Dockerizing python app

Python scripts/apps can be dockerised rather than using a venv and/or installing in the host system.

For example, ffpb is a wrapper for ffmpeg to show progress. Using pipx comes with a lot of dependencies, a standalone package is not available, so dockerizing it is the simplest option and bundling it with the latest statically linked ffmpeg binaries.

Startup code - which is the content of the ffpb script in /usr/bin when installed:

main.py

```
#!/bin/python
import sys
from ffpb import main
if __name__ == '__main__':
    if sys.argv[0].endswith('.exe'):
        sys.argv[0] = sys.argv[0][:-4]
    sys.exit(main())
```

This Dockerfile uses the slim Python 3.9 docker image, adds the main.py

Dockerfile

```
FROM python:3.9-slim
ADD main.py .
COPY ffmpeg-n8.0-latest-linux64-lgpl-8.0/bin/* /usr/bin
#RUN pip install ffpb && wget
https://github.com/BtbN/FFmpeg-Builds/releases/download/latest/ffmpeg-n
8.0-latest-linux64-lgpl-8.0.tar.xz && tar -xvf ffmpeg-n8.0-latest-
linux64-lgpl-8.0.tar.xz && mv ffmpeg-n8.0-latest-linux64-lgpl-8.0/bin/*
/usr/bin/
RUN pip install ffpb
ENTRYPOINT ["python", "main.py"]
```

```
docker build -t python-ffpb .
```

It can just be called using this command which could be wrapped into a shell wrapper or alias for convenience. It supports command line options as well:

```
docker run python-ffpb
```