



Nama: **Fathan Andi Kartagama (122140055)**  
Environment untuk Multimedia

Tugas Ke: **Worksheet 1: Setup Python**

Mata Kuliah: **Sistem Teknologi Multimedia (IF25-40305)**

Tanggal: August 26, 2025

## 1 Instruksi Tugas

### 1.1 Persiapan

- Menginstall Python 3.8 atau lebih baru (disarankan 3.10)
- Memilih salah satu tool manajemen environment: **conda**, **venv**, atau **uv**. Pada mata kuliah ini saya menggunakan manajemen environment **UV**
- Membuka terminal/command prompt
- Menyiapkan dokumen  $\text{\LaTeX}$  ini untuk dokumentasi

### 1.2 Bagian 1: Membuat Environment Python

#### 1.2.1 Menggunakan uv

```
1 # Install uv terlebih dahulu jika belum ada
2 pip install uv
3
4 # bisa dengan powershell (Windows)
5 powershell -ExecutionPolicy ByPass -c "irm https://astral.sh/uv/install.ps1 | iex"
6
7 # atau terminal (MacOS dan Linux)
8 curl -Lsf https://astral.sh/uv/install.sh | sh
9
10 # Membuat environment baru
11 uv venv multimedia-uv
12
13 # Mengaktifkan environment (Linux/Mac)
14 source multimedia-uv/bin/activate
15
16 # Mengaktifkan environment (Windows)
17 multimedia-uv\Scripts\activate
18
19 # Verifikasi environment aktif (MacOS dan Linux)
20 which python
21
22 # Verifikasi environment aktif (Windows)
23 ## Jika Menggunakan CMD
24 where python
25
26 ## Jika Menggunakan Powershell
27 Get-Command python | Select-Object -ExpandProperty Source
```

Kode 1: Membuat environment dengan uv

### Dokumentasikan di sini:

- Tool manajemen environment yang Anda pilih: [UV]
- Screenshot atau copy-paste output dari perintah verifikasi environment

```
PowerShell 7.5.2
Pataangg ~ 0ms powershell -ExecutionPolicy ByPass -c "irm https://astral.sh/uv/install.ps1 | iex"

Downloading uv 0.8.13 (x86_64-pc-windows-msvc)
Installing to C:\Users\Administrator\.local\bin
uv.exe
uvx.exe
uvw.exe
everything's installed!
Pataangg ~ 6.384s cd D:\Coding\Sistem-Teknologi-Multimedia
Pataangg Sistem-Teknologi-Multimedia 12ms uv venv multimedia-uv
Using CPython 3.10.18
Creating virtual environment at: multimedia-uv
Activate with: multimedia-uv\Scripts\activate
Pataangg Sistem-Teknologi-Multimedia 299ms .\multimedia-uv\Scripts\activate
(multimedia-uv) Pataangg Sistem-Teknologi-Multimedia 60ms which python
which: The term 'which' is not recognized as a name of a cmdlet, function, script file, or executable program.
Check the spelling of the name, or if a path was included, verify that the path is correct and try again.
(multimedia-uv) Pataangg Sistem-Teknologi-Multimedia 184ms python --version
Python 3.10.18
(multimedia-uv) Pataangg Sistem-Teknologi-Multimedia 205ms where python
(multimedia-uv) Pataangg Sistem-Teknologi-Multimedia 2ms Get-Command python | Select-Object -ExpandProperty Source
D:\Coding\Sistem-Teknologi-Multimedia\multimedia-uv\Scripts\python.exe
(multimedia-uv) Pataangg Sistem-Teknologi-Multimedia 46ms
```

Gambar 1: Output verifikasi environment

## 1.3 Bagian 2: Instalasi Library Multimedia

Setelah environment aktif, install library-library berikut:

### 1.3.1 Library Audio Processing

```
1 # Untuk pip (venv/uv):
2 pip install librosa soundfile scipy
```

Kode 2: Instalasi library audio

### 1.3.2 Library Image Processing

```
1 # Untuk pip (venv/uv):
2 pip install opencv-python pillow scikit-image matplotlib
```

Kode 3: Instalasi library image

### 1.3.3 Library Video Processing

```
1 # Untuk pip (venv/uv):
2 pip install moviepy imageio-ffmpeg
```

Kode 4: Instalasi library video

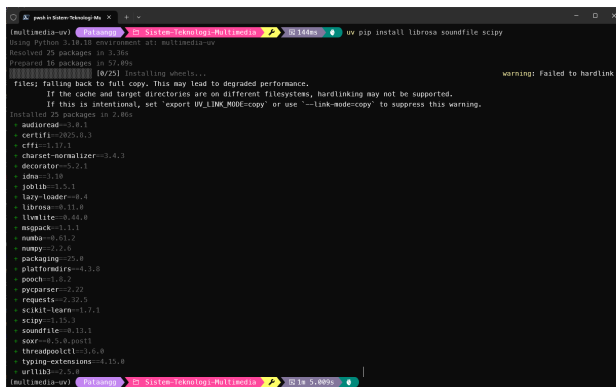
### 1.3.4 Library General Purpose

```
1 # Untuk pip (venv/uv):
2 pip install numpy pandas jupyter
```

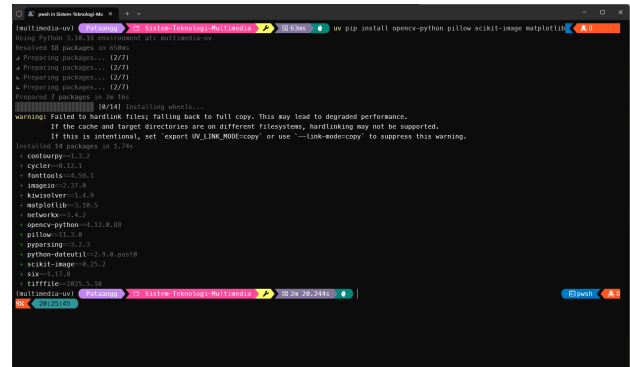
Kode 5: Instalasi library umum

Dokumentasikan di sini:

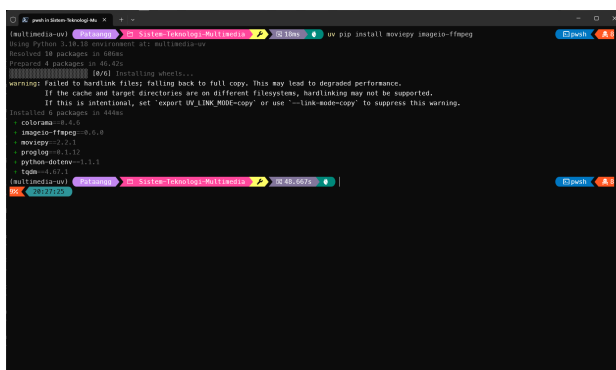
- Perintah instalasi yang Anda gunakan:
  - Audio: `uv pip install librosa soundfile scipy`
  - Image: `uv pip install opencv-python pillow scikit-image matplotlib`
  - Video: `uv pip install moviepy imageio-ffmpeg`
  - General: `uv pip install numpy pandas jupyter`
- Screenshot proses instalasi atau output sukses:



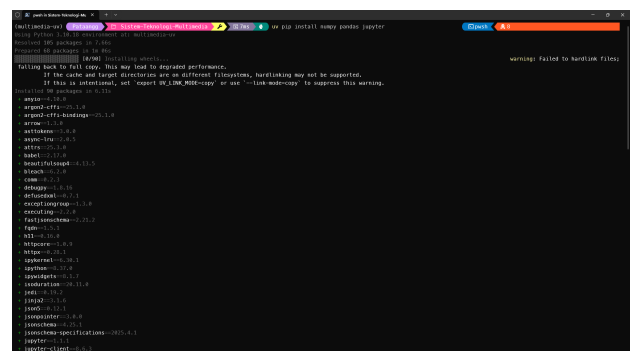
(a) Audio



(b) Image



(c) Video



(d) General

Gambar 2: Screenshots of installation steps in uv environment

- Daftar library yang berhasil diinstall dengan versinya:

```

anyio==4.10.0
argon2-cffi==25.1.0
argon2-cffi-bindings==25.1.0
arrow==1.3.0
asttokens==3.0.0
async-lru==2.0.5
attrs==25.3.0
audioread==3.0.1
babel==2.17.0
beautifulsoup4==4.13.5
bleach==6.2.0
certifi==2025.8.3
cffi==1.17.1
charset-normalizer==3.4.3
colorama==0.4.6
comm==0.2.3
contourpy==1.3.2
cycler==0.12.1
debugpy==1.8.16
decorator==5.2.1
defusedxml==0.7.1
exceptiongroup==1.3.0
executing==2.2.0
fastjsonschema==2.21.2
fonttools==4.59.1
fqdn==1.5.1
h11==0.16.0
httpcore==1.0.9
httpx==0.28.1
idna==3.10
imageio==2.37.0
imageio-ffmpeg==0.6.0
ipykernel==6.30.1
ipython==8.37.0
ipywidgets==8.1.7
isoduration==20.11.0
jedi==0.19.2
jinja2==3.1.6
joblib==1.5.1
json5==0.12.1
jsonpointer==3.0.0
jsonschema==4.25.1
jsonschema-specifications==2025.4.1
jupyter==1.1.1
jupyter-client==8.6.3
jupyter-console==6.6.3
jupyter-core==5.8.1
jupyter-events==0.12.0
jupyter-lsp==2.2.6
jupyter-server==2.17.0
jupyter-server-terminals==0.5.3
jupyterlab==4.4.6
jupyterlab-pygments==0.3.0
jupyterlab-server==2.27.3
jupyterlab-widgets==3.0.15
kiwisolver==1.4.9
lark==1.2.2
lazy-loader==0.4
librosa==0.11.0
llvmlite==0.44.0
markupsafe==3.0.2
matplotlib==3.10.5
matplotlib-inline==0.1.7
mistune==3.1.3
moviepy==2.2.1
msgpack==1.1.1
nbclient==0.10.2
nbconvert==7.16.6
nbformat==5.10.4
nest-asyncio==1.6.0
networkx==3.4.2
notebook==7.4.5
notebook-shim==0.2.4
numba==0.61.2
numpy==2.2.6
opencv-python==4.12.0.88
overrides==7.7.0
packaging==25.0
pandas==2.3.2
pandocfilters==1.5.1
parso==0.8.5
pillow==11.3.0
platformdirs==4.3.8
pooch==1.8.2
proglog==0.1.12
prometheus-client==0.22.1
prompt-toolkit==3.0.51
psutil==7.0.0
pure-eval==0.2.3
pycparser==2.22
pygments==2.19.2
pyparsing==3.2.3
python-dateutil==2.9.0.post0
python-dotenv==1.1.1
python-json-logger==3.3.0
pytz==2025.2
pywin32==311
pywinpty==3.0.0
pyyaml==6.0.2
pymz==27.0.2
referencing==0.36.2
requests==2.32.5
rfc3339-validator==0.1.4
rfc3986-validator==0.1.1
rfc3987-syntax==1.1.0
rpds-py==0.27.0
scikit-image==0.25.2
scikit-learn==1.7.1
scipy==1.15.3
send2trash==1.8.3
setuptools==80.9.0
six==1.17.0
sniffio==1.3.1
soundfile==0.13.1
soupsieve==2.7
soxr==0.5.0.post1
stack-data==0.6.3
terminado==0.18.1
threadpoolctl==3.6.0
tifffile==2025.5.10
tinycss2==1.4.0
tomli==2.2.1
tornado==6.5.2
tqdm==4.67.1
traitlets==5.14.3
types-python-dateutil==2.9.0.20250822
typing-extensions==4.15.0
tzdata==2025.2
uri-template==1.3.0
urllib3==2.5.0
wcwidth==0.2.13
webcolors==24.11.1
webencodings==0.5.1
websocket-client==1.8.0
widgetsnbextension==4.0.14

```

## 1.4 Bagian 3: Verifikasi Instalasi

Buat file Python sederhana untuk menguji semua library yang telah diinstall:

```

1 import importlib
2
3 # Daftar library dari requirements.txt
4 libraries = [
5     "anyio", "argon2", "arrow", "asttokens", "async_lru", "attrs", "audioread",
6     "babel", "bs4", "bleach", "certifi", "cffi", "charset-normalizer", "colorama",
7     "comm", "contourpy", "cycler", "debugpy", "decorator", "defusedxml", "exceptiongroup",
8     "executing", "fastjsonschema", "fonttools", "fqdn", "h11", "httpcore", "httpx",
9     "idna", "imageio", "imageio-ffmpeg", "ipykernel", "IPython", "ipywidgets",
10    "isoduration", "jedi", "jinja2", "joblib", "json5", "jsonpointer", "jsonschema",
11    "jupyter", "jupyter_client", "jupyter_console", "jupyter_core", "jupyter_events",
12    "jupyter_lsp", "jupyter_server", "jupyter_server_terminals", "jupyterlab",
13    "jupyterlab_pygments", "jupyterlab_server", "jupyterlab_widgets", "kiwisolver",
14    "lark", "lazy_loader", "librosa", "llvmlite", "markupsafe", "matplotlib",
15    "matplotlib_inline", "mistune", "moviepy", "msgpack", "nbclient", "nbconvert",
16    "nbformat", "nest_asyncio", "networkx", "notebook", "notebook_shim", "numba",
17    "numpy", "cv2", "overrides", "packaging", "pandas", "pandocfilters", "parso",
18    "PIL", "platformdirs", "pooch", "proglog", "prometheus_client", "prompt_toolkit",
19    "psutil", "pure_eval", "pycparser", "pygments", "pyparsing", "dateutil",

```

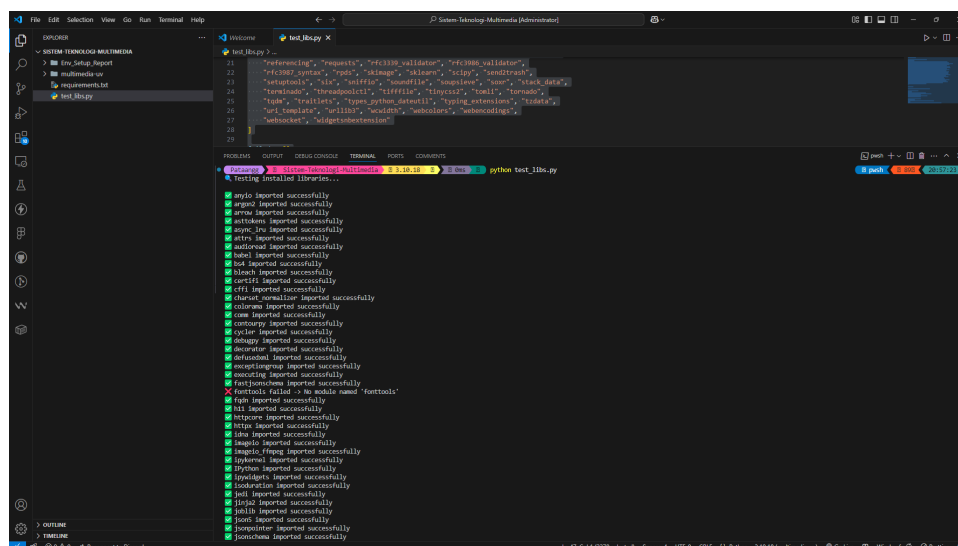
```

20 "dotenv", "python_json_logger", "pytz", "pywin32", "pywinpty", "yaml", "zmq",
21 "referencing", "requests", "rfc3339_validator", "rfc3986_validator",
22 "rfc3987_syntax", "rpds", "skimage", "sklearn", "scipy", "send2trash",
23 "setuptools", "six", "sniffio", "soundfile", "soupsieve", "soxr", "stack_data",
24 "terminado", "threadpoolctl", "tiffio", "tinycss2", "tomli", "tornado",
25 "tqdm", "traitlets", "types_python_dateutil", "typing_extensions", "tzdata",
26 "uri_template", "urllib3", "wcwidth", "webcolors", "webencodings",
27 "websocket", "widgetsnbextension"
28 ]
29
30 failed = []
31
32 print("    Testing installed libraries...\n")
33
34 for lib in libraries:
35     try:
36         importlib.import_module(lib)
37         print(f"    {lib} imported successfully")
38     except Exception as e:
39         print(f"    {lib} failed -> {e}")
40         failed.append(lib)
41
42 print("\n=== SUMMARY ===")
43 if not failed:
44     print("    All libraries imported successfully!")
45 else:
46     print(f"    Failed to import {len(failed)} libraries: {failed}")

```

Kode 6: Kode Python Sederhana untuk Cek library terinstall

Jalankan script dan dokumentasikan hasilnya:



Gambar 3: Script Python sederhana untuk cek library

## 1.5 Bagian 4: Simple Test dengan Sample Code

Buat dan jalankan contoh sederhana untuk setiap kategori multimedia:

### 1.5.1 Test Audio Processing

```
1 import numpy as np
```

```

2 import matplotlib.pyplot as plt
3
4 # Generate simple sine wave
5 duration = 2 # seconds
6 sample_rate = 44100
7 frequency = 440 # A4 note
8
9 t = np.linspace(0, duration, int(sample_rate * duration))
10 audio_signal = np.sin(2 * np.pi * frequency * t)
11
12 # Plot waveform
13 plt.figure(figsize=(10, 4))
14 plt.plot(t[:1000], audio_signal[:1000]) # Plot first 1000 samples
15 plt.title('Sine Wave (440 Hz)')
16 plt.xlabel('Time (s)')
17 plt.ylabel('Amplitude')
18 plt.grid(True)
19 plt.savefig('sine_wave_test.png', dpi=150, bbox_inches='tight')
20 plt.show()
21
22 print(f"Generated {duration}s sine wave at {frequency}Hz")
23 print(f"Sample rate: {sample_rate}Hz")
24 print(f"Total samples: {len(audio_signal)}")

```

Kode 7: Test audio processing sederhana

### 1.5.2 Test Image Processing

```

1 import numpy as np
2 import matplotlib.pyplot as plt
3 from PIL import Image
4
5 # Create a simple test image
6 width, height = 400, 300
7 image = np.zeros((height, width, 3), dtype=np.uint8)
8
9 # Add some patterns
10 image[:, :width//3, 0] = 255 # Red section
11 image[:, width//3:2*width//3, 1] = 255 # Green section
12 image[:, 2*width//3:, 2] = 255 # Blue section
13
14 # Add a white circle in the center
15 center_x, center_y = width//2, height//2
16 radius = 50
17 Y, X = np.ogrid[:height, :width]
18 mask = (X - center_x)**2 + (Y - center_y)**2 <= radius**2
19 image[mask] = [255, 255, 255]
20
21 # Display and save
22 plt.figure(figsize=(8, 6))
23 plt.imshow(image)
24 plt.title('Test Image with RGB Stripes and White Circle')
25 plt.axis('off')
26 plt.savefig('test_image.png', dpi=150, bbox_inches='tight')
27 plt.show()
28
29 print(f"Created test image: {width}x{height} pixels")
30 print(f"Image shape: {image.shape}")
31 print(f"Image dtype: {image.dtype}")

```

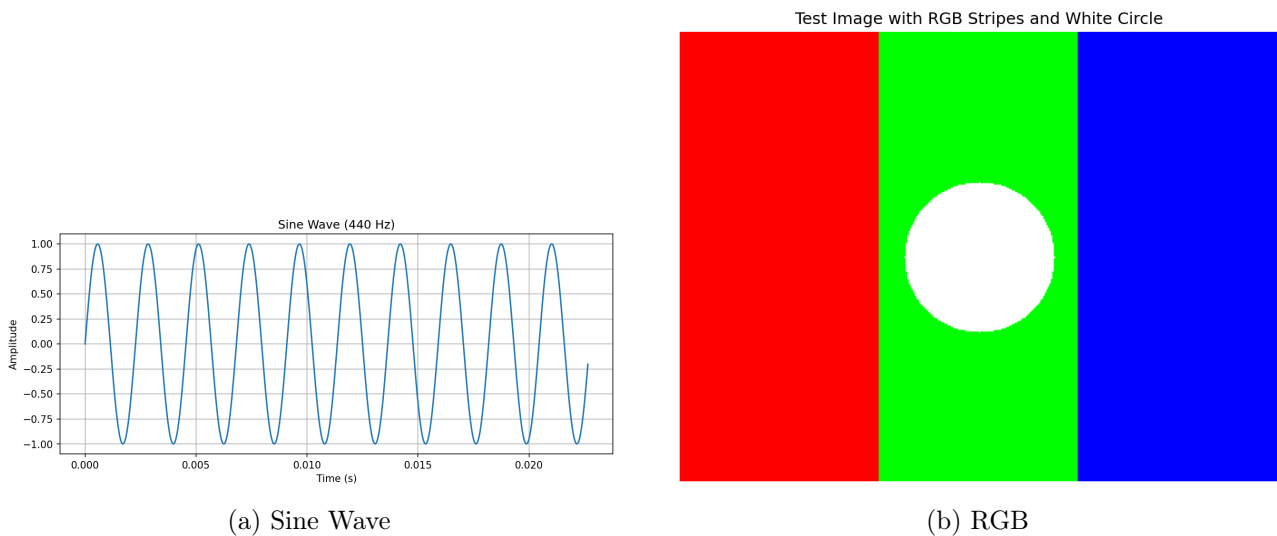
Kode 8: Test image processing sederhana

### Dokumentasikan hasil eksekusi:

- Screenshot output dari kedua script di atas

```
(multimedia-uv) Pataangg [Sistem-Teknologi-Multimedia] 3.10.18 344ms python .\test_multimedia.py
Generated 2s sine wave at 440Hz
Sample rate: 44100Hz
Total samples: 88200
Created test image: 400x300 pixels
Image shape: (300, 400, 3)
Image dtype: uint8
(multimedia-uv) Pataangg [Sistem-Teknologi-Multimedia] 3.10.18 5.44s 21:22:16
```

Gambar 4: Screenshot output terminal dari kedua script



Gambar 5: Gambar yang dihasilkan (sine\_wave\_test.png dan test\_image.png)

## 2 Bagian Laporan

### 2.1 Output Verifikasi Instalasi

Copy-paste output lengkap dari script **test\_multimedia.py** di sini:

```
1 [Generated 2s sine wave at 440Hz
2 Sample rate: 44100Hz
3 Total samples: 88200
4 Created test image: 400x300 pixels
5 Image shape: (300, 400, 3)
6 Image dtype: uint8]
```

Kode 9: Output verifikasi instalasi

## 2.2 Screenshot Hasil Test

```
(multimedia-uv) Pataangg [Sistem-Teknologi-Multimedia] 3.10.18 [344ms] python .\test_multimedia.py
Generated 2s sine wave at 440Hz
Sample rate: 44100Hz
Total samples: 88200
Created test image: 400x300 pixels
Image shape: (300, 400, 3)
Image dtype: uint8
(multimedia-uv) Pataangg [Sistem-Teknologi-Multimedia] 3.10.18 [5.44s]
```

Gambar 6: Screenshot output terminal dari kedua script

## 2.3 Analisis dan Refleksi

Jawab pertanyaan berikut:

1. Mengapa penting menggunakan environment terpisah untuk project multimedia?

Untuk **isolasi dependensi**. Supaya gaada konflik antar proyek yang membutuhkan versi library yang berbeda, memastikan **reproducibility** (proyek dapat dijalankan di mana saja dengan hasil yang sama), dan menjaga instalasi Python global tetap bersih

2. Apa perbedaan utama antara conda, venv, dan uv? Mengapa Anda memilih tool yang Anda gunakan?

**venv** Bawaan Python, ringan, hanya mengelola paket Python

**conda** Mengelola paket Python, versi Python, dan dependensi non-Python (misal: CUDA). Sangat kuat namun lebih lambat

**uv** Installer dan manajer environment modern yang **sangat cepat** karena ditulis dalam Rust

**uv** dipilih karena **kecepatannya** yang superior dalam menginstal dan menyelesaikan dependensi, sangat menghemat waktu pada proyek dengan banyak library

3. Library mana yang paling sulit diinstall dan mengapa?

Gaada yang sulit sih semuanya bisa terinstall dengan baik

4. Bagaimana cara mengatasi masalah dependency conflict jika terjadi?

- Baca pesan error untuk mengidentifikasi paket yang konflik
- Coba upgrade paket utama yang menyebabkan masalah
- Tentukan versi paket yang kompatibel untuk semua dependensi secara manual
- Gunakan tool manajemen seperti **poetry** buat lock versi library
- Kalo semua gagal, buat ulang environment dari awal wkukwk

5. Jelaskan fungsi dari masing-masing library yang berhasil Anda install!

- **Audio Processing**
  - **librosa**: Analisis dan ekstraksi fitur audio (spektogram, MFCC)
  - **soundfile**: Membaca dan menulis berbagai format file audio
  - **scipy**: Komputasi saintifik dan pemrosesan sinyal digital
- **Image Processing**
  - **opencv-python**: Pustaka utama untuk computer vision

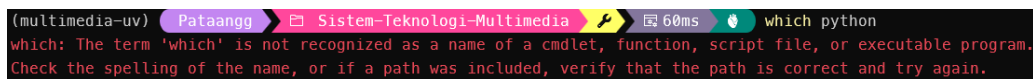


- *Pillow*: Manipulasi gambar dasar (*crop, resize, rotasi*)
- *scikit-image*: Algoritma analisis citra untuk riset
- *matplotlib*: Visualisasi data dan menampilkan gambar
- **Video Processing**
  - *moviepy*: Editing video melalui skrip (*memotong, menggabung*)
  - *imageio-ffmpeg*: Wrapper untuk FFMpeg agar bisa membaca/menulis format video
- **General Purpose**
  - *numpy*: Komputasi numerik fundamental dengan array *N-dimensi*
  - *pandas*: Analisis dan manipulasi data terstruktur (*tabel*)
  - *jupyter*: Lingkungan pemrograman interaktif (*Notebook*)

## 2.4 Troubleshooting

Dokumentasikan masalah yang Anda hadapi (jika ada) dan cara mengatasinya:

- **Masalah 1:** *["Which" command tidak dikenali]*



Gambar 7: Which command invalid

Solusi:

- Step 1. Tanya GPT-5
- Step 2. Ternyata masalahnya karena saya pake windows dan command "Which" itu cuma untuk MacOS dan Linux
- Step 3. Akhirnya saya coba command "where python" ternyata gabisa juga. Sebab command itu bisanya dipake di CMD sedangkan saya pakenya terminal (powershell)

## 3 Export Environment untuk Reproduksi

Sebagai langkah terakhir, export environment Anda agar dapat direproduksi:

### 3.1 Untuk venv/uv

```
1 pip freeze > requirements.txt
```

Kode 10: Export pip requirements

Copy-paste isi file `environment.yml` atau `requirements.txt` di sini:

```
1 [anyio==4.10.0
2 argon2-cffi==25.1.0
3 argon2-cffi-bindings==25.1.0
4 arrow==1.3.0
5 asttokens==3.0.0
6 async-lru==2.0.5
7 attrs==25.3.0
8 audioread==3.0.1
9 babel==2.17.0
10 beautifulsoup4==4.13.5
```

```
11 bleach==6.2.0
12 certifi==2025.8.3
13 cffi==1.17.1
14 charset-normalizer==3.4.3
15 colorama==0.4.6
16 comm==0.2.3
17 contourpy==1.3.2
18 cycler==0.12.1
19 debugpy==1.8.16
20 decorator==5.2.1
21 defusedxml==0.7.1
22 exceptiongroup==1.3.0
23 executing==2.2.0
24 fastjsonschema==2.21.2
25 fonttools==4.59.1
26 fqdn==1.5.1
27 h11==0.16.0
28 httpcore==1.0.9
29 httpx==0.28.1
30 idna==3.10
31 imageio==2.37.0
32 imageio-ffmpeg==0.6.0
33 ipykernel==6.30.1
34 ipython==8.37.0
35 ipywidgets==8.1.7
36 isoduration==20.11.0
37 jedi==0.19.2
38 jinja2==3.1.6
39 joblib==1.5.1
40 json5==0.12.1
41 jsonpointer==3.0.0
42 jsonschema==4.25.1
43 jsonschema-specifications==2025.4.1
44 jupyter==1.1.1
45 jupyter-client==8.6.3
46 jupyter-console==6.6.3
47 jupyter-core==5.8.1
48 jupyter-events==0.12.0
49 jupyter-lsp==2.2.6
50 jupyter-server==2.17.0
51 jupyter-server-terminals==0.5.3
52 jupyterlab==4.4.6
53 jupyterlab-pygments==0.3.0
54 jupyterlab-server==2.27.3
55 jupyterlab-widgets==3.0.15
56 kiwisolver==1.4.9
57 lark==1.2.2
58 lazy-loader==0.4
59 librosa==0.11.0
60 llvmlite==0.44.0
61 markupsafe==3.0.2
62 matplotlib==3.10.5
63 matplotlib-inline==0.1.7
64 mistune==3.1.3
65 moviepy==2.2.1
66 msgpack==1.1.1
67 nbclient==0.10.2
68 nbconvert==7.16.6
69 nbformat==5.10.4
70 nest-asyncio==1.6.0
71 networkx==3.4.2
72 notebook==7.4.5
```

```
73 notebook-shim==0.2.4
74 numba==0.61.2
75 numpy==2.2.6
76 opencv-python==4.12.0.88
77 overrides==7.7.0
78 packaging==25.0
79 pandas==2.3.2
80 pandocfilters==1.5.1
81 parso==0.8.5
82 pillow==11.3.0
83 platformdirs==4.3.8
84 pooch==1.8.2
85 proglog==0.1.12
86 prometheus-client==0.22.1
87 prompt-toolkit==3.0.51
88 psutil==7.0.0
89 pure-eval==0.2.3
90 pycparser==2.22
91 pygments==2.19.2
92 pyparsing==3.2.3
93 python-dateutil==2.9.0.post0
94 python-dotenv==1.1.1
95 python-json-logger==3.3.0
96 pytz==2025.2
97 pywin32==311
98 pywinpty==3.0.0
99 pyyaml==6.0.2
100 pyzmq==27.0.2
101 referencing==0.36.2
102 requests==2.32.5
103 rfc3339-validator==0.1.4
104 rfc3986-validator==0.1.1
105 rfc3987-syntax==1.1.0
106 rpds-py==0.27.0
107 scikit-image==0.25.2
108 scikit-learn==1.7.1
109 scipy==1.15.3
110 send2trash==1.8.3
111 setuptools==80.9.0
112 six==1.17.0
113 sniffio==1.3.1
114 soundfile==0.13.1
115 soupsieve==2.7
116 soxr==0.5.0.post1
117 stack-data==0.6.3
118 terminado==0.18.1
119 threadpoolctl==3.6.0
120 tifffile==2025.5.10
121 tinycss2==1.4.0
122 tomli==2.2.1
123 tornado==6.5.2
124 tqdm==4.67.1
125 traitlets==5.14.3
126 types-python-dateutil==2.9.0.20250822
127 typing-extensions==4.15.0
128 tzdata==2025.2
129 uri-template==1.3.0
130 urllib3==2.5.0
131 wcwidth==0.2.13
132 webcolors==24.11.1
133 webencodings==0.5.1
134 websocket-client==1.8.0
```

```
135 widgetsnbextension==4.0.14
136 ]
```

Kode 11: Environment/Requirements file

## 4 Kesimpulan

Tuliskan kesimpulan Anda mengenai:

- Pengalaman setup Python environment untuk multimedia
- Persiapan untuk project multimedia selanjutnya
- Saran untuk mahasiswa lain yang akan melakukan setup serupa

*Setup environment Python buat multimedia itu wajib hukumnya biar nggak ribet sama konflik library. Makanya pakai **uv** atau **venv** itu bukan sekadar opsional, tapi memang keharusan. Dari pengalaman, **uv** kerasa banget cepetnya pas install paket gede kayak **OpenCV** dan **SciPy***

**Biar aman ke depannya:**

1. Selalu mulai dari environment baru yang bersih
2. Habis install, langsung simpan **requirements.txt** pakai **uv pip freeze > requirements.txt**
3. Pahami fungsi inti tiap library biar nggak salah pilih alat

**Tips buat temen-temen lain:**

- Jangan install global, bikin environment dari awal
- Lebih enak langsung pakai **uv**, simpel dan cepat
- Kalau error pas install, baca pesan error dulu—biasanya jelas apa yang kurang
- Install library seperlunya aja, biar gampang kalau ada masalah

## 5 Referensi

- [UV Official Documentation](#)
- [ChatGPT](#)
- [LaTeX Wikibook: Comprehensive Guide](#)
- [Overleaf Learn LaTeX Documentation](#)

## 6 Lampiran

- [Github](#)