GET 라우트인 /event와 /event/{id}, POST 라우트인 /event/new를 모두 테스트했다. 이어서 변경 및 삭제 라우트 테스트를 작성해보자.

변경 라우트 테스트

변경 라우트 테스트 함수를 작성해보자.

```
@pytest.mark.asyncio
async def test_update_event(default_client: httpx.AsyncClient, mock_event: Event,
access_token: str) -> None:
    test_payload = {
        "title": "Updated FastAPI event"
    }

    headers = {
        "Content-Type": "application/json",
        "Authorization": f"Bearer {access_token}"
    }

    url = f"/event/{str(mock_event.id)}"

    response = await default_client.put(url, json=test_payload, headers=headers)

    assert response.status_code == 200
    assert response.json()["title"] == test_payload["title"]
```

이 코드는 mock_event 픽스처에서 추출한 ID를 사용해 데이터베이스에 저장된 해당 이벤트를 수정한다. 그런 다음 요청 페이로드와 헤더를 정의하고 response 변수에 요청 결과를 저장한다. 마지막으로 이 결과가 예상한 값과 일치하는지 확인한다. 다음 명령을 사용해 테스트를 실행해보자.

```
(venv)$ pytest tests/test_routes.py
```

실행 결과는 [그림 8-10]과 같다.