Berikut adalah tutorial langkah demi langkah untuk menginstal Flask dan pymongo pada Windows 10:

1. Pastikan Anda memiliki Python terinstal:
   * Buka browser dan kunjungi <https://www.python.org/downloads/>.
   * Unduh installer Python terbaru sesuai dengan arsitektur sistem Anda (32-bit atau 64-bit).
   * Jalankan installer Python dan ikuti instruksi yang diberikan untuk menginstal Python.
2. Buka Command Prompt (cmd):
   * Tekan tombol Windows + R pada keyboard untuk membuka "Run".
   * Ketik "cmd" dan tekan Enter. Ini akan membuka Command Prompt.
3. Buat Virtual Environment (Opsional):
   * Di Command Prompt, navigasikan ke folder proyek Anda dengan perintah cd /path/to/your/project.
   * Buat virtual environment dengan perintah python -m venv env.
   * Aktifkan virtual environment dengan perintah env\Scripts\activate.
4. Instal Flask:
   * Di Command Prompt, pastikan Anda berada dalam virtual environment jika Anda membuatnya.
   * Ketik perintah pip install Flask dan tekan Enter.
   * Tunggu sampai proses instalasi selesai.
5. Instal pymongo:
   * Di Command Prompt, pastikan Anda berada dalam virtual environment jika Anda membuatnya.
   * Ketik perintah pip install pymongo dan tekan Enter.
   * Tunggu sampai proses instalasi selesai.

Setelah Anda menyelesaikan langkah-langkah di atas, Anda sekarang memiliki Flask dan pymongo terinstal pada sistem Anda. Anda dapat menggunakan framework Flask dan modul pymongo untuk mengembangkan aplikasi web dengan menggunakan database MongoDB.

Contoh penggunaan Flask dan pymongo bisa ditemukan dalam tutorial sebelumnya yang telah saya berikan. Anda dapat mengikuti tutorial tersebut untuk memulai pengembangan aplikasi web menggunakan Flask dan pymongo.

Pastikan untuk selalu menjalankan perintah dalam virtual environment jika Anda membuatnya untuk memastikan dependensi yang terinstal terisolasi dalam lingkungan proyek Anda.

Semoga tutorial ini membantu Anda dalam menginstal Flask dan pymongo pada Windows 10 dan memulai pengembangan aplikasi web Anda.

Berikut adalah tutorial lengkap beserta source code untuk CRUD menggunakan Flask, MongoDB, Jinja, dan CSS, dengan contoh koleksi "mahasiswa" yang memiliki field "nim", "nama", dan "alamat".

1. Persiapan dan Instalasi:
   * Pastikan Anda telah menginstal Python dan pip.
   * Buatlah folder proyek baru dan buat virtual environment di dalamnya.
   * Aktifkan virtual environment.
   * Instal Flask dan pymongo melalui pip.
2. Struktur Proyek: Buat struktur direktori proyek sebagai berikut:

├── app.py

├── templates/

│ ├── base.html

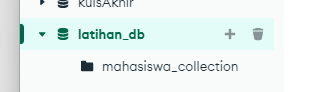
│ └── mahasiswa.html

├── static/

│ └── style.css

└── config.py

1. Pada mongodb pastikan sudah membuat database dengan nama **latihan\_db** dan collection dengan nama **mahasiswa\_collection**



1. Configurasi: Buatlah file config.py dengan isi sebagai berikut:

MONGO\_URI = "mongodb://localhost:27017/"

DATABASE\_NAME = "latihan\_db"

COLLECTION\_NAME = "mahasiswa\_collection"

1. Template Base: Buatlah file base.html di dalam folder templates dan masukkan kode berikut:

<!DOCTYPE html>

<html>

<head>

    <title>CRUD Mahasiswa</title>

    <link rel="stylesheet" type="text/css" href="*{{ url\_for('static', filename='style.css') }}*">

</head>

<body>

    <div class="container">

*{% block content %}{% endblock %}*

    </div>

</body>

</html>

1. CSS Styling: Buatlah file style.css di dalam folder static dan masukkan kode CSS sesuai keinginan.

html,

body {

*background-color*: #c4cacc;

*text-align*: center;

*font-family*: 'Raleway', Helvetica, sans-serif;

*min-width*: 320px;

}

header {

*font-size*: 2em;

*font-weight*: bold;

*margin*: 20px;

}

#form-outer {

*background-color*: rgb(250, 250, 250);

*margin*: 0 auto;

*border-radius*: 4px;

*width*: 75%;

*max-width*: 900px;

*padding*: 10px;

*padding-top*: 20px;

}

.labels {

*display*: inline-block;

*text-align*: right;

*width*: 40%;

*padding*: 5px;

*vertical-align*: top;

*margin-top*: 10px;

}

.rightTab {

*display*: inline-block;

*text-align*: left;

*width*: 48%;

*vertical-align*: middle;

}

.input-field {

*height*: 20px;

*width*: 280px;

*padding*: 5px;

*margin*: 10px;

*border*: 1px solid #c0c0c0;

*border-radius*: 2px;

}

#userAge {

*width*: 40px;

}

.userRatings,

input[type="checkbox"] {

*float*: left;

*margin-right*: 5px;

}

#submit {

*background-color*: #1a8cff;

*border-radius*: 4px;

*color*: white;

*font-size*: 1em;

*height*: 40px;

*width*: 96px;

*margin*: 10px;

*border*: 0px solid;

}

.button1 {

*font-size*: 16px;

*background-color*: #4CAF50;

*color*: black;

*border*: 2px solid #4CAF50;

*box-shadow*: 0 8px 16px 0 rgba(0,0,0,0.2), 0 6px 20px 0 rgba(0,0,0,0.19);

}

.button1:hover {

*background-color*: #4CAF50;

*color*: white;

*box-shadow*: 0 12px 16px 0 rgba(0,0,0,0.24),0 17px 50px 0 rgba(0,0,0,0.19);

  }

  .button3 {

*font-size*: 16px;

*background-color*: #f44336;

*color*: black;

*border*: 2px solid #f44336;

*box-shadow*: 0 8px 16px 0 rgba(0,0,0,0.2), 0 6px 20px 0 rgba(0,0,0,0.19);

  }

  .button3:hover {

*background-color*: #f44336;

*color*: white;

*box-shadow*: 0 12px 16px 0 rgba(0,0,0,0.24),0 17px 50px 0 rgba(0,0,0,0.19);

  }

.btn\_view {

*background-color*: #ffc61a;

*border-radius*: 4px;

*color*: white;

*font-size*: 1em;

*height*: 40px;

*width*: 96px;

*margin*: 10px;

*border*: 0px solid;

  }

#reset {

*background-color*: #1a8cff;

*border-radius*: 4px;

*color*: white;

*font-size*: 1em;

*height*: 40px;

*width*: 96px;

*margin*: 10px;

*border*: 0px solid;

}

.dropdown {

*height*: 35px;

*width*: 140px;

*padding*: 5px;

*margin*: 10px;

*margin-top*: 15px;

*border*: 1px solid #c0c0c0;

*border-radius*: 2px;

}

.radio, .checkbox {

*position*: relative;

*left*: -43px;

*margin-left*: 10px;

*display*: block;

*padding-bottom*: 10px;

}

@media screen and (*max-width*: 833px) {

  .input-field {

*width*: 80%;

  }

  select {

*width*: 90%;

  }

}

@media screen and (*max-width*: 520px) {

  .labels {

*width*: 100%;

*text-align*: left;

  }

  .rightTab {

*width*: 80%;

*float*: left;

  }

  .input-field {

*width*: 100%;

  }

  select {

*width*: 100%;

  }

}

.alert.success {

*color*: green;

*text-align*: center;

}

.alert.error {

*color*: red;

*text-align*: center;

}

table {

*table-layout*: fixed;

*width*: 100%;

*border-collapse*: separate;

*border*: 3px solid white;

  }

  th,

  td {

*padding*: 20px;

*text-align*: center;

  }

  tbody tr:nth-child(odd) {

*background-color*: #ff33cc;

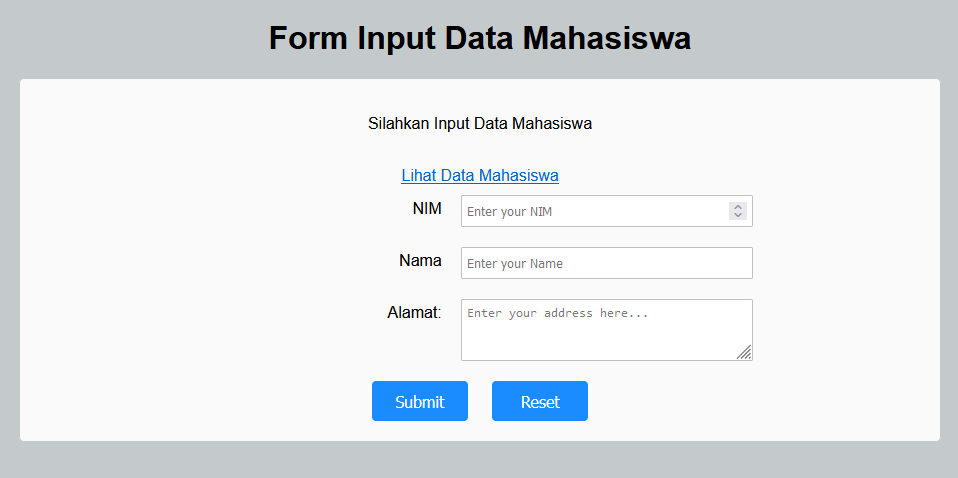
  }

  tbody tr:nth-child(even) {

*background-color*: #e495e4;

  }

1. Template Mahasiswa: Buatlah file mahasiswa.html di dalam folder templates dengan tampilan sebagai berikut :



Masukan kode program berikut pada mahasiswa.html

*{% extends 'base.html' %}*

*{% block content %}*

<h1 id="title">Form Input Data Mahasiswa</h1>

<div id="form-outer">

  <p id="description">

    Silahkan Input Data Mahasiswa

  </p>

<br>

  <a href="/list\_mahasiswa" class="nav-link">Lihat Data Mahasiswa</a>

  <br>

  <form id="survey-form" method="POST" action="/mahasiswa" onsubmit="return validateForm()">

      <div class="rowTab">

        <div class="labels">

          <label id="name-label" for="name">NIM </label>

        </div>

        <div class="rightTab">

          <input autofocus type="number" name="i\_nim" id="nim" class="input-field" placeholder="Enter your NIM" required>

        </div>

      </div>

      <div class="rowTab">

        <div class="labels">

          <label id="name-label" for="name">Nama </label>

        </div>

        <div class="rightTab">

          <input autofocus type="text" name="i\_nama" id="nama" class="input-field" placeholder="Enter your Name" required>

        </div>

      </div>

    <div class="rowTab">

      <div class="labels">

        <label for="address">Alamat: </label>

      </div>

      <div class="rightTab">

        <textarea id="comments" class="input-field" style="height:50px;resize:vertical;" name="i\_alamat" placeholder="Enter your address here..."></textarea>

      </div>

    </div>

    <button id="submit" type="submit">Submit</button>

    <button id="reset" type="reset">Reset</button>

  </form>

</div>

*{% endblock %}*

1. Membuat Koneksi ke MongoDB: Buatlah file app.py dan masukkan kode berikut:

from flask import Flask, render\_template, request, redirect

from pymongo import MongoClient

from bson.objectid import ObjectId

from config import MONGO\_URI, DATABASE\_NAME, COLLECTION\_NAME

app = Flask(\_\_name\_\_)

client = MongoClient(MONGO\_URI)

db = client[DATABASE\_NAME]

collection = db[COLLECTION\_NAME]

@app.route('/')

*def* index():

    return "halo Semua"

@app.route('/mahasiswa', *methods*=['GET', 'POST'])

*def* mahasiswa():

    if request.method == 'POST':

        nim = request.form['nim']

        nama = request.form['nama']

        alamat = request.form['alamat']

        data = {

            'nim': nim,

            'nama': nama,

            'alamat': alamat

            }

        collection.insert\_one(data)

        return render\_template('mahasiswa.html')

    return render\_template('mahasiswa.html')

if \_\_name\_\_ == '\_\_main\_\_':

    app.run(*debug*=True)

1. Tambahkan fungsi /list\_mahasiswa untuk berintarsi dengan database yang telah kita buat, dengan code program sebagai berikut :

@app.route('/list\_mahasiswa')

*def* get\_list\_mahasiswa():

    list\_data\_mhs = {}

    list\_data\_mhs = collection.find()

    return render\_template("list\_mahasiswa.html", *data\_mahasiswa*=list\_data\_mhs)

1. Buat tampilan untuk menampilkan list data mahasiswa, berikut kode html untuk list\_mahasiswa.html

*{% extends 'base.html' %}*

*{% block content %}*

<h1>Data Dosen </h1>

  <table id="data" class="table table-striped">

    <thead>

      <tr>

        <th>NIM</th>

        <th>Nama</th>

        <th>Alamat</th>

        <th>Aksi</th>

      </tr>

    </thead>

    <tbody>

*{% for data in data\_mahasiswa %}*

        <tr>

          <td>*{{ data.nim }}*</td>

          <td>*{{ data.nama }}*</td>

          <td>*{{ data.alamat }}*</td>

          <td>

            <a href="*{{ url\_for('edit\_mahasiswa', id=mhs.\_id) }}*">

                <button class="button1" type="submit">EDIT1</button>

            </a>

            <a href="#">

              <button class="button3" type="submit">DELETE</button>

          </a>

        </td>

        </tr>

*{% endfor %}*

    </tbody>

  </table>

*{% endblock %}*

1. Fungsi Update: Tambahkan fungsi edit\_mahasiswa dan update\_mahasiswa di dalam app.py:

@app.route('/edit/<id>')

*def* edit\_mahasiswa(*id*):

    mahasiswa = collection.find\_one({'\_id': ObjectId(*id*)})

    print (mahasiswa)

    return render\_template('edit\_mahasiswa.html', *mahasiswa*=mahasiswa)

@app.route('/update/<id>', *methods*=['POST'])

*def* update\_mahasiswa(*id*):

    print (" ----------------- ini id --------------")

    print (*id*)

    nim = request.form['i\_nim']

    nama = request.form['i\_nama']

    alamat = request.form['i\_alamat']

    data = {

        'nim': nim,

        'nama': nama,

        'alamat': alamat

        }

    collection.update\_one({'\_id': ObjectId(*id*)}, {'$set': data})

    return redirect('/list\_mahasiswa')

Buat Tampilan untuk merubah data, berikut adalah

Fungsi Delete: Tambahkan fungsi delete\_mahasiswa di dalam app.py:

python

 @app.route('/delete/<id>')

def delete\_mahasiswa(id):

collection.delete\_one({'\_id': ObjectId(id)})

return redirect('/')

 Menjalankan Aplikasi:

* Pastikan MongoDB berjalan di lokal.
* Jalankan aplikasi dengan menjalankan app.py:

python app.py

Buka browser dan akses http://localhost:5000.

Itulah tutorial lengkap beserta source code untuk CRUD menggunakan Flask, MongoDB, Jinja, dan CSS dengan contoh koleksi "mahasiswa". Anda dapat menyesuaikan CSS dan mengatur tampilan halaman sesuai kebutuhan Anda.