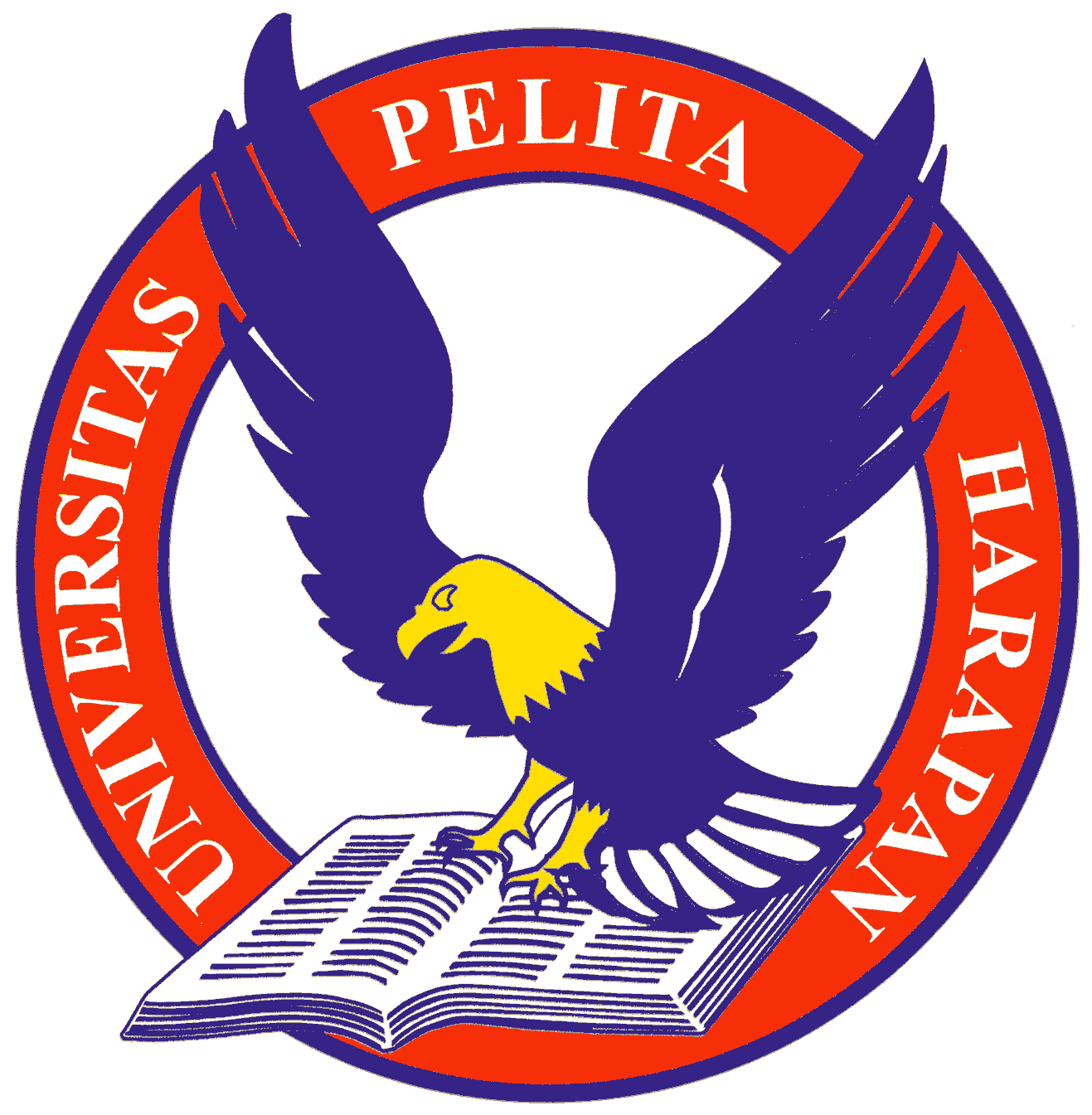
**UAS CRYPTO LIST AND PRICE USING COINCAP API**

****

**Disusun Oleh:**

|  |  |  |
| --- | --- | --- |
| **Nama** | **:** | **Julina Happy** |
| **NIM** | **:** | **03082180002** |
| **Kelas** | **:** | **18TI2** |

**PROGRAM STUDI TEKNIK INFORMATIKA**

**FAKULTAS ILMU KOMPUTER**

**UNIVERSITAS PELITA HARAPAN**

**MEDAN**

**2021**

Project ini dibuat dengan menggunakan Visual Studio Code, dan dibantu dengan framework Django, untuk frontend dan backend. Setelah berhasil mengambil(get) data dari CoinCap API, maka data tersebut akan dikelola untuk dapat menghasilkan output yang diinginkan yaitu List dan Price dari Crypto, dimana arsitektur dan code yang digunakan untuk pengimplementasian project ini dapat dilihat seperti berikut:

Arsitektur



Project tersebut dibuat dengan bantuan framework django, dimana untuk men-install django dan membuat project maka perlu dijalankan code segai berikut

py -m pip install django

django-admin startproject uas

django-admin startapp crypto

Setelah berhasil men-install django, maka untuk melihat apakah server sudah jalan dapat dilihat dengan:

py manage.py runserver

Dimana code di atas akan menampil lokasi server(URL) yang dapat diakses

Code yang digunakan dalam pembuatan Crypto List dan Price adalah sebagai berikut:

Folder project uas

urls.py

from django.contrib import admin

from django.urls import path, include

urlpatterns = [

    path('', include('crypto.urls')),

    path('admin/', admin.site.urls),

]

settings.py

from pathlib import Path

import os

*# Build paths inside the project like this: BASE\_DIR / 'subdir'.*

BASE\_DIR = Path(\_\_file\_\_).resolve().parent.parent

*# Quick-start development settings - unsuitable for production*

*# See https://docs.djangoproject.com/en/3.2/howto/deployment/checklist/*

*# SECURITY WARNING: keep the secret key used in production secret!*

SECRET\_KEY = 'django-insecure-uj5tp6^7%-xb23&y1+n&13gtzo\_&%!af4&#uiu1$o(vtn51ylo'

*# SECURITY WARNING: don't run with debug turned on in production!*

DEBUG = True

ALLOWED\_HOSTS = ['\*']

*# Application definition*

INSTALLED\_APPS = [

    'django.contrib.admin',

    'django.contrib.auth',

    'django.contrib.contenttypes',

    'django.contrib.sessions',

    'django.contrib.messages',

    'django.contrib.staticfiles',

    'crypto',

    'client'

]

MIDDLEWARE = [

    'django.middleware.security.SecurityMiddleware',

    'django.contrib.sessions.middleware.SessionMiddleware',

    'django.middleware.common.CommonMiddleware',

    'django.middleware.csrf.CsrfViewMiddleware',

    'django.contrib.auth.middleware.AuthenticationMiddleware',

    'django.contrib.messages.middleware.MessageMiddleware',

    'django.middleware.clickjacking.XFrameOptionsMiddleware',

]

ROOT\_URLCONF = 'uas.urls'

TEMPLATES = [

    {

        'BACKEND': 'django.template.backends.django.DjangoTemplates',

        'DIRS': [ os.path.join(BASE\_DIR, 'templates'), ],

        'APP\_DIRS': True,

        'OPTIONS': {

            'context\_processors': [

                'django.template.context\_processors.debug',

                'django.template.context\_processors.request',

                'django.contrib.auth.context\_processors.auth',

                'django.contrib.messages.context\_processors.messages',

            ],

        },

    },

]

WSGI\_APPLICATION = 'uas.wsgi.application'

*# Database*

*# https://docs.djangoproject.com/en/3.2/ref/settings/#databases*

DATABASES = {

    'default': {

        'ENGINE': 'django.db.backends.sqlite3',

        'NAME': BASE\_DIR / 'db.sqlite3',

    }

}

*# Password validation*

*# https://docs.djangoproject.com/en/3.2/ref/settings/#auth-password-validators*

AUTH\_PASSWORD\_VALIDATORS = [

    {

        'NAME': 'django.contrib.auth.password\_validation.UserAttributeSimilarityValidator',

    },

    {

        'NAME': 'django.contrib.auth.password\_validation.MinimumLengthValidator',

    },

    {

        'NAME': 'django.contrib.auth.password\_validation.CommonPasswordValidator',

    },

    {

        'NAME': 'django.contrib.auth.password\_validation.NumericPasswordValidator',

    },

]

*# Internationalization*

*# https://docs.djangoproject.com/en/3.2/topics/i18n/*

LANGUAGE\_CODE = 'en-us'

TIME\_ZONE = 'UTC'

USE\_I18N = True

USE\_L10N = True

USE\_TZ = True

*# Static files (CSS, JavaScript, Images)*

*# https://docs.djangoproject.com/en/3.2/howto/static-files/*

STATIC\_URL = '/static/'

STATICFILES\_DIRS = (

    os.path.join(BASE\_DIR, 'static'),

)

*# Default primary key field type*

*# https://docs.djangoproject.com/en/3.2/ref/settings/#default-auto-field*

DEFAULT\_AUTO\_FIELD = 'django.db.models.BigAutoField'

Folder crypto

File views.py

from django.shortcuts import render

from django.http import HttpResponse

import client

json\_assets = []

def index(request):

    return HttpResponse("hi, from crypto")

def coin\_list(request):

*global* json\_assets

    json\_assets = client.get\_assets()['data']

    response = ""

    for eachCoin in json\_assets:

        response += "<a style='display:block;' href='coin-list/" + eachCoin['id'] + "'>"

        response += eachCoin['id']

        response += "</a>"

*# <p>Bitcoin</p><p>Etherium</p>*

    context = {'coinList' : response }

    return render(request, 'index.html', context)

def coin\_view(request, coinName):

*global* json\_assets

    if(json\_assets == []):

        return HttpResponse("JSON IS KOSONG")

    response="Price Not Found"

    for eachCoin in json\_assets:

        if(eachCoin['id'] == coinName):

            response = "<p>" + eachCoin['priceUsd'] + "</p>"

            break

    context = { 'coinHarga': response }

    return render(request, 'price.html', context)

file urls.py

from django.urls import path, include

from . import views

urlpatterns = [

    path('', views.index, name="homescreen"),

    path('api/coin-list', views.coin\_list, name="coin list"),

    path('api/coin-list/<str:coinName>', views.coin\_view, name="Coin Desc")

]

Folder Client

File \_init\_.py

import requests *#digunakan untuk mengirim request html*

base\_url = 'https://api.coincap.io/v2/'

base\_payload = {}

base\_headers = {}

def get\_assets() -> str:

    """

    Returns a list of all assets

    TODO: add optional params

    """

    method = 'assets'

    url = base\_url + method

    response = requests.get(url)

    print(type(response))

    return response.json()

Folder templates

File index.html

{% load static %}

<!DOCTYPE *html*>

<html *lang*="en">

<head>

    <meta *charset*="UTF-8">

    <link *rel*="stylesheet" *href*="{% static 'css/style.css' %}">

    <meta *http-equiv*="X-UA-Compatible" *content*="IE=edge">

    <meta *name*="viewport" *content*="width=device-width, initial-scale=1.0">

    <title>Crypto</title>

</head>

<body>

    <div *id*="listKoinWrapper">

        {% autoescape off %}

        {{ coinList }}

        {% endautoescape %}

    </div>

</body>

</html>

File price.html

{% load static %}

<!DOCTYPE *html*>

<html *lang*="en">

<head>

    <meta *charset*="UTF-8">

    <link *rel*="stylesheet" *href*="{% static 'css/style.css' %}">

    <meta *http-equiv*="X-UA-Compatible" *content*="IE=edge">

    <meta *name*="viewport" *content*="width=device-width, initial-scale=1.0">

    <title>Halo</title>

</head>

<body>

    {% autoescape off %}

    {{ coinHarga }}

    {% endautoescape %}

</body>

</html>

Folder static\css

File style.css

html, body {

    margin: 0px;

    padding: 0px;

    height: 100%;

}

body {

    display: flex;

    flex-direction: column;

    align-items: center;

    justify-content: center;

}

a {

    text-decoration: none;

    color: inherit;

    background-color: #ffffff;

    padding: 8px 16px;

    border: 1px solid #c4c4c4;

    border-radius: 2px;

    margin: 30px;

    width: 300px;

}

a:hover {

    background-color: #fd8484;

    filter: drop-shadow(2px 0px 0px #f4f4f4);

}

#listKoinWrapper {

    width: 400px;

    height: 600px;

    overflow: auto;

    scroll-behavior: smooth;

}