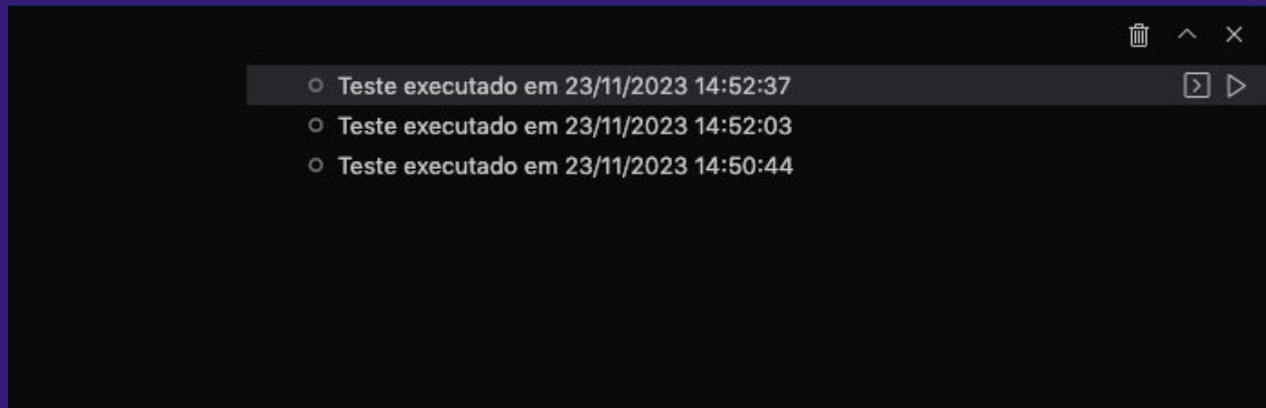


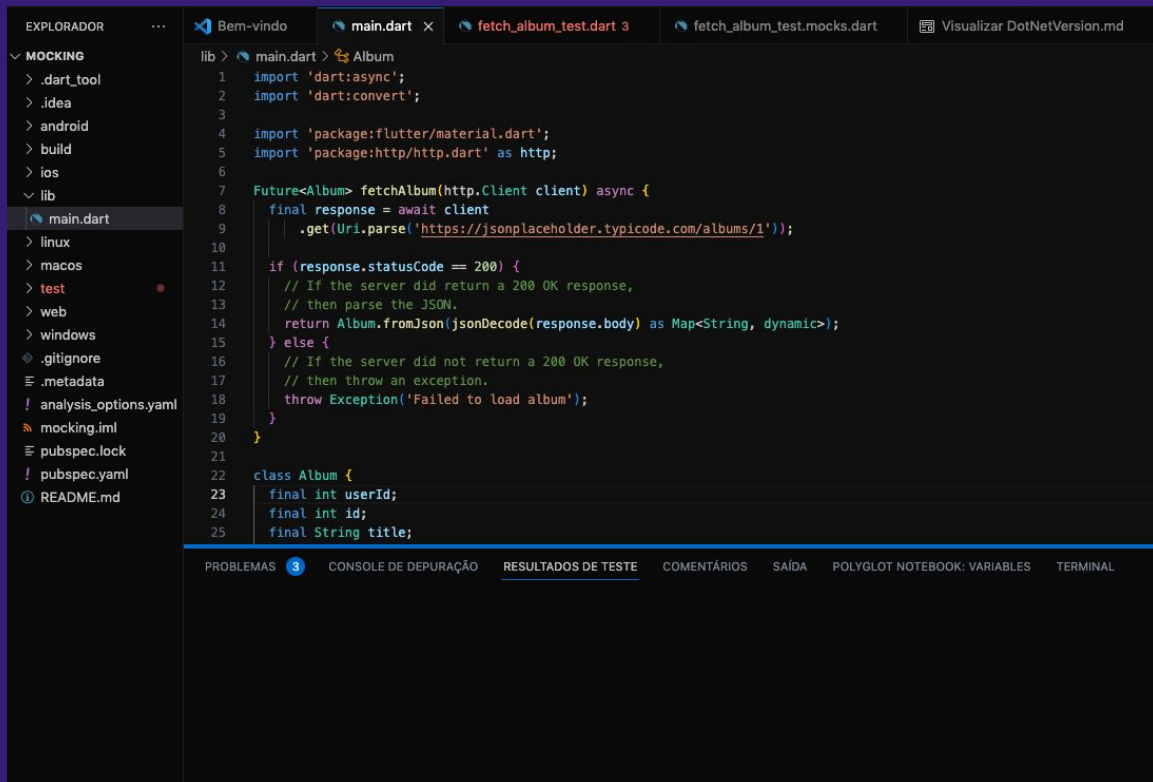
Implementar o código definido em <https://docs.flutter.dev/cookbook/testing/unit/mocking> que usa a dependência Mockito.
Evidenciar a execução do teste e o código persistido no GitHub.



Avaliação 09

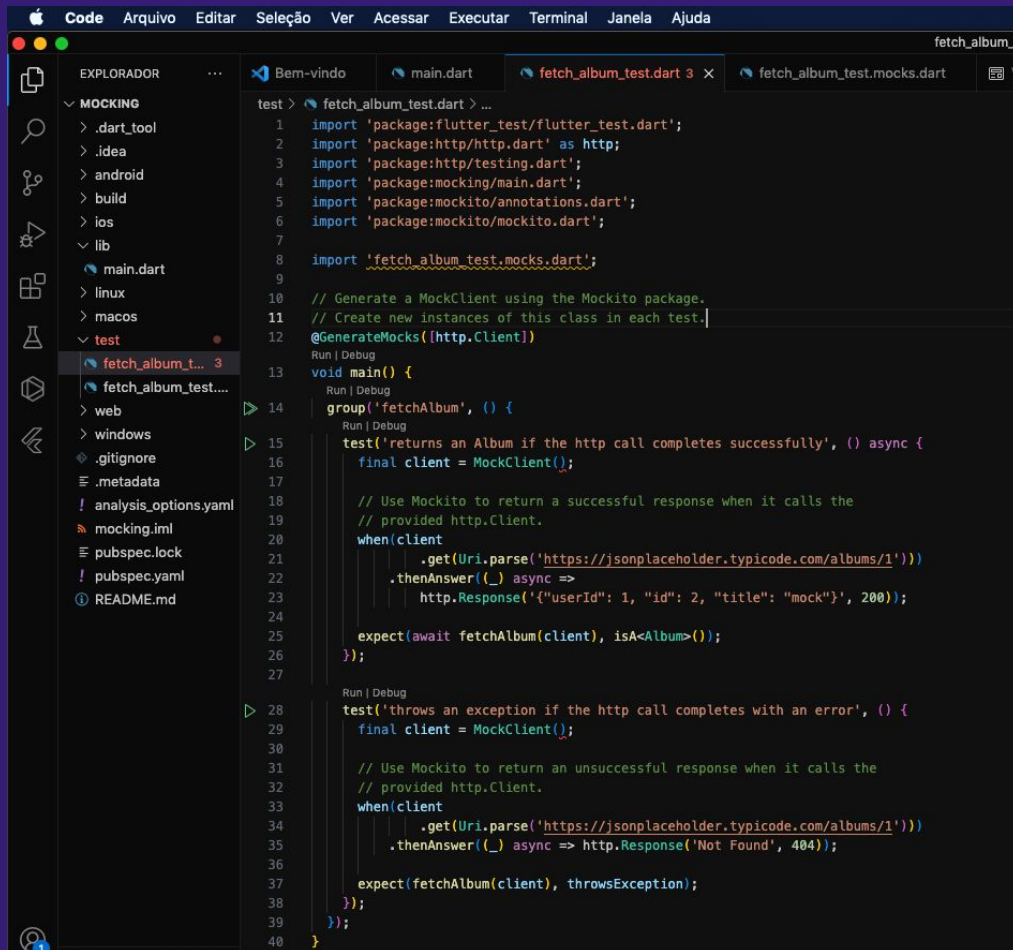
mockito

Aluna: Larissa dos Santos Holanda



Avaliação 09 mockito

Aluna: Larissa dos Santos Holanda



The screenshot shows an IDE window with the following components:

- Explorer (Left):** A file tree showing a project structure with folders like `.dart_tool`, `.idea`, `android`, `build`, `ios`, `lib`, `main.dart`, `linux`, `macos`, `test`, `web`, `windows`, and files like `.gitignore`, `.metadata`, `analysis_options.yaml`, `mocking.iml`, `pubspec.lock`, `pubspec.yaml`, and `README.md`. The `test` folder is expanded, showing `fetch_album_t...` and `fetch_album_test...`.
- Editor (Center):** The file `fetch_album_test.dart` is open. It contains Dart code for testing the `fetchAlbum` function using Mockito. The code includes imports for `flutter_test`, `http`, `http_testing`, `mocking`, `mockito_annotations`, `mockito`, and `fetch_album_test.mocks`. It defines a `main` function with two test cases: one for a successful response and one for a 404 error.
- Terminal (Bottom):** The terminal is empty, showing the prompt `test >`.

```
test > fetch_album_test.dart > ...
1 import 'package:flutter_test/flutter_test.dart';
2 import 'package:http/http.dart' as http;
3 import 'package:http/testing.dart';
4 import 'package:mocking/main.dart';
5 import 'package:mockito/annotations.dart';
6 import 'package:mockito/mockito.dart';
7
8 import 'fetch_album_test.mocks.dart';
9
10 // Generate a MockClient using the Mockito package.
11 // Create new instances of this class in each test.
12 @GenerateMocks([http.Client])
13 void main() {
14   group('fetchAlbum', () {
15     test('returns an Album if the http call completes successfully', () async {
16       final client = MockClient();
17
18       // Use Mockito to return a successful response when it calls the
19       // provided http.Client.
20       when(client
21         .get(Uri.parse('https://jsonplaceholder.typicode.com/albums/1')))
22         .thenReturn(
23           http.Response({'userId': 1, 'id': 2, 'title': 'mock'}, 200));
24
25       expect(await fetchAlbum(client), isA<Album>());
26     });
27
28     test('throws an exception if the http call completes with an error', () {
29       final client = MockClient();
30
31       // Use Mockito to return an unsuccessful response when it calls the
32       // provided http.Client.
33       when(client
34         .get(Uri.parse('https://jsonplaceholder.typicode.com/albums/1')))
35         .thenReturn(
36           http.Response('Not Found', 404));
37
38       expect(fetchAlbum(client), throwsException);
39     });
40   });
41 }
```

Avaliação 09 mockito

Aluna: Larissa dos Santos Holanda