



Nama: **Zidan Raihan (122140100)** Tugas Ke: **Worksheet 1: Setup Python Environment untuk Multimedia**

Mata Kuliah: **Sistem Teknologi Multimedia (IF25-40305)** Tanggal: August 29, 2025

1 Tujuan Pembelajaran

Setelah menyelesaikan worksheet ini, mahasiswa diharapkan mampu:

- Memahami pentingnya manajemen environment Python untuk pengembangan multimedia
- Menginstall dan mengkonfigurasi Python environment menggunakan conda, venv, atau uv
- Menginstall library-library Python yang diperlukan untuk multimedia processing
- Memverifikasi instalasi dengan mengimpor dan menguji library multimedia
- Mendokumentasikan proses konfigurasi dan hasil pengujian dalam format \LaTeX

2 Latar Belakang

Python telah menjadi bahasa pemrograman yang sangat populer untuk multimedia processing karena memiliki ekosistem library yang sangat kaya. Namun, untuk dapat bekerja dengan multimedia secara efektif, kita perlu mengatur environment Python dengan benar dan menginstall library-library yang tepat.

Manajemen environment Python sangat penting untuk:

- Menghindari konflik antar library (dependency conflict)
- Memastikan reproducibility dari project
- Memudahkan kolaborasi antar developer
- Memisahkan project yang berbeda dengan requirement yang berbeda

3 Instruksi Tugas

3.1 Persiapan

Sebelum memulai, pastikan Anda telah:

- Menginstall Python 3.8 atau lebih baru di sistem Anda
- Memilih salah satu tool manajemen environment: **conda**, **venv**, atau **uv**
- Membuka terminal/command prompt
- Menyiapkan dokumen \LaTeX ini untuk dokumentasi

3.2 Bagian 1: Membuat Environment Python

Pilih **SALAH SATU** dari tiga opsi berikut dan ikuti langkah-langkahnya:

3.2.1 Opsi 1: Menggunakan Conda (Direkomendasikan untuk pemula)

Jalankan perintah berikut di terminal:

```
1 # Membuat environment baru dengan nama 'multimedia'
2 conda create -n multimedia python=3.11
3
4 # Mengaktifkan environment
5 conda activate multimedia
6
7 # Verifikasi environment aktif
8 conda info --envs
```

Kode 1: Membuat environment dengan Conda

3.2.2 Opsi 2: Menggunakan venv (Built-in Python)

```
1 # Membuat environment baru
2 python3 -m venv multimedia-env
3
4 # Mengaktifkan environment (Linux/Mac)
5 source multimedia-env/bin/activate
6
7 # Mengaktifkan environment (Windows)
8 # multimedia-env\Scripts\activate
9
10 # Verifikasi environment aktif
11 which python
```

Kode 2: Membuat environment dengan venv

3.2.3 Opsi 3: Menggunakan uv (Modern dan cepat)

```
1 # Install uv terlebih dahulu jika belum ada
2 # pip install uv
3
4 # Membuat environment baru
5 uv venv multimedia-uv
6
7 # Mengaktifkan environment (Linux/Mac)
8 source multimedia-uv/bin/activate
9
10 # Mengaktifkan environment (Windows)
11 # multimedia-uv\Scripts\activate
12
13 # Verifikasi environment aktif
14 which python
```

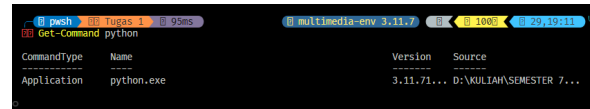
Kode 3: Membuat environment dengan uv

Dokumentasikan di sini:

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



(a) Mengaktifkan environment



(b) Verifikasi environment

3.3 Bagian 2: Instalasi Library Multimedia

Setelah environment aktif, install library-library berikut:

3.3.1 Library Audio Processing

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

Kode 4: Instalasi library audio

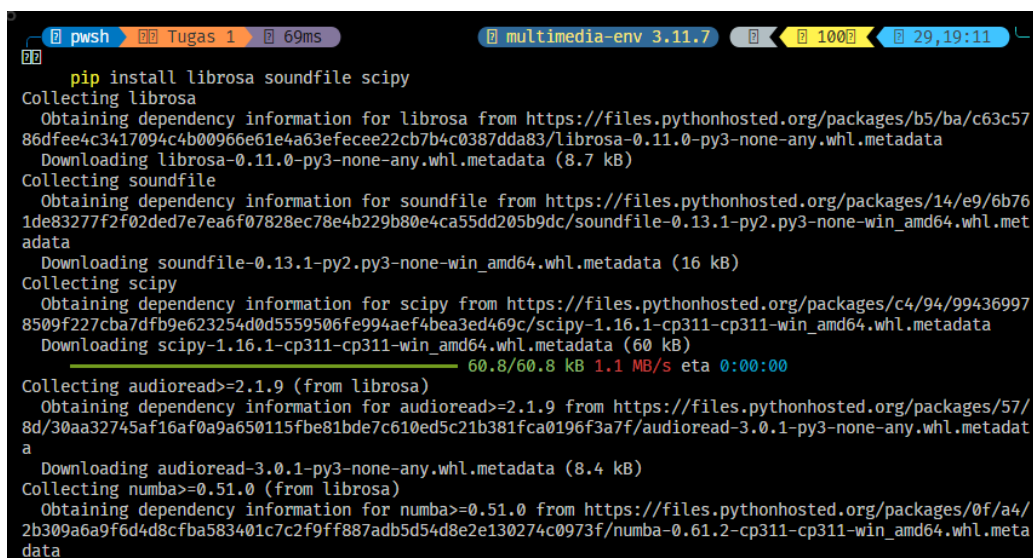


Figure 2: Proses instalasi librosa, soundfile, dan scipy

3.3.2 Library Image Processing

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

Kode 5: Instalasi library image

```

pwsh Tugas 1 1m 20s 694ms multimedia-env 3.11.7 100% 29,19:13
pip install opencv-python pillow scikit-image matplotlib
Collecting opencv-python
  Obtaining dependency information for opencv-python from https://files.pythonhosted.org/packages/fa/80/eb88edc2e2b11cd2dd2e56f1c80b5784d11d6e6b7f04a1145df64df40065/opencv_python-4.12.0.88-cp37-abi3-win_amd64.whl.metadata
  Downloading opencv_python-4.12.0.88-cp37-abi3-win_amd64.whl.metadata (19 kB)
Collecting pillow
  Obtaining dependency information for pillow from https://files.pythonhosted.org/packages/f1/cc/29c0f5d64ab8eae20f3232da8f8571660aa0ab4b8f1331da5c2f5f9a938e/pillow-11.3.0-cp311-cp311-win_amd64.whl.metadata
  Downloading pillow-11.3.0-cp311-cp311-win_amd64.whl.metadata (9.2 kB)
Collecting scikit-image
  Obtaining dependency information for scikit-image from https://files.pythonhosted.org/packages/de/ec/b57c500ee85885df5f2188f8bb70398481393a69de44a00d6f1d055f103c/scikit_image-0.25.2-cp311-cp311-win_amd64.whl.metadata
  Downloading scikit_image-0.25.2-cp311-cp311-win_amd64.whl.metadata (14 kB)
Collecting matplotlib
  Obtaining dependency information for matplotlib from https://files.pythonhosted.org/packages/b7/81/3200b792a5e8b354f31f4101ad7834743ad07b6d620259f2059317b25e4d/matplotlib-3.10.5-cp311-cp311-win_amd64.whl.metadata
  Downloading matplotlib-3.10.5-cp311-cp311-win_amd64.whl.metadata (11 kB)
Requirement already satisfied: numpy<2.3.0,>=2 in d:\kuliah\semester 7\multimedia\tugas 1\multimedia-env\lib\site-packages (from opencv-python) (2.2.6)
Requirement already satisfied: scipy>=1.11.4 in d:\kuliah\semester 7\multimedia\tugas 1\multimedia-env\lib\site-packages (from scikit-image) (1.16.1)
Collecting networkx>=3.0 (from scikit-image)
  Obtaining dependency information for networkx>=3.0 from https://files.pythonhosted.org/packages/eb/8d/776adee7bbf76365fdd7f2552710282c79a4ead5d2a46408c9043a2b70ba/networkx-3.5-py3-none-any.whl.metadata
  Downloading networkx-3.5-py3-none-any.whl.metadata (6.3 kB)
Collecting imageio!>=2.35.0,>=2.33 (from scikit-image)
  Obtaining dependency information for imageio!>=2.35.0,>=2.33 from https://files.pythonhosted.org/packages/cb/bd/b394387b598ed84d8d0fa90611a90bee0adc2021820ad5729f7ced74a8e2/imageio-2.37.0-py3-none-any.whl.metadata
  Downloading imageio-2.37.0-py3-none-any.whl.metadata (5.2 kB)
Collecting tifffile>=2022.8.12 (from scikit-image)
  Obtaining dependency information for tifffile>=2022.8.12 from https://files.pythonhosted.org/packages/56/b3/23e760215910609914dd99aba23ce1c72a3bcbe046ee44f45adf740452/tifffile-2025.8.28-py3-none-any.whl.metadata

```

Figure 3: Proses instalasi opencv-python, pillow, scikit-image, dan matplotlib

3.3.3 Library Video Processing

```

1 # Untuk pip (venv/uv):
2 pip install moviepy

```

Kode 6: Instalasi library video

```

pwsh Tugas 1 52s 97ms multimedia-env 3.11.7 100% 29,19:15
pip install moviepy
Collecting moviepy
  Downloading moviepy-2.2.1-py3-none-any.whl.metadata (6.9 kB)
Requirement already satisfied: decorator<6.0,>=4.0.2 in d:\kuliah\semester 7\multimedia\tugas 1\multimedia-env\lib\site-packages (from moviepy) (5.2.1)
Requirement already satisfied: imageio<3.0,>=2.5 in d:\kuliah\semester 7\multimedia\tugas 1\multimedia-env\lib\site-packages (from moviepy) (2.37.0)
Collecting imageio_ffmpeg>=0.2.0 (from moviepy)
  Downloading imageio_ffmpeg-0.6.0-py3-none-win_amd64.whl.metadata (1.5 kB)
Requirement already satisfied: numpy>=1.25.0 in d:\kuliah\semester 7\multimedia\tugas 1\multimedia-env\lib\site-packages (from moviepy) (2.2.6)
Collecting proglog<=1.0.0 (from moviepy)
  Downloading proglog-0.1.12-py3-none-any.whl.metadata (794 bytes)
Collecting python-dotenv>=0.10 (from moviepy)
  Downloading python_dotenv-1.1.1-py3-none-any.whl.metadata (24 kB)
Requirement already satisfied: pillow<12.0,>=9.2.0 in d:\kuliah\semester 7\multimedia\tugas 1\multimedia-env\lib\site-packages (from moviepy) (11.3.0)
Collecting tqdm (from proglog<=1.0.0->moviepy)
  Using cached tqdm-4.67.1-py3-none-any.whl.metadata (57 kB)

```

Figure 4: Proses instalasi moviepy

3.3.4 Library General Purpose

```

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

```

Kode 7: Instalasi library umum

```

pwsh Tugas 1 13s 602ms multimedia-env 3.11.7 100% 29,19:16
pip install numpy pandas jupyter
Requirement already satisfied: numpy in d:\kuliah\semester 7\multimedia\tugas 1\multimedia-env\lib\site-packages (2.2.6)
Collecting pandas
  Downloading pandas-2.3.2-cp311-cp311-win_amd64.whl.metadata (19 kB)
Collecting jupyter
  Downloading jupyter-1.1.1-py2.py3-none-any.whl.metadata (2.0 kB)
Requirement already satisfied: python-dateutil>=2.8.2 in d:\kuliah\semester 7\multimedia\tugas 1\multimedia-env\lib\site-packages (from pandas) (2.9.0.post0)
Collecting pytz>=2020.1 (from pandas)
  Using cached pytz-2025.2-py2.py3-none-any.whl.metadata (22 kB)
Collecting tzdata>=2022.7 (from pandas)
  Using cached tzdata-2025.2-py2.py3-none-any.whl.metadata (1.4 kB)
Collecting notebook (from jupyter)
  Downloading notebook-7.4.5-py3-none-any.whl.metadata (10 kB)
Collecting jupyter-console (from jupyter)
  Downloading jupyter_console-6.6.3-py3-none-any.whl.metadata (5.8 kB)
Collecting nbconvert (from jupyter)
  Using cached nbconvert-7.16.6-py3-none-any.whl.metadata (8.5 kB)
Collecting ipykernel (from jupyter)
  Downloading ipykernel-6.30.1-py3-none-any.whl.metadata (6.2 kB)

```

Figure 5: Proses instalasi numpy, pandas, dan jupyter

Dokumentasikan di sini:

- Daftar library yang berhasil diinstall dengan versinya

```

Image dtype: uint8
pwsh Tugas 1 5s 280ms multimedia-env 3.11.7 100% 29,19:31
python -u "d:\KULIAH\SEMESTER 7\Multimedia\tugas 1\test_libraries.py"
librosa: 0.11.0
soundfile: 0.13.1
scipy: 1.16.1
opencv-python: 4.12.0
pillow: 11.3.0
scikit-image: 0.25.2
matplotlib: 3.10.5
numpy: 2.2.6
pandas: 2.3.2

```

Figure 6: Proses instalasi numpy, pandas, dan jupyter

3.4 Bagian 3: Verifikasi Instalasi

Buat file Python sederhana untuk menguji semua library yang telah diinstall. Berikut adalah contoh script untuk verifikasi:

```

1 import librosa
2 import soundfile
3 import scipy
4 import cv2
5 import PIL
6 import skimage
7 import matplotlib
8 import numpy
9 import pandas
10
11 print("librosa:", librosa.__version__)
12 print("soundfile:", soundfile.__version__)
13 print("scipy:", scipy.__version__)
14 print("opencv-python:", cv2.__version__)
15 print("pillow:", PIL.__version__)
16 print("scikit-image:", skimage.__version__)
17 print("matplotlib:", matplotlib.__version__)

```

```

18 print("numpy:", numpy.__version__)
19 print("pandas:", pandas.__version__)

```

Kode 8: Instalasi library video

Jalankan script dan dokumentasikan hasilnya: Hasil dokumentasi dapat dilihat pada Gambar 6

3.5 Bagian 4: Simple Test dengan Sample Code

Buat dan jalankan contoh sederhana untuk setiap kategori multimedia:

3.5.1 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 9: Test audio processing sederhana

3.5.2 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 10: Test image processing sederhana

Dokumentasikan hasil eksekusi:

- Screenshot output dari kedua script di atas
- Gambar yang dihasilkan (sine_wave_test.png dan test_image.png)

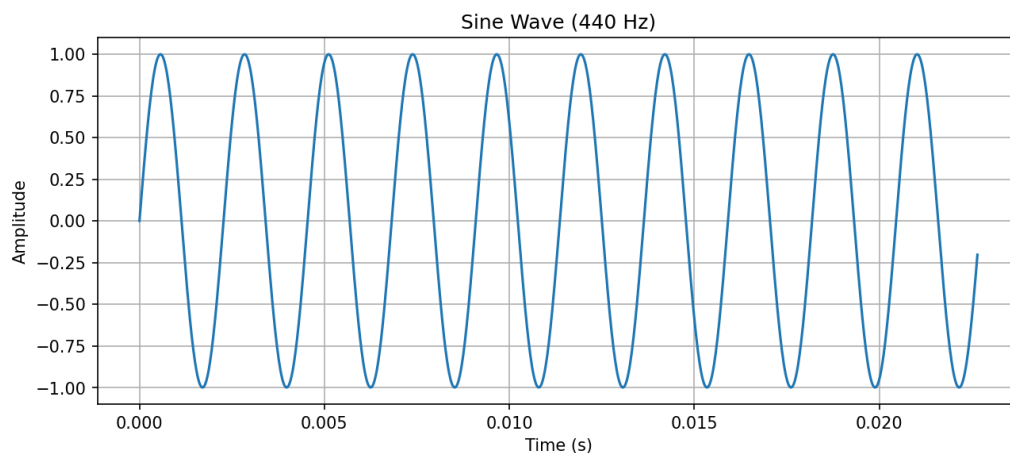


Figure 7: Output dari script test audio (sine wave)

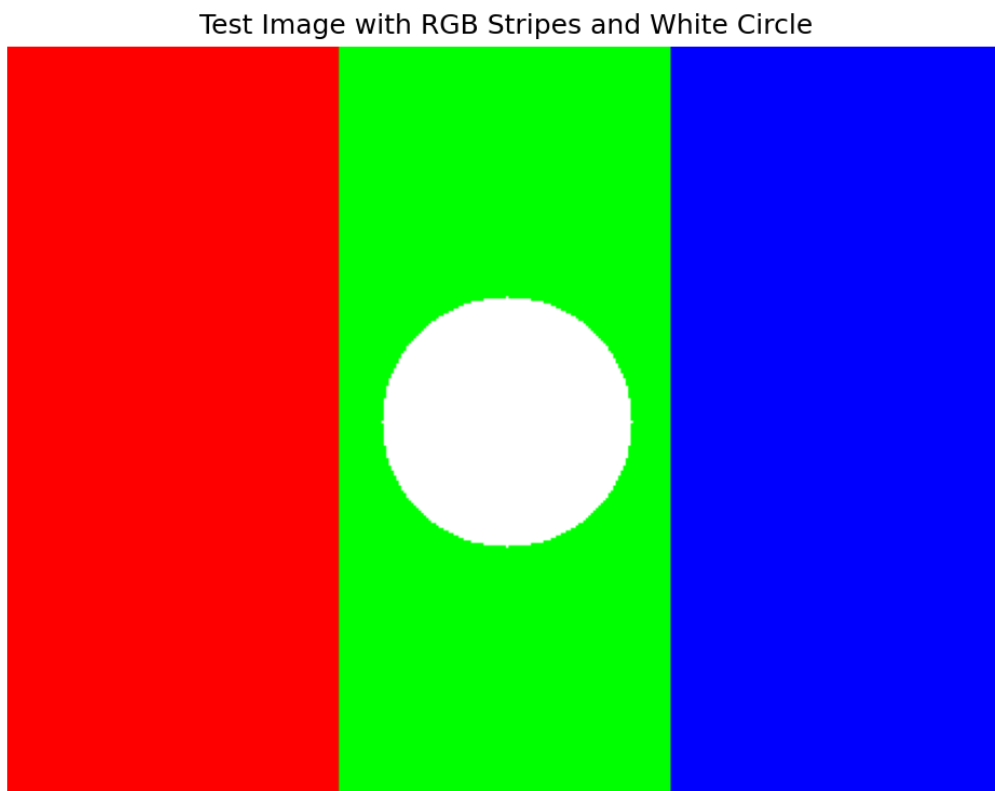


Figure 8: Output dari script test image (RGB stripes dengan circle)

- Error message jika ada dan cara mengatasinya

4 Bagian Laporan

4.1 Output Verifikasi Instalasi

Copy-paste output lengkap dari script **test_multimedia.py** di sini:

```

1 Get-Command python
2
3 CommandType      Name                               Version      Source
4 -----
5 Application       python.exe                        3.11.71...  D:\KULIAH\SEMESTER 7\
   Multimedia\Tugas 1\multim...

```

Kode 11: Output verifikasi instalasi

4.2 Screenshot Hasil Test

Sisipkan screenshot atau gambar hasil dari:

- Terminal/command prompt yang menunjukkan environment aktif
- Output dari script test audio (sine wave plot)



Figure 9: Menunjukkan environment aktif

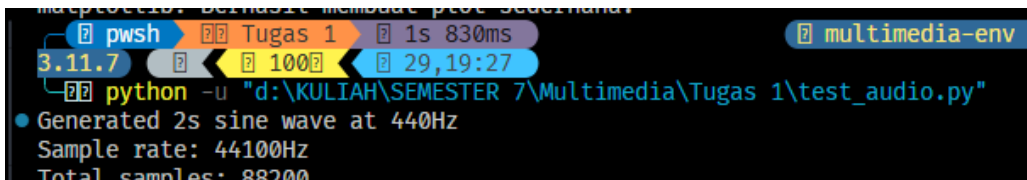


Figure 10: Output dari script test audio (sine wave plot)

- Output dari script test image (RGB stripes dengan circle)

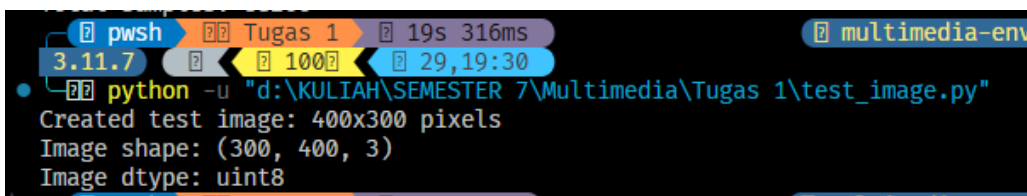


Figure 11: Output dari script test image (RGB stripes dengan circle)

4.3 Analisis dan Refleksi

Jawab pertanyaan berikut:

1. Mengapa penting menggunakan environment terpisah untuk project multimedia?

Karena dengan menggunakan environment yang terpisah, kita dapat mengelola dependensi dan versi library secara lebih efektif tanpa mempengaruhi proyek lain.

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

conda adalah package manager yang dapat mengelola paket dan environment untuk berbagai bahasa pemrograman, termasuk Python. venv adalah modul bawaan Python untuk membuat virtual environment dan lebih ringan dibandingkan dengan conda, sedangkan uv (Universal Virtualenv) adalah alat yang lebih baru dan lebih fleksibel untuk mengelola virtual environment, uv juga lebih cepat dalam pembuatan environmentnya. Saya memilih venv karena sudah terintegrasi dengan Python dan cukup untuk kebutuhan saya.

3. Library mana yang paling sulit diinstall dan mengapa?

Library yang sulit di install ketika setup adalah opencv-python karena hash yang tidak cocok dengan versi yang tersedia di repository. Solusi yang saya lakukan adalah update pip dan mencoba menginstall ulang library tersebut.

4. Bagaimana cara mengatasi masalah dependency conflict jika terjadi?

Memastikan proyek berjalan di environment yang sesuai, melakukan update pada library yang bermasalah, dan jika perlu, mengubah versi library yang digunakan. Dapat juga menggunakan pipdeptree untuk menganalisis dependensi dan mencari solusi.

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

numpy digunakan untuk komputasi numerik dan manipulasi array, matplotlib untuk visualisasi data, opencv-python untuk pemrosesan citra dan video, scikit-learn untuk machine learning dan analisis data, librosa untuk analisis audio, soundfile untuk membaca dan menulis file audio, scipy untuk komputasi ilmiah, pillow untuk manipulasi gambar, dan scikit-image untuk pemrosesan citra.

4.4 Troubleshooting

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

- **Masalah 1:** *[Deskripsi masalah]*
Solusi: *[Cara mengatasi]*
- **Masalah 2:** *[Deskripsi masalah]*
Solusi: *[Cara mengatasi]*

5 Export Environment untuk Reproduksi

Sebagai langkah terakhir, export environment Anda agar dapat direproduksi:

5.1 Untuk venv/uv

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

Kode 12: Export pip requirements

Berikut adalah isi file requirements.txt:

```
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==1.3.3
18 cycler==0.12.1
19 debugpy==1.8.16
20 decorator==5.2.1
21 defusedxml==0.7.1
22 executing==2.2.0
23 fastjsonschema==2.21.2
24 fonttools==4.59.2
```

```
25 fqdn==1.5.1
26 h11==0.16.0
27 httpcore==1.0.9
28 httpx==0.28.1
29 idna==3.10
30 imageio==2.37.0
31 imageio-ffmpeg==0.6.0
32 ipykernel==6.30.1
33 ipython==9.4.0
34 ipython-pygments-lexers==1.1.1
35 ipywidgets==8.1.7
36 isoduration==20.11.0
37 jedi==0.19.2
38 Jinja2==3.1.6
39 joblib==1.5.2
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-console==6.6.3
46 jupyter-events==0.12.0
47 jupyter-lsp==2.3.0
48 jupyter_client==8.6.3
49 jupyter_core==5.8.1
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==3.1.4
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==3.5
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==4.4.0
84 pooch==1.8.2
85 proglog==0.1.12
86 prometheus_client==0.22.1
```

```
87 prompt_toolkit==3.0.52
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==0.27.1
107 scikit-image==0.25.2
108 scikit-learn==1.7.1
109 scipy==1.16.1
110 Send2Trash==1.8.3
111 six==1.17.0
112 sniffio==1.3.1
113 soundfile==0.13.1
114 soupsieve==2.8
115 soxr==0.5.0.post1
116 stack-data==0.6.3
117 terminado==0.18.1
118 threadpoolctl==3.6.0
119 tifffile==2025.8.28
120 tinycss2==1.4.0
121 tornado==6.5.2
122 tqdm==4.67.1
123 traitlets==5.14.3
124 types-python-dateutil==2.9.0.20250822
125 typing_extensions==4.15.0
126 tzdata==2025.2
127 uri-template==1.3.0
128 urllib3==2.5.0
129 wcwidth==0.2.13
130 webcolors==24.11.1
131 webencodings==0.5.1
132 websocket-client==1.8.0
133 widgetsnbextension==4.0.14
```

Kode 13: Environment/Requirements file

6 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

Dalam pengerjaan setup Python environment untuk multimedia, saya belajar betapa pentingnya manajemen environment untuk menghindari konflik antar library dan memastikan reproducibility project. Dengan menggunakan venv, saya dapat mengelola dependensi setiap project dengan lebih efisien, yang sangat membantu dalam pengembangan proyek. Apapun jenis environment yang digunakan penting untuk selalu memeriksa dan mengelola dependensi dengan baik.

7 Referensi

Sertakan referensi yang Anda gunakan selama proses setup dan troubleshooting.

References

- [1] <https://stackoverflow.com/questions/50861082/how-to-print-environment-variables-to-the-console-in-powershell>
- [2] <https://medium.com/codefile/uv-or-conda-for-virtual-environments-7372a258c7d5>