Migration

From pip to a uv project

Init project on `temp-project`

```
mkdir lesson4\temp-project
cd lesson4\temp-project
uv init
```

Create new file `requirements.in`

```
# on requirements.in
fastapi
pydantic>2
```

These dependencies can be compiled into a `requirements.txt`

```
uv pip compile requirements.in -o requirements.txt
# You can see all the versions constrains are exact.
```

better then `pip freeze > requirements.txt`

Platform-specific dependencies

Create new file `requirements.in`

```
# on requirements.in
tqdm
```

Use `uv compile --universal`

```
uv pip compile --universal requirements.in -o requirements.txt
```

you can see the `colorama` Which is a Windows-only dependency of `tqdm`

Migrating to a uv project

The `pyproject.toml` is a standardized file for Python project metadata. It replaces `requirements.in` files, allowing you to represent arbitrary groups of project dependencies.

It also provides a centralized location for metadata about your project, such as the build system or tool settings.

For example, the `requirements.in` and `requirements-dev.in` files above can be translated to a `pyproject.toml` as follows:

```toml
[project]
name = "example"
version = "0.0.1"
dependencies = [
    "fastapi",
    "pydantic>2"
]

[dependency-groups]
dev = ["pytest"]
```

The uv lockfile

uv uses a lockfile (`uv.lock`) file to lock package versions. The format of this file is specific to uv, allowing uv to support advanced features. It replaces `requirements.txt` files.

```
uv add -r requirements.in
# or better way
uv pip compile requirements.in -o reuiqrements-win.txt \
--python-platform windows \
--no-strip-markers
```

Once each `requirements.txt` file has been transformed, the dependencies can be imported to the `pyproject.toml` and `uv.lock` with `uv add`:

```
uv add -r requirements-win.txt
```

When you re-create virtual env or clone sources, you can easily sync package as a `uv sync`

```
# delete original .venv directory
# and re-create .venv
uv venv -p 3.12 .venv

# lib directory has no packages
# but you have uv.lock file
uv sync
# you can see the packages in lib diretory
```