload the completed Jupyter Notebook to Github.
6. Send me the username and password of the admin user.

Thanks for joining me today!