* Pip3 install flask

Models:

* A class is a set of attributes – E.g: class Dogs:
  + Name = “Bilbo Waggins”
  + Dog-1 = Dogs()
* A model is a class that describes your table

Flask

* Framework
* It’s a structure for python code to make web apps
* Micro framework – can’t do much on its own, need other things like (SQLAlchemy)
* Structure of a flask app:
* My project
  + App.py – the app itself (it’s at the root of the project)
  + Index.html – the routes
  + Routes.py
  + Templates
  + --Init --.py
* To install we use – “(Sudo) pip3 install flask” – PIP is an appstore for python
* Simplest Flask App:
  + From flask import Flask - #second Flask is a class
  + app = Flask(\_ \_name\_ \_) - #app is a variable |
  + @app.route(‘/’) - # @ is a decorator | (‘/ ’) is an index
  + def hello\_world()
    - Return “hello world”
* If \_ \_ name \_ \_ == “ \_ \_main \_ \_”: - #YOU need this code for importing
  + App.run(debug=True) - #Shows error instead of “500 SERVER ERROR”
* @app.route(‘/home’, methods = []) - #By default it is a get request
* Redirect (‘/new\_page’) - #Redirects to new page
* @app.route(‘/’)
* Def home():
  + Return “hello world”
* Dynamic URL’s
* Variables into url.
* App.route(‘/home/<name>’)

**Database Layer**

* Flask App – #Performs operations
* Database – #Stores Data
* SQLAlchemy – #Works with flask to perform DB operations
* PyMySQL – #Connects
* App.config[‘SQLALCHEMY.DATABASE\_URI] = sqlike:///data.db #adding to dictionairy (sqlite is local)

**Connection String**

* Mysql + pymysql://username:password@ip:port/database\_name  
  sqlite:///data.db protocol://LOCATION

**Telling Python to connect**

* From flask\_sqlalchemy import SQLAlchemy 🡨 Class
* db = SQLAlchemy(app)
* app.config [‘SQLAlchemy\_DATABASE\_URI’] = “Connection String”
* db.create\_all() - #Creates all of our tables LOCALLY
* db.drop\_all() - #Deletes all of our tables LOCALLY