



---

Nama: **M.Arief Rahman Hakim (NIM: 122140083)** Tugas Ke: **Worksheet 1: Setup Python Environment untuk Multimedia**

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

Tanggal: August 28, 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  $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$

## 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  $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$  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
```

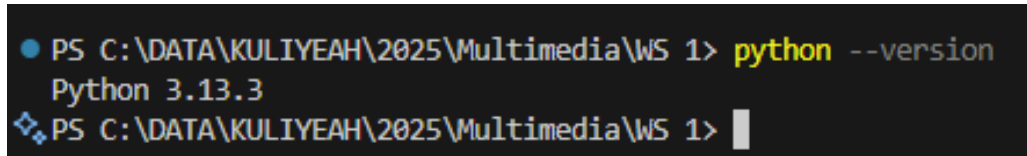
Kode 3: Membuat environment dengan uv

#### Dokumentasikan pengerjaan:

- Tool manajemen environment yang saya pilih:  
Pada penyelesaian worksheet 1 ini saya menggunakan uv sebagai environment karena uv ini lebih ringan dibandingkan conda dan venv. uv ini juga sudah include pip sehingga tidak perlu install pip lagi.

Langkah-langkah yang saya lakukan pada dasarnya sama seperti yang telah dijelaskan di atas, namun akan saya sertakan dokumentasi prosesnya pada bagian berikut.

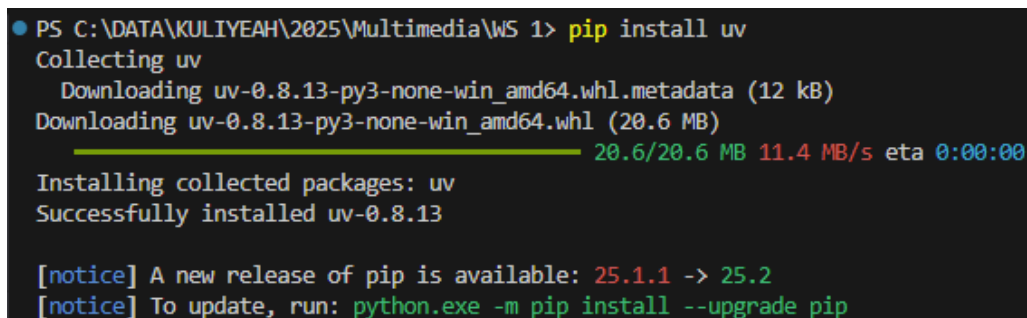
1. Memastikan Python sudah terinstall di perangkat. Pada tahap ini, saya menjalankan perintah `python --version` untuk mengecek apakah Python sudah terpasang dan memastikan versinya minimal 3.8.



```
PS C:\DATA\KULIYEAH\2025\Multimedia\WS 1> python --version
Python 3.13.3
PS C:\DATA\KULIYEAH\2025\Multimedia\WS 1>
```

Gambar 1: Cek instalasi dan versi Python

2. Menjalankan perintah `pip install uv` untuk memastikan bahwa tool uv sudah terpasang sebelum membuat environment baru.

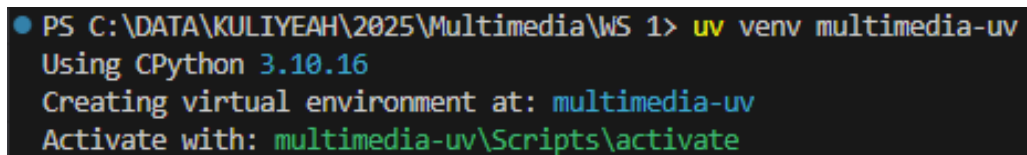


```
PS C:\DATA\KULIYEAH\2025\Multimedia\WS 1> pip install uv
Collecting uv
  Downloading uv-0.8.13-py3-none-win_amd64.whl.metadata (12 kB)
  Downloading uv-0.8.13-py3-none-win_amd64.whl (20.6 MB)
    ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 20.6/20.6 MB 11.4 MB/s eta 0:00:00
Installing collected packages: uv
Successfully installed uv-0.8.13

[notice] A new release of pip is available: 25.1.1 -> 25.2
[notice] To update, run: python.exe -m pip install --upgrade pip
```

Gambar 2: Cek instalasi uv

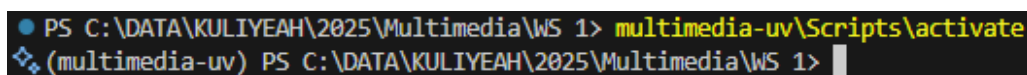
3. Membuat environment baru bernama `multimedia-uv` menggunakan perintah `uv venv multimedia-uv`.



```
PS C:\DATA\KULIYEAH\2025\Multimedia\WS 1> uv venv multimedia-uv
Using CPython 3.10.16
Creating virtual environment at: multimedia-uv
Activate with: multimedia-uv\Scripts\activate
```

Gambar 3: Membuat environment baru

4. Mengaktifkan environment yang sudah dibuat dengan perintah `multimedia-uv\Scripts\activate`. Setelah environment aktif, di terminal akan muncul tanda di sebelah kiri berupa nama environment (`(multimedia-uv)`) sebagai penanda bahwa environment sudah benar-benar aktif.



```
PS C:\DATA\KULIYEAH\2025\Multimedia\WS 1> multimedia-uv\Scripts\activate
(multimedia-uv) PS C:\DATA\KULIYEAH\2025\Multimedia\WS 1>
```

Gambar 4: Mengaktifkan environment

### 3.3 Bagian 2: Instalasi Library Multimedia

Setelah environment aktif, install library-library berikut:

### 3.3.1 Library Audio Processing

```
1 # Untuk conda:
2 conda install -c conda-forge librosa soundfile scipy
3
4 # Untuk pip (venv/uv):
5 pip install librosa soundfile scipy
```

Kode 4: Instalasi library audio

### 3.3.2 Library Image Processing

```
1 # Untuk conda:
2 conda install -c conda-forge opencv pillow scikit-image matplotlib
3
4 # Untuk pip (venv/uv):
5 pip install opencv-python pillow scikit-image matplotlib
```

Kode 5: Instalasi library image

### 3.3.3 Library Video Processing

```
1 # Untuk conda:
2 conda install -c conda-forge ffmpeg
3 pip install moviepy
4
5 # Untuk pip (venv/uv):
6 pip install moviepy
```

Kode 6: Instalasi library video

### 3.3.4 Library General Purpose

```
1 # Untuk conda:
2 conda install numpy pandas jupyter
3
4 # Untuk pip (venv/uv):
5 pip install numpy pandas jupyter
```

Kode 7: Instalasi library umum

#### Dokumentasikan di sini:

- Dikarenakan saya menggunakan environment jenis **uv**, maka langkah-langkah yang saya lakukan pada dasarnya sama seperti yang telah dijelaskan di atas, namun akan saya sertakan dokumentasi prosesnya pada bagian berikut.

#### 1. Menginstal Library Audio Processing dengan



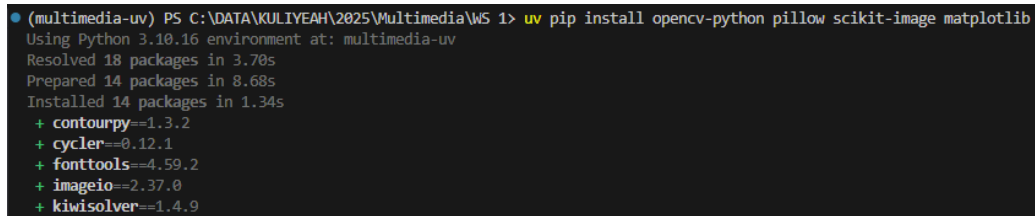
```
● (multimedia-uv) PS C:\DATA\KULIYEAH\2025\Multimedia\WS 1> uv pip install librosa soundfile scipy
Using Python 3.10.16 environment at: multimedia-uv
Resolved 25 packages in 798ms
Prepared 24 packages in 11.92s
Installed 25 packages in 2.65s
+ audioread==3.0.1
+ certifi==2025.8.3
+ cffi==1.17.1
+ charset-normalizer==3.4.3
+ decorator==5.2.1
+ idna==3.10
```

Gambar 5: Instalasi Library Audio Processing

Library yang berhasil diinstall dengan versinya adalah:

- librosa versi 0.11.0 - untuk pemrosesan audio dan musik
- soundfile versi 0.13.1 - untuk membaca dan menulis file audio
- scipy versi 1.15.3 - untuk algoritma komputasi ilmiah

## 2. Menginstal Library Image Processing

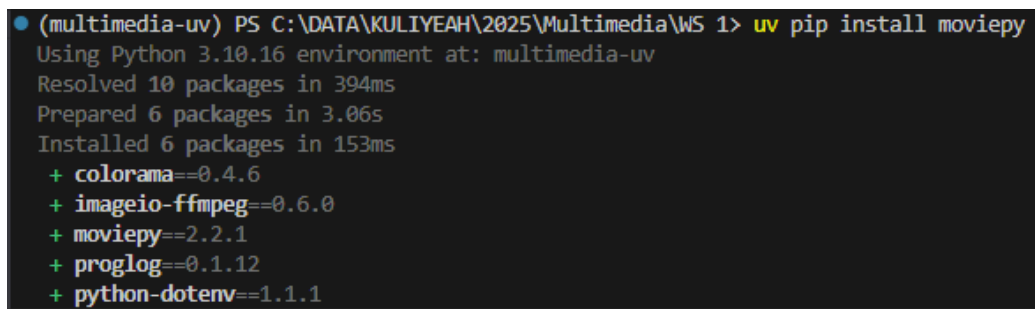


```
(multimedia-uv) PS C:\DATA\KULIYEAH\2025\Multimedia\WS 1> uv pip install opencv-python pillow scikit-image matplotlib
Using Python 3.10.16 environment at: multimedia-uv
Resolved 18 packages in 3.70s
Prepared 14 packages in 8.68s
Installed 14 packages in 1.34s
+ contourpy==1.3.2
+ cycler==0.12.1
+ fonttools==4.59.2
+ imageio==2.37.0
+ kiwisolver==1.4.9
```

Gambar 6: Instalasi Library Image Processing

- opencv-python versi 4.12.0.88 - untuk computer vision dan pemrosesan gambar
- pillow versi 11.3.0 - untuk manipulasi gambar dasar (Python Imaging Library)
- scikit-image versi 0.25.2 - untuk algoritma pemrosesan gambar tingkat lanjut
- matplotlib versi 3.10.5 - untuk visualisasi dan plotting

## 3. Menginstal Library Video Processing dengan perintah `pip install moviepy`.



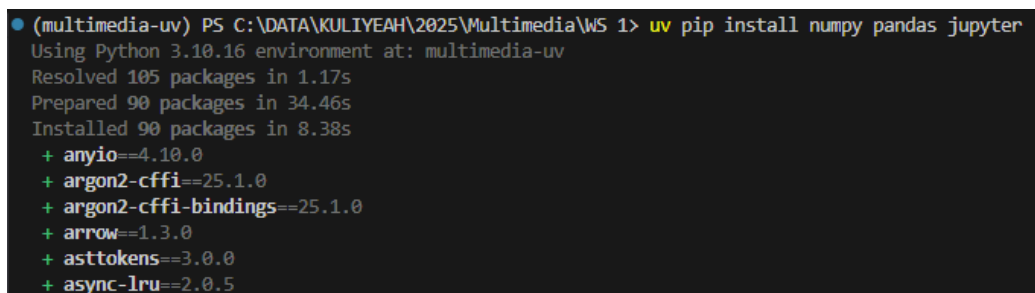
```
(multimedia-uv) PS C:\DATA\KULIYEAH\2025\Multimedia\WS 1> uv pip install moviepy
Using Python 3.10.16 environment at: multimedia-uv
Resolved 10 packages in 394ms
Prepared 6 packages in 3.06s
Installed 6 packages in 153ms
+ colorama==0.4.6
+ imageio-ffmpeg==0.6.0
+ moviepy==2.2.1
+ proglog==0.1.12
+ python-dotenv==1.1.1
```

Gambar 7: Instalasi Library Video Processing

Library yang berhasil diinstall dengan versinya adalah:

- moviepy versi 2.2.1 - untuk editing dan pemrosesan video

## 4. Menginstal Library General Purpose dengan perintah `pip install numpy pandas jupyter`.



```
(multimedia-uv) PS C:\DATA\KULIYEAH\2025\Multimedia\WS 1> uv pip install numpy pandas jupyter
Using Python 3.10.16 environment at: multimedia-uv
Resolved 105 packages in 1.17s
Prepared 90 packages in 34.46s
Installed 90 packages in 8.38s
+ 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
```

Gambar 8: Instalasi Library General Purpose

Library yang berhasil diinstall dengan versinya adalah:

- numpy versi 2.2.6 - untuk komputasi numerik dan array multidimensi
- pandas versi 2.3.2 - untuk manipulasi dan analisis data
- jupyter versi 1.1.1 - untuk interactive notebook environment

### 3.4 Bagian 3: Verifikasi Instalasi

Buat file Python sederhana untuk menguji semua library yang telah diinstall:  
**Jalankan script dan dokumentasikan hasilnya:**

### 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 8: 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}")

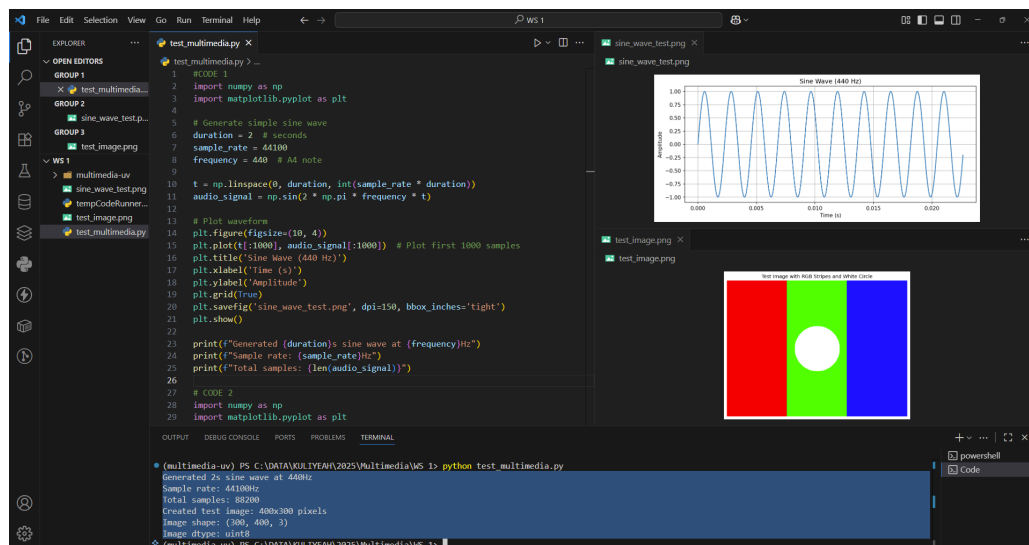
```

Kode 9: Test image processing sederhana

**Dokumentasikan hasil eksekusi:**

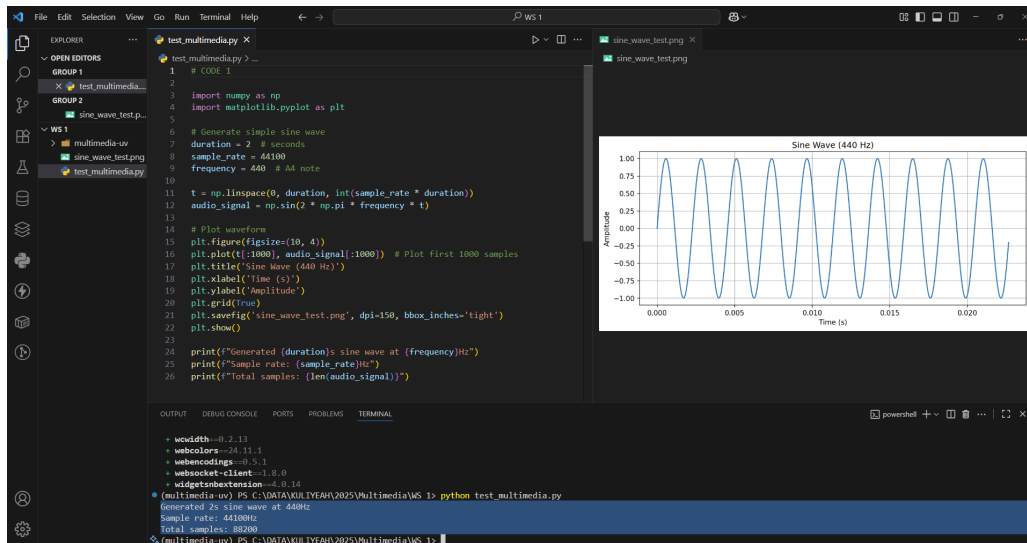
- Screenshot output dari kedua script di atas
- Gambar yang dihasilkan (sine\_wave\_test.png dan test\_image.png)
- Error message jika ada dan cara mengatasinya

## 1. Screenshot output dari kedua script di atas

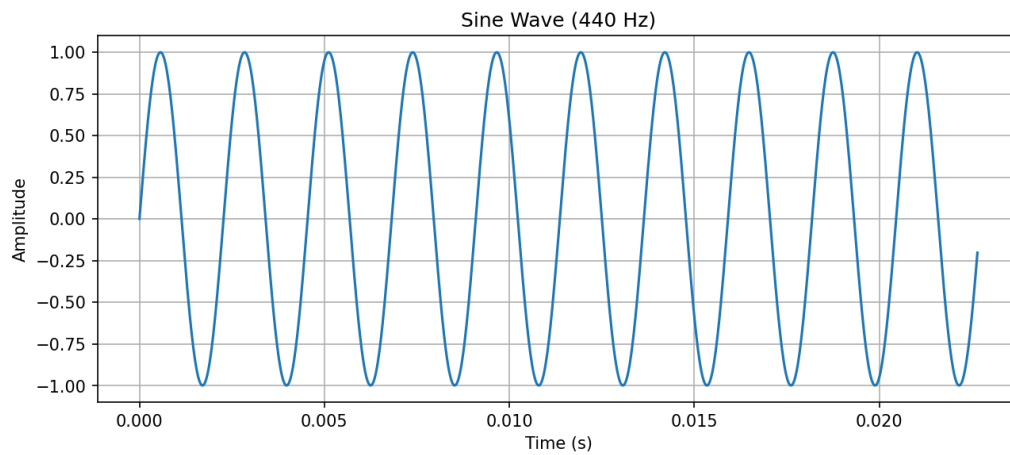


Gambar 9: Output dari 2 code

## 2. Screenshot code 1 dan hasilnya



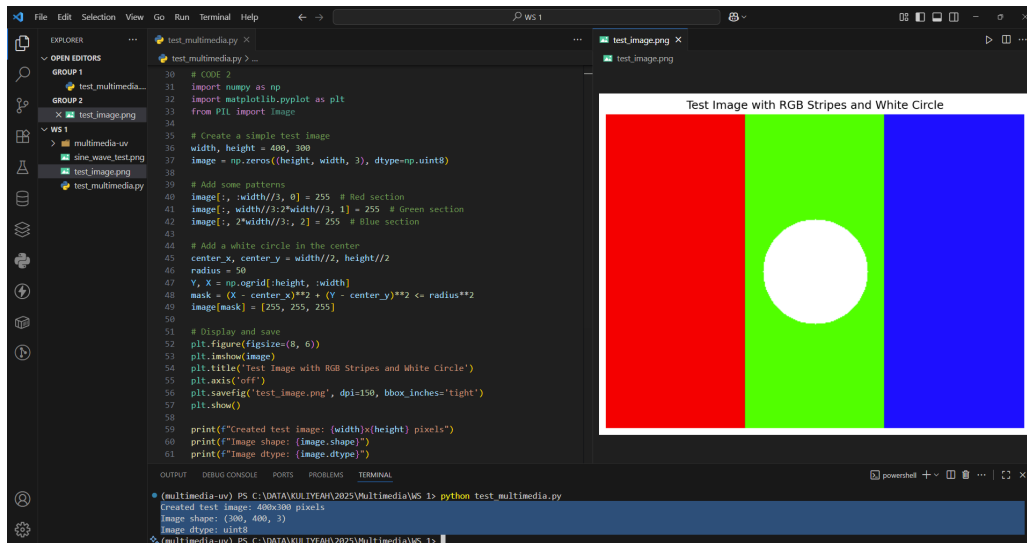
Gambar 10: Output Code 1



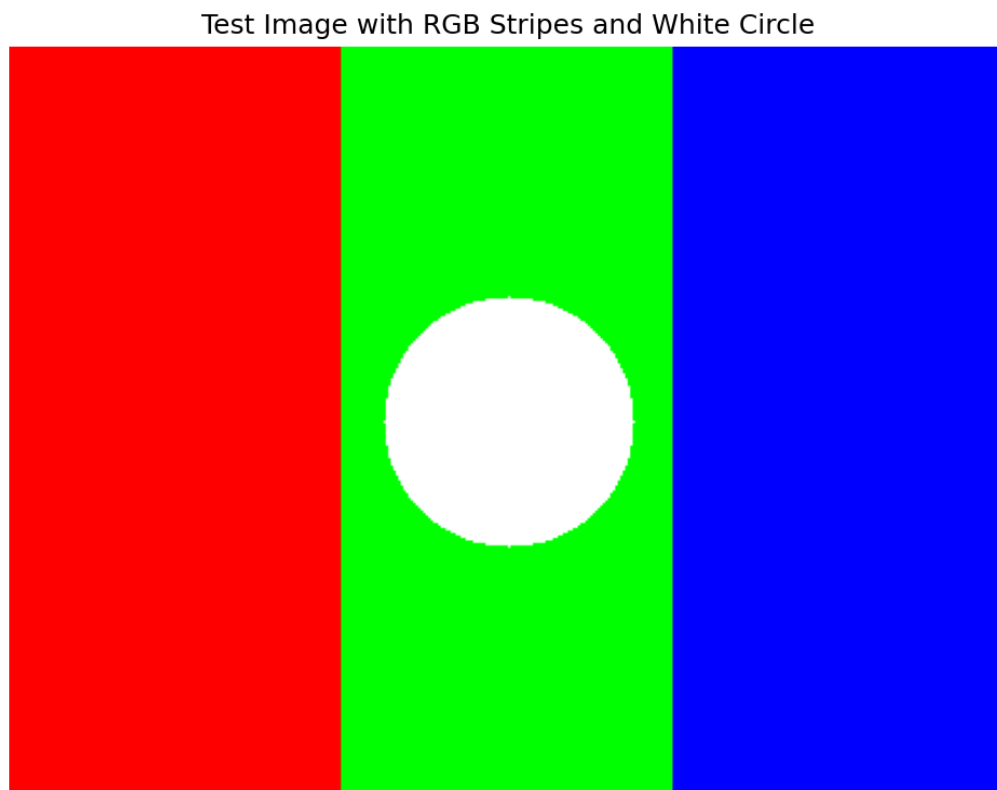
Gambar 11: Sine Wave (440 Hz)

- Generated 2s sine wave at 440Hz
  - Sample rate: 44100Hz
  - Total samples: 88200
3. Screenshot code 2 dan hasilnya





Gambar 12: Output Code 2



Gambar 13: Test Image with RGB Stripes and White Circle

- Created test image: 400x300 pixels
- Image shape: (300, 400, 3)
- Image dtype: uint8

## 4 Bagian Laporan

### 4.1 Output Verifikasi Instalasi

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

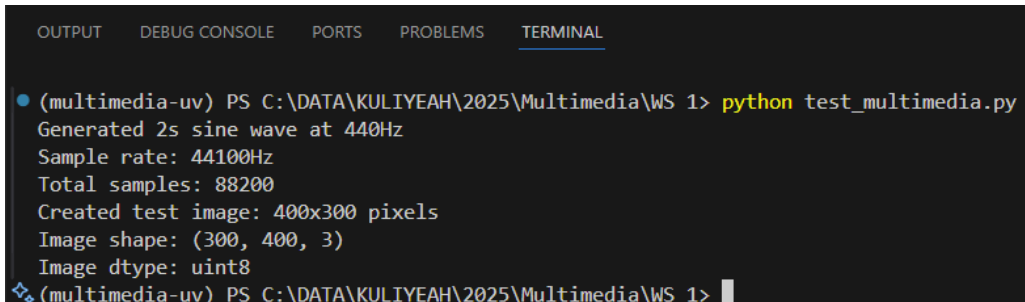
```
1 Generated 2s sine wave at 440Hz
2 Sample rate: 44100Hz
3 Total samples: 88200
4 Created test image: 400x300 pixels
5 Image shape: (300, 400, 3)
6 Image dtype: uint8
```

Kode 10: Output verifikasi instalasi

### 4.2 Screenshot Hasil Test

Sisipkan screenshot atau gambar hasil dari:

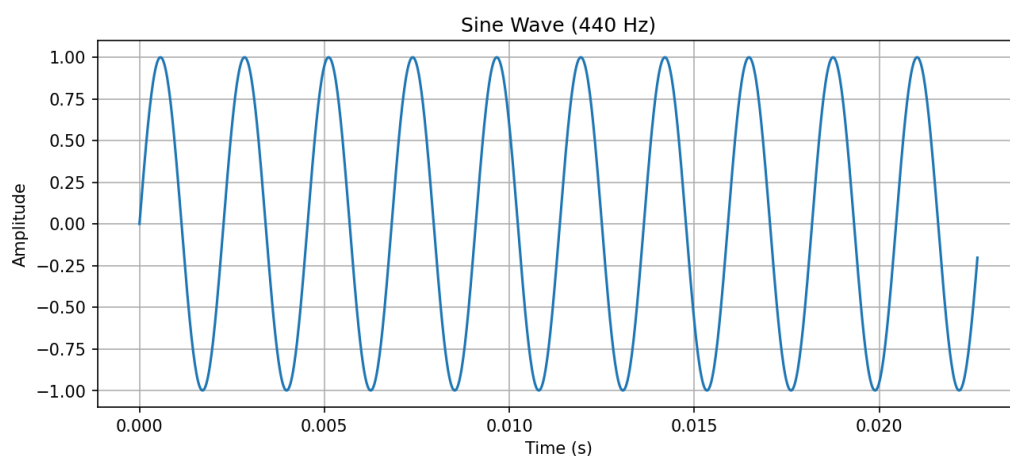
- Terminal/command prompt yang menunjukkan environment aktif



```
OUTPUT  DEBUG CONSOLE  PORTS  PROBLEMS  TERMINAL
(multimedia-uv) PS C:\DATA\KULIYEAH\2025\Multimedia\WS 1> 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) PS C:\DATA\KULIYEAH\2025\Multimedia\WS 1>
```

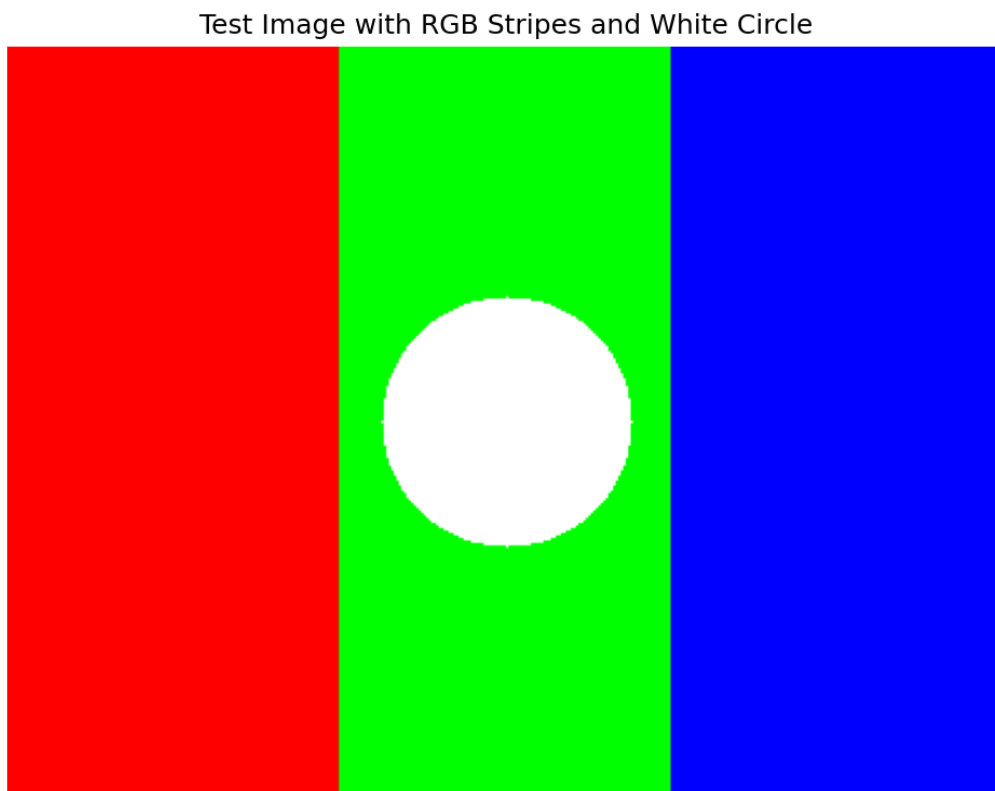
Gambar 14: Environment Aktif

- Output dari script test audio (sine wave plot)



Gambar 15: Sine Wave (440 Hz)

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



Gambar 16: Test Image with RGB Stripes and White Circle

*Gunakan perintah `\includegraphics` untuk menyisipkan gambar*

### 4.3 Analisis dan Refleksi

**Jawab pertanyaan berikut:**

**1. Mengapa penting menggunakan environment terpisah untuk project multimedia?**

*Berdasarkan Pengetahuan Pribadi, menggunakan environment terpisah sangat penting untuk menghindari konflik antara berbagai library dan dependensi yang mungkin dibutuhkan oleh project multimedia. Oleh karena itu dengan menggunakan environment terpisah, kita dapat memastikan bahwa setiap project memiliki konfigurasi yang sesuai tanpa mempengaruhi project lain bukan hanya multimedia tetapi semua project yang ada di dalam perangkat kita.*

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

*Berdasarkan Perkuliahan PSD dan Perkuliahan mg1 Multimedia, Pak Martin mengatakan bahwa perbedaan utama antara conda, venv, dan uv adalah Manajemen dependency dan Instalasinya, semuanya punya kelebihan dan kekurangan masing-masing. Saya sendiri menggunakan UV dikarenakan tidak perlu menginstal aplikasi ke 3 seperti anaconda atau miniconda dan memiliki performa yang lebih baik dalam proses menginstal package. Meskipun begitu UV memiliki kekurangan dalam hal fleksibilitas, berbeda dengan env conda yang bisa digunakan untuk berbagai project yang berbeda.*

**3. Library mana yang paling sulit diinstall dan mengapa?**

Berdasarkan pengetahuan yang saya cari lewat AI, library yang paling sulit diinstall adalah **MoviePy** karena memiliki dependency pada FFmpeg yang tidak dapat diinstall secara otomatis melalui pip/uv. Beberapa alasan yang membuat MoviePy challenging:

- MoviePy membutuhkan FFmpeg executable yang harus diinstall secara manual di sistem operasi
- Dengan menggunakan uv/pip, FFmpeg tidak terinstal otomatis seperti di conda (conda install -c conda-forge ffmpeg)
- Sering terjadi error "FFmpeg not found" saat mencoba memproses video karena PATH configuration
- Di Windows perlu download manual FFmpeg binary dan setting environment variable
- Codec compatibility issues antar platform (Windows/Linux/MacOS)

Library kedua yang cukup challenging adalah **OpenCV** karena versi pip (opencv-python) memiliki fitur terbatas dibanding versi conda. OpenCV full features membutuhkan kompilasi dari source dengan berbagai system dependencies seperti CMake, kompiler C++, dan codec libraries.

#### 4. Bagaimana cara mengatasi masalah dependency conflict jika terjadi?

Dependency conflict terjadi ketika dua atau lebih library membutuhkan versi berbeda dari library yang sama. Berikut contoh dan cara mengatasinya:

##### Contoh Dependency Conflict:

- Library A membutuhkan numpy $\geq$ 1.20.0, sementara Library B membutuhkan numpy $<$ 1.19.0
- TensorFlow versi lama tidak kompatibel dengan OpenCV versi terbaru
- Matplotlib versi berbeda dengan requirements scikit-image

##### Cara Mengatasi:

- (a) **Gunakan Environment Terpisah** - Buat environment berbeda untuk project yang memiliki requirements berbeda
- (b) **Pinning Version** - Tentukan versi spesifik di requirements.txt (contoh: numpy==1.21.0)
- (c) **Update Secara Bertahap** - Update satu library dalam satu waktu dan test compatibility
- (d) **Gunakan pip-tools** - Tools untuk resolver dependency yang lebih baik
- (e) **Konsultasi Documentation** - Cek compatibility matrix dari library official

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

Berikut adalah penjelasan fungsi dari setiap library yang telah diinstall, dikelompokkan berdasarkan kategori multimedia processing:

##### Audio Processing Libraries:

- **librosa** versi 0.11.0 - Library utama untuk analisis dan pemrosesan audio/musik, menyediakan fitur spektrogram, MFCC, beat tracking, pitch detection, dan audio feature extraction
- **soundfile** versi 0.13.1 - Library untuk membaca dan menulis berbagai format file audio (WAV, FLAC, OGG) dengan interface sederhana untuk I/O audio
- **scipy** versi 1.15.3 - Menyediakan algoritma signal processing untuk audio seperti filtering, FFT, convolution, dan window functions

##### Image Processing Libraries:

- *opencv-python versi 4.12.0.88* - Library computer vision terlengkap untuk pemrosesan gambar dengan fitur *filtering, edge detection, object detection, dan face recognition*
- *pillow versi 11.3.0* - *Python Imaging Library* untuk manipulasi gambar dasar seperti *resize, crop, rotate, dan format conversion*
- *scikit-image versi 0.25.2* - Library pemrosesan gambar tingkat lanjut dengan algoritma ilmiah untuk *segmentation, morphology, dan feature detection*
- *matplotlib versi 3.10.5* - Library visualisasi untuk *plotting dan display gambar, essential untuk analisis dan presentasi data visual*

#### Video Processing Libraries:

- *moviepy versi 2.2.1* - Library utama untuk *editing dan pemrosesan video* dengan fitur *video cutting, concatenation, effects, dan audio-video synchronization*

#### General Purpose Libraries:

- *numpy versi 2.2.6* - Foundation library untuk *komputasi numerik dan array multidimensi, dasar untuk semua library multimedia*
- *pandas versi 2.3.2* - Library untuk *manipulasi dan analisis data terstruktur, berguna untuk metadata analysis dan dataset management*
- *jupyter versi 1.1.1* - *Interactive notebook environment untuk eksperimen multimedia, ideal untuk prototyping dan pembelajaran*

## 4.4 Troubleshooting

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

- **Masalah 1:** *pdfTex error (font expansion): auto expansion is only possible with scalable fonts*  
**Solusi:** Menambahkan `\usepackage{lmodern}` di bagian atas (*preamble*) file *latex*.
- **Masalah 2:** *UV Tidak bisa membaca package*  
**Solusi:** Mengingat *wkwkwk*, jadi kalau kita gunain *UV* itu harus ada kata *uv* sebelum *pip install*

## 5 Export Environment untuk Reproduksi

Sebagai langkah terakhir, export environment Anda agar dapat direproduksi:

### 5.1 Untuk Conda

```
1 conda env export > environment.yml
```

Kode 11: Export conda environment

### 5.2 Untuk venv/uv

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

Kode 12: 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==4.59.2
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==1.5.2
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-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==2.3.0
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==4.4.0
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==1.15.3
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==2.8
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
```

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

*Bukan Pengalaman baru, tapi salah satu langkah yang baik untuk mengingat kembali proses setup yang benar dan memastikan semua library yang diperlukan sudah terinstall dengan benar. Dengan environment yang terkelola dengan baik, saya merasa lebih siap untuk memulai project multimedia selanjutnya tanpa khawatir tentang konflik dependency atau masalah kompatibilitas. Saran untuk mahasiswa lain yang akan melakukan setup serupa adalah untuk selalu mencatat langkah-langkah yang diambil dan masalah yang dihadapi, sehingga dapat menjadi referensi di masa depan.*



## 7 Referensi

Sertakan referensi yang Anda gunakan selama proses setup dan troubleshooting.

1. Manullang, Martin. "Materi Perkuliahan Minggu 1 - Sistem Teknologi Multimedia (IF25-40305)." Institut Teknologi Sumatera, 2025.
2. "Percakapan dengan AI mengenai Python Environment Management dan Multimedia Libraries." Link Drive:  
[https://drive.google.com/drive/folders/1il2jW19MLZKb2l\\_rmeYMt9j7QniIKAVd?usp=drive\\_link](https://drive.google.com/drive/folders/1il2jW19MLZKb2l_rmeYMt9j7QniIKAVd?usp=drive_link)