Testing Dependencies with Overrides

Overriding dependencies during testing

There are some scenarios where you might want to override a dependency during testing.

You don't want the original dependency to run (nor any of the sub-dependencies it might have).

Instead, you want to provide a different dependency that will be used only during tests (possibly only some specific tests), and will provide a value that can be used where the value of the original dependency was used.

Use cases: external service

An example could be that you have an external authentication provider that you need to call.

You send it a token and it returns an authenticated user.

This provider might be charging you per request, and calling it might take some extra time than if you had a fixed mock user for tests.

You probably want to test the external provider once, but not necessarily call it for every test that runs.

In this case, you can override the dependency that calls that provider, and use a custom dependency that returns a mock user, only for your tests.

Use the `app.dependency_overrides` attribute

For these cases, your **FastAPI** application has an attribute `app.dependency_overrides`, it is a simple `dict`.

To override a dependency for testing, you put as a key the original dependency (a function), and as the value, your dependency override (another function).

And then **FastAPI** will call that override instead of the original dependency.

```
{!../../../docs_src/dependency_testing/tutorial001.py!}
```

!!! tip You can set a dependency override for a dependency used anywhere in your **FastAPI** application.

```
The original dependency could be used in a *path operation function*, a *path
operation decorator* (when you don't use the return value), a `.include_router()`
call, etc.

FastAPI will still be able to override it.
```

Then you can reset your overrides (remove them) by setting `app.dependency_overrides` to be an empty `dict`:

```
app.dependency_overrides = {}
```

!!! tip If you want to override a dependency only during some tests, you can set the override at the beginning of the test (inside the test function) and reset it at the end (at the end of the test function).