

Effective Unit & Integration Testing

Pitfalls and How to Avoid Them

Atanas Gardev, Petyo Dimitrov

30-05-2024

Qinshift 

About us



Atanas Gardev

Software Architect

20 years' experience in IT, Java Backend focus

3 kids :)

Petyo Dimitrov

Senior Software Architect

17 years' experience in IT

no kids (yet)



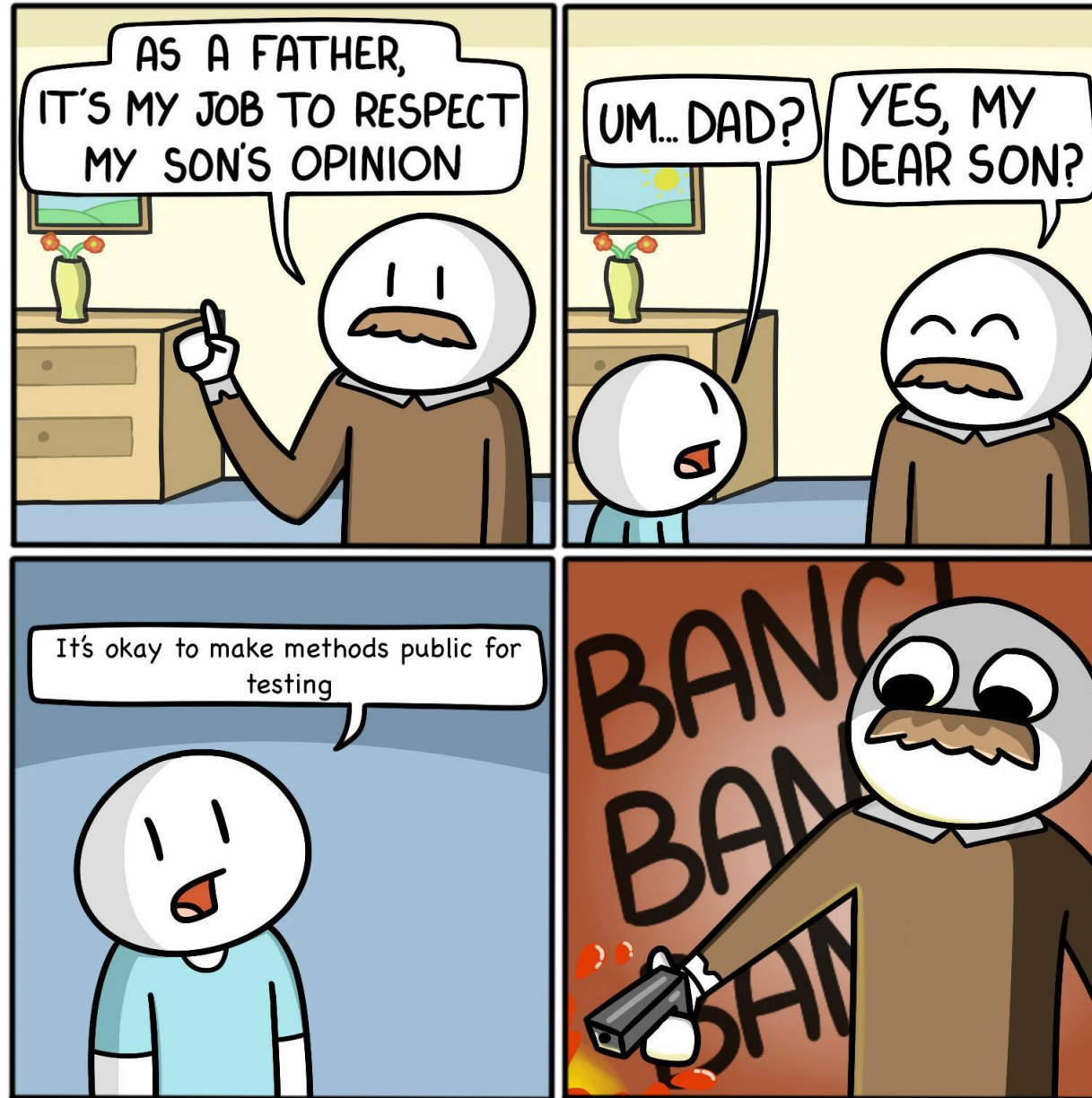
Agenda



- 01 What is a good test?
- 02 Anti-patterns
- 03 How to make a better test?
- 04 AI benefits in testing
- 05 Q&A

What is a
good test?

-FATHER-



Confidence

```
✓ core > infosystem > version > Execute two info
- core > infosystem > models > Execute two info
✓ core > infosystem > groupParameters > nest test with groupParameters
[2018-05-25T20:40:08.576Z - info] <SICON-GW> Neither EventTypesID, nor Type is set
[2018-05-25T20:40:08.581Z - info] <SICON-GW> Neither EventTypesID, nor Type is set
✓ core > infosystem > collection > parseFilter - Turn a string with equals into an array consumable by get
ms option
✓ core > infosystem > collection > parseFilter - undefined should return an empty array
✓ core > infosystem > collection > parseFilter - Keep array format if array
✓ core > infosystem > collection > parseFilter - multiple filter types
✓ core > infosystem > collection > Invalid limit should throw an error (114ms)
✓ core > infosystem > collection > offset out of range should throw an error
✓ core > infosystem > groupParameters > transform parameters into their complex groups, when they have subIn
exes (223ms)
✓ core > infosystem > groupParameters > group parameters by parameterGroup0-3 (217ms)
✓ core > infosystem > models > insert unique of a single event with new EventClass (191ms)
✓ core > infosystem > models > insert unique of an existing single event (185ms)
✓ core > infosystem > models > Get GUIDD by filename that does not exist will throw an error (178ms)
✓ core > infosystem > collection > Default limit should be 10 (378ms)
✓ core > infosystem > collection > Get a list of items with limit (400ms)
✓ core > infosystem > collection > Filter with filter option (378ms)
✓ core > infosystem > collection > Filter with filter option (411ms)
✓ core > infosystem > collection > Select only a few properties in one transaction (242ms)
✓ core > infosystem > collection > Execute two info system methods (400ms)
✓ core > infosystem > models > Execute two info system methods (400ms)
✓ core > infosystem > collection > group result by a property (485ms)
✓ core > infosystem > collection > group result by two property (504ms)
✓ core > infosystem > collection > group Devices by their Location (485ms)
✓ core > infosystem > collection > Setting limit as infinity or -1 should get all entries (504ms)
✓ core > infosystem > collection > Filter with equals operator should be the same with omitted (598ms)
✓ core > infosystem > collection > Filter with equals operator should be the same with omitted (598ms)
✓ core > infosystem > models > Execute two transactions simultaneously (693ms)
✓ core > infosystem > models > Count adapter for a set of properties (739ms)
✓ core > infosystem > models > Update adapter by QRCode (770ms)
✓ core > infosystem > models > Count devices for a set of properties (811ms)
✓ core > infosystem > models > Connect device by setting ConnectionState (856ms)
✓ core > infosystem > models > Disconnect device by setting ConnectionState (856ms)
✓ core > infosystem > models > Insert new DeviceControl (894ms)
✓ core > infosystem > models > Insert new DeviceFile (965ms)
✓ core > infosystem > models > Update existing DeviceFile (1s)
✓ core > infosystem > models > Insert new Event to EPC (1.1s)
✓ core > infosystem > models > Insert new EPCConfig (1.1s)
✓ core > infosystem > models > Insert an Error in ErrorMapper (1.2s)
✓ core > infosystem > models > Disappeared Event is no longer active (1.3s)
✓ core > infosystem > models > update a single event (1.3s)
✓ core > infosystem > models > Update Value for ISDU Diagnosis entry identified by DevID, Index, St
3s)
✓ core > infosystem > models > Insert into ISDU Diagnosis (1.3s)
✓ core > infosystem > models > Insert new ProcessData (1.3s)
✓ core > infosystem > models > insert of a process data mode (1.4s)
✓ core > infosystem > models > Update vendor by ID (1.4s)
✓ core > infosystem > models > Get adapter by id (1.7s)
✓ core > infosystem > models > Get selected properties of an adapter by id (1.7s)
✓ core > infosystem > models > Get selected properties of an adapter by filter (1.7s)
✓ core > infosystem > models > Insert unique existing datatype (1.8s)
✓ core > infosystem > models > Get selected properties of a device by id (1.8s)
```





Quick feedback



Robust



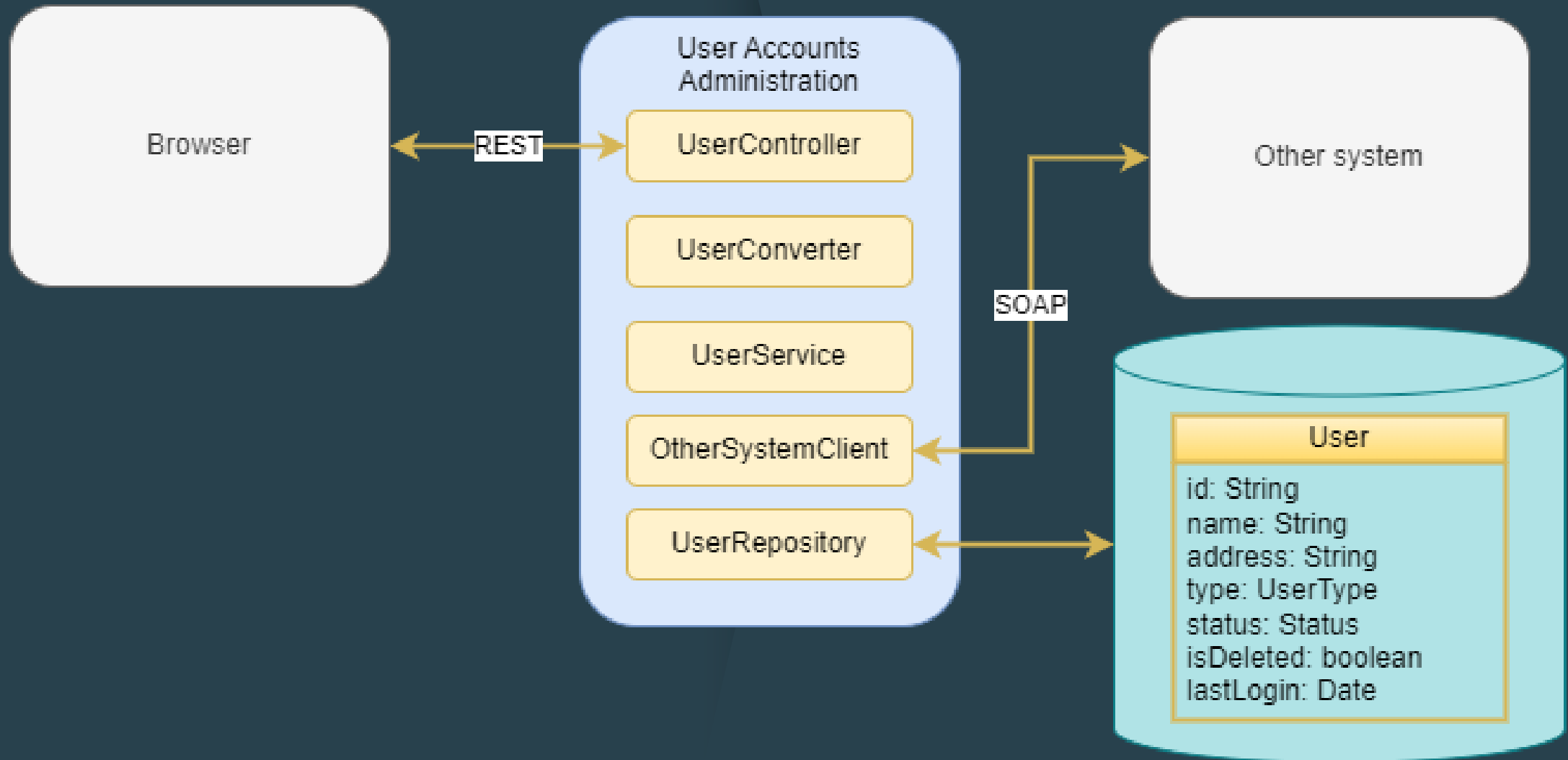


Maintainable



Anti-patterns & fixes

Context



Spock structure



```
def "test method name" () {  
  given: "prepare"  
  ...  
  when: "stimulus"  
  ...  
  then: "assertions"  
  ...  
}
```



Spock syntax



```
def user = Mock(User)

user.name >> "Johnny Bravo"

user.isDeleted == false

1 * converter.convert(user)

0 * converter._
```



Anti-pattern #1



```
def user = Mock(User)
```

```
user.name >> "Johnny Bravo"
```

```
user.address >> "Sofia"
```

```
user.type >> UserType.Corporate
```

```
user.status >> UserStatus.Active
```

```
user.isDeleted >> false
```

```
user.lastLogin >> new Date()
```

Mocking overuse - solution



(1)

```
def johnny = new User("Johnny Bravo",  
                    "Sofia", UserType.Active)  
johnny.isDeleted = false  
johnny.lastLogin = new Date()
```

(2)

```
def johnny = UserMother.johnny()
```

Object Mother pattern



```
UserMother.johnny()  
UserMother.littleSuzy()  
AddressMother.sofia()
```

```
UserMother.johnny()  
    .withAddress(  
        AddressMother.sofia().build()  
    ).build()
```



Anti-pattern #2



```
def "user conversion works" () {  
  given:  
    def input = new User("Johnny Bravo", "Sofia", Active)  
  
  when:  
    def result = underTest.convert(input)  
  
  then:  
    result.name == "Johnny Bravo"  
    result.address == "Sofia"  
    result.type == Active  
}
```

Repeating yourself – solution



```
def "user conversion works" () {  
  given:  
    def input = UserMother.johnny()  
  
  when:  
    def result = underTest.convert(input)  
  
  then:  
    result.name == input.name  
    result.address == input.address  
    result.type == input.type  
}
```

Utility classes



```
def "user conversion works" () {  
  given:  
    def input = UserMother.johnny()  
  
  when:  
    def result = underTest.convert(input)  
  
  then:  
    Utils.matchProperties(result, input,  
                          "name", "address", "type", ...)  
}
```



Anti-pattern #3



```
def "user service can create user" () {  
  given:  
    def user = UserMother.johnny()  
    def userForm = new UserForm(user.name, user.address, user.type)  
  
  when:  
    def result = underTest.createUser(userForm)  
  
  then:  
    1 * userRepository.persist(_ as User) >> user  
    0 * userRepository._  
    result == user  
}
```

Testing implementation vs contract – solution



```
def "user service can create user" () {  
  given:  
    def user = UserMother.johnny()  
    def userForm = new UserForm(user.name, user.address, user.type)  
    userRepository.persist(_ as User) >> user  
  
  when:  
    def result = underTest.createUser(userForm)  
  
  then:  
    0 * userRepository._  
    result == user  
}
```

Anti-pattern #4 – setup



```
@PostMapping
@ResponseBody
public UserDto create(@RequestBody @Valid UserForm userForm) {

    User user = userService.create(userForm);

    return userConverter.convert(user);

}
```

Anti-pattern #4



```
def "controller calls the service and converter" () {  
  given:  
    def user = UserMother.johnny()  
    def userForm = new UserForm(user.name, user.address, user.type)  
  
  when:  
    def result = underTest.create(form)  
  
  then:  
    1 * userService.create(form) >> user  
    1 * userConverter.convert(user) >> ...  
  
    result.name == user.name  
    result.address == user.address  
}
```

Don't write unit tests,
write integration tests!

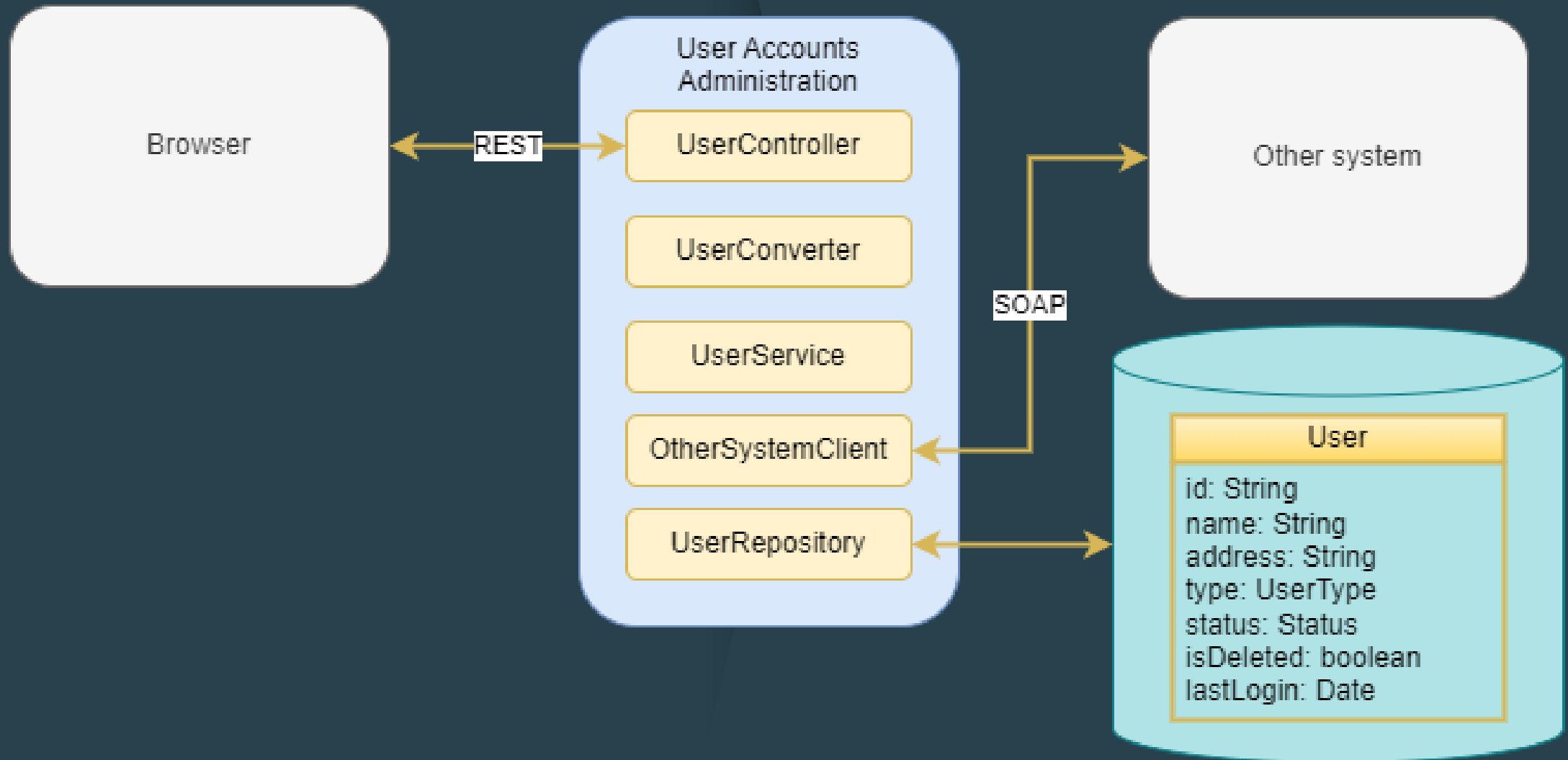


Why?



- A few Integration/API Tests replace numerous Unit Tests
- Easy to develop (e.g. in Spring - `@DataJpaTest`, `@MockBean`, `MockMvc`)
- Work with real REST requests / async events
- Allow access to the database (where test data may be set up in advance)
- 3rd party system calls are handled by mocking the API rather than our client code

A refresher



Integration test example (1)



```
def "user is created, updated and deactivated after some time" () {  
  given:  
    def user = UserMother.johnny()  
  
  when:  
    def createdUserDto = /** trigger create REST API */  
  
  then:  
    createdUserDto.name == user.name  
  
    userRepository.findByName(user.name) != null  
}
```

Integration test example (2)



when:

```
user.lastLogin = now().minusDays(30)
```

```
def updatedUserDto = /** trigger update REST API */
```

then:

```
updatedUserDto.name == user.name
```

```
updatedUserDto.lastLogin == user.lastLogin
```

when:

```
jobsUtil.triggerUserPeriodicJob()
```

```
def userDto = /** trigger read REST API */
```

then:

```
userDto.name == user.name
```

```
userDto.status == UserStatus.Deactivated
```

Don't write integration tests,
write unit tests!



Source code example



```
def updateState(newState: UserState) {  
    var validStates = []  
    when (this.state) {  
        Inactive -> validStates = [Active, Deleted]  
        Active -> validStates = [Inactive]  
    }  
  
    if (validStates.contains(newState)) {  
        state = newState  
    } else {  
        throw IllegalArgumentException()  
    }  
}
```



Unit test example



```
@Unroll
def "updateState should change state from #currentState to #newState"() {
  given:
    def user = new User(state: currentState)
  when:
    user.updateState(newState)
  then:
    user.state == newState

  where:
    currentState      | newState
    UserState.Inactive | UserState.Active
    UserState.Inactive | UserState.Deleted
    UserState.Active  | UserState.Inactive
}
```



AI benefits in testing

Popular tools



ChatGPT



GitHub
Copilot



This is a Java method that is annotated with `@PostMapping` and maps to the endpoint `/submit/{userId}/{type}`. The method takes in four parameters: `userId`, `type`, `status`, and `submissionType`. `userId` and `type` are path variables, while `status` and `submissionType` are request parameters. The `submitPracticalTask` method generates a `PracticalTaskPrimaryKeyDTO` object using the `userId` and `type` parameters, and then calls the `submitPracticalTaskSync` method of the `practicalTaskService` object, passing in the `PracticalTaskPrimaryKeyDTO` object, `status`, and `submissionType` as arguments.

The purpose of this method is to handle the submission of a practical task by a user. The `userId` parameter is used to identify the user who is submitting the task, while the `type` parameter is used to specify the type of task being submitted. The `status` parameter is used to specify the status of the task submission, with a default value of

```

72         PracticalTask practicalTask = practicalTaskService.create
73         practicalTaskMapper.practicalTaskToPracticalTaskBaseDTO(p
74     }
75
76     @Operation(summary = "Endpoint to submit practical task")
77     @PostMapping("/submit/{userId}/{type}")
78     public void submitPracticalTask(@PathVariable UUID userId,
79     @PathVariable PracticalTaskType
80     @RequestParam(defaultValue =
81     @RequestParam(defaultValue =
82     PracticalTaskPrimaryKeyDTO primaryKeyDTO = generatePracti
83     practicalTaskService.submitPracticalTaskSync(primaryKeyDT
84     }
85
86     @Operation(summary = "Endpoint to delete a practical task")
87     @DeleteMapping("/{userId}/{type}")
88     public PracticalTaskBaseDTO deletePracticalTask(@PathVariable
89     PracticalTaskPrimaryKeyDTO primaryKeyDTO = generatePracti
90
91     Prac
92     retu
93     }
94
95     @Operation(summary = "Endpoint to start practical task")
96     @PostMapping("/start/{userId}/{type}")
97     public PracticalTaskBaseDTO startPracticalTask(@PathVariable
98     @PathVariable

```


Explanation

```

> main > java > com > musala > artemis > modules > practicaltask > controller > J AdminPracticalTaskController.java > AdminPracticalTaskController.java
41 @RequiredArgsConstructor
42 @RequestMapping("/practical-tasks")
43 @Tag(name = "Practical Task Controller", description = "Endpoints handling admin operation")
44 @FeatureToggle(FeatureToggleEnum.BE_PRACTICAL_TASK)
45 public class AdminPracticalTaskController {
46
47     private final PracticalTaskService practicalTaskService;
48     private final PracticalTaskMapper practicalTaskMapper = Mappers.getMapper(clazz:PracticalTaskMapper.class);
49
50     @Operation(summary = "Endpoint to get the practical task")
51     @GetMapping("/{userId}/{type}")
52     public PracticalTaskBaseDTO getPracticalTask(@PathVariable UUID userId, @PathVariable PracticalTaskTypeEnum type, @PathVariable PracticalTaskPrimaryKeyDTO primaryKeyDTO = generatePracticalTaskPrimaryKey(userId, type));
53     private PracticalTaskBaseDTO practicalTaskBaseDTO = practicalTaskService.getByPrimarykey(userId, type, primaryKeyDTO);
54     private PracticalTask practicalTask = practicalTaskService.getByPrimarykey(primaryKeyDTO);
55     private PracticalTaskBaseDTO practicalTaskBaseDTO = practicalTaskMapper.practicalTaskToPracticalTaskBaseDTO(practicalTask);
56     return practicalTaskBaseDTO;
57 }
58
59 @Operation(summary = "Endpoint to get all practical task of user")
60 @GetMapping("/{userId}")
61 public List<PracticalTaskBaseDTO> getAllPracticalTaskOfUser(@PathVariable UUID userId) {
62     List<PracticalTask> practicalTasks = practicalTaskService.getByUserId(userId);
63     List<PracticalTaskBaseDTO> practicalTaskBaseDTOS = practicalTaskMapper.practicalTasksToPracticalTaskBaseDTOS(practicalTasks);
64     return practicalTaskBaseDTOS;
65 }
66
67 }
68
69 @Operation(summary = "Endpoint to create new practical task")
70 @PostMapping("/{userId}")
71 public PracticalTaskBaseDTO createPracticalTask(@PathVariable UUID userId, @RequestBody PracticalTaskCreateDTO practicalTaskCreateDTO) {
72     PracticalTask practicalTask = practicalTaskService.createPracticalTask(userId, practicalTaskCreateDTO);
73     PracticalTaskBaseDTO practicalTaskBaseDTO = practicalTaskMapper.practicalTaskToPracticalTaskBaseDTO(practicalTask);
74     return practicalTaskBaseDTO;
75 }
76 }

```



 petyodimitrov-musala

generate test

 GitHub Copilot



Here's an example of a test for the
getPracticalTask method:

```

@Test
void getPracticalTask_ReturnsP
    UUID userId = UUID.randomUUID();
    PracticalTaskTypeEnum type = PracticalTaskTypeEnum.PRIMARY_KEY;
    PracticalTaskPrimaryKeyDTO primaryKeyDTO = generatePracticalTaskPrimaryKey(userId, type);
    PracticalTask practicalTask = practicalTaskService.getByPrimarykey(userId, type, primaryKeyDTO);
    PracticalTaskBaseDTO expectedPracticalTaskBaseDTO = practicalTaskMapper.practicalTaskToPracticalTaskBaseDTO(practicalTask);

    when(practicalTaskService.getByPrimarykey(userId, type, primaryKeyDTO)).thenReturn(practicalTask);

    mockMvc.perform(get(ADMIN_PRACTICAL_TASKS_URL + "/{userId}/{type}", userId, type)
        .contentType(MediaType.APPLICATION_JSON)
        .andExpect(status().isOk())
        .andExpect(content().contentType(MediaType.APPLICATION_JSON))
        .andExpect(jsonPath("$.type").value(type.name()))
        .andExpect(jsonPath("$.userId").value(userId.toString()))
        .andExpect(jsonPath("$.primaryKey").value(primaryKeyDTO.getPrimaryKey()))
        .andExpect(jsonPath("$.description").value(practicalTask.getDescription()))
        .andExpect(jsonPath("$.status").value(practicalTask.getStatus().name()))
    )
    .andExpect(jsonPath("$.type").value(type.name()))
    .andExpect(jsonPath("$.userId").value(userId.toString()))
    .andExpect(jsonPath("$.primaryKey").value(primaryKeyDTO.getPrimaryKey()))
    .andExpect(jsonPath("$.description").value(practicalTask.getDescription()))
    .andExpect(jsonPath("$.status").value(practicalTask.getStatus().name()))
    .andExpect(jsonPath("$.description").value(practicalTask.getDescription()))
    .andExpect(jsonPath("$.status").value(practicalTask.getStatus().name()))
    .andExpect(jsonPath("$.description").value(practicalTask.getDescription()))
    .andExpect(jsonPath("$.status").value(practicalTask.getStatus().name()));
}

```

Test generation

What is the purpose of unit testing?



```
/**
 * 1. Function to create 100 random users using faker library
 * 2. Each user should have between 1 and 5 posts
 * 3. The user and the post should be sent to the API via a post request
 */
public static void createUserData() {
    Faker faker = new Faker();
    Random random = new Random();

    for (int i = 0; i < 100; i++) {
        String user = faker.name().fullName();
        int userId = i + 1;
        Api.createUser(user, userId);

        int numPosts = random.nextInt(5) + 1; // Generate a random number between 1 and 5

        for (int j = 0; j < numPosts; j++) {
            int postId = j + 1;
            String postTitle = faker.lorem().sentence(10); // Generate a sentence with 10 words
            String postBody = faker.lorem().paragraph(4); // Generate a paragraph with 4 sentences
            Api.createPost(postTitle, postBody, userId);
        }
    }
}
```

Test data

Test	Duration
FunctionalTests (12)	
IntegrationTests (4)	
PublicApiIntegrationTests (15)	
UnitTests (44)	700 ms
Microsoft.eShopWeb.UnitTests.ApplicationCore...	30 ms
BasketAddItem (6)	19 ms
AddsBasketItemIfNotPresent	< 1 ms
CantAddItemWithNegativeQuantity	< 1 ms
CantModifyQuantityToNegativeNumber	19 ms
DefaultsToQuantityOfOne	< 1 ms
IncrementsQuantityOfItemIfPresent	< 1 ms
KeepsOriginalUnitPriceIfMoreItemsAdded	< 1 ms
BasketRemoveEmptyItems (1)	5 ms
RemovesEmptyBasketItems	5 ms
BasketTotalItems (2)	6 ms
ReturnsTotalQuantityWithMultipleItems	1 ms
ReturnsTotalQuantityWithOneItem	5 ms
Microsoft.eShopWeb.UnitTests.ApplicationCore...	4 ms
OrderTotal (3)	4 ms
IsCorrectGiven1Item	< 1 ms
IsCorrectGiven3Items	< 1 ms
IsZeroForNewOrder	4 ms
Microsoft.eShopWeb.UnitTests.ApplicationCore...	52 ms
JsonExtensions (3)	52 ms
CorrectlyDeserializesJson (2)	39 ms
CorrectlyDeserializesJson(json: "{ \"id\": 3...	< 1 ms
CorrectlyDeserializesJson(json: "{ \"id\": 9...	39 ms
CorrectlySerializesAndDeserializesObject	13 ms
Microsoft.eShopWeb.UnitTests.ApplicationCore...	253 ms
AddItemToBasket (2)	81 ms
InvokesBasketRepositoryGetBySpecAsyncO...	1 ms
InvokesBasketRepositoryUpdateAsyncOnce	80 ms
DeleteBasket (1)	85 ms
ShouldInvokeBasketRepositoryDeleteAsync...	85 ms
TransferBasket (4)	87 ms

Run | Debug | Ask Copilot

Test Detail Summary

✖ Microsoft.eShopWeb.UnitTests.ApplicationCore.Entities.BasketTests.Ba

Source: `BasketAddItem.cs` line 28

Duration: < 1 ms

Message:

`Microsoft.eShopWeb.ApplicationCore.Exceptions.Dupl:`

Stack Trace:

`Basket.AddItem(Int32 catalogItemId, Decimal unitPr:
BasketAddItem.IncrementsQuantityOfItemIfPresent()
RuntimeMethodHandle.InvokeMethod(Object target, Vo:
MethodInvoker.Invoke(Object obj, IntPtr* args, Bin`

Debug

GitHub Copilot

Hi! Let's build together.

I'm powered by AI, so surprises and mistakes are possible. Make sure to verify any generated code or suggestions, and [share internal feedback](#).

- **Include slash** commands (i.e. `/fix`) at the beginning of your prompt to indicate intent.
- **Type #** to refer to code you want to include.
- **Use the Alt + /** shortcut to open the inline chat and refine code in the editor.

Use `/help` for more guidance.

Debug the failed test `BasketAddItem.IncrementsQuantityOfItemIfPresent`.

GitHub Copilot

Here is an approach for debugging this failure:

The exception is thrown at line 37 in the `AddItem` method of the `Basket` class. This happens when an item with the same `catalogItemId` is added to the basket again.

You should inspect the `catalogItemId` parameter passed to the `AddItem` method and the `_items` list in the `Basket` class. Check if an item with the same `catalogItemId` already exists in the `_items` list before the `AddItem` method is called.

Also, check the `BasketAddItem.IncrementsQuantityOfItemIfPresent` test method in the `BasketTests` class. Inspect the values passed to the `AddItem` method in this test.

```
Basket.AddItem(int catalogItemId, decimal unitPrice, int quantity = 1)
{
    Basket._items
    BasketItem.AddQuantity(int quantity)
    BasketTests.BasketAddItem.IncrementsQuantityOfItemIfPresent()
}
```

Preview

Starting test debugging.

References

Was this helpful? 🗨️

```
Test Explorer | BasketItem.cs | Basket.cs | Microsoft.eShopWeb.ApplicationCore.Entities.BasketAggr | AddItem(int catalogItemId, decimal unitPrice, int quantity)
12 {
13     12 references | 2/4 passing | 0 changes | 0 authors, 0 changes
14     public string BuyerId { get; private set; }
15     private readonly List<BasketItem> _items = new List<BasketItem>();
16     23 references | 3/8 passing | 0 changes | 0 authors, 0 changes
17     public IReadOnlyCollection<BasketItem> Items => _items.AsReadOnly();
18
19     2 references | 1/2 passing | 0 changes | 0 authors, 0 changes
20     public int TotalItems => _items.Sum(i => i.Quantity);
21
22     21 references | 9/16 passing | 0 changes | 0 authors, 0 changes
23     public Basket(string buyerId)
24     {
25         BuyerId = buyerId;
26     }
27
28     public void AddItem(int catalogItemId, decimal unitPrice, int quantity)
29     {
30         if (!Items.Any(i => i.CatalogItemId == catalogItemId))
31         {
32             //if (!ValidateItemID(catalogItemId))
33             //    return;
34
35             _items.Add(new BasketItem(catalogItemId, unitPrice, quantity));
36             return;
37         }
38         else
39         {
40             throw new DuplicateException("Duplicate item ID");
41         }
42
43         var existingItem = Items.First(i => i.CatalogItemId == catalogItemId);
44         existingItem.AddQuantity(quantity);
45     }
46 }
```

Debug Test: Microsoft.eShopWeb.UnitTests.ApplicationCore.Entities.BasketTests.Basket...

unitPrice);" line is executed?

What happens when the 'AddItem' method is called the second time?

References

Current values at line 25 of C:\dev\eshop\src\ApplicationCore\Entities\BasketAggregate\Basket.cs are:

- catalogItemId = 123
- unitPrice = 1.23M
- quantity = 2
- _items = Count = 1

GitHub Copilot

The `_items` list now contains an item, and the `catalogItemId` being added is the same as the one already in the list. This will cause the `DuplicateException` to be thrown, leading to the test failure.

To fix this, the code should be updated to increment the quantity of the existing item instead of throwing an exception. The `else` block throwing the `DuplicateException` should be removed, and the code to increment the quantity should be moved inside the `if` block.

```
public void AddItem(int catalogItemId, decimal unitPrice, int quantity)
{
    var existingItem = Items.FirstOrDefault(i => i.CatalogItemId == catalogItemId);

    if (existingItem == null)
    {
        _items.Add(new BasketItem(catalogItemId, unitPrice, quantity));
    }
    else
    {
        existingItem.AddQuantity(quantity);
    }
}
```

Preview

Does the updated 'AddItem' method now pass the test?

Are there any other tests that fail due to this code change?

References



Debug

AI considerations



- Syntax & logical errors
- Limited edge case coverage
- Not as creative *
- Difficulty with complex tests
- Data Privacy
- Intellectual Property

ChatGPT config



Temporary Chat

Not in history

Temporary chats won't appear in your history. For safety purposes, we keep a copy of your chat for up to 30 days.

No model training

Temporary chats won't be used to improve our models.

Follows Custom Instructions

Custom Instructions will still be followed if you have them enabled.

Continue

Model improvement

Improve the model for everyone

Allow your content to be used to train our models, which makes ChatGPT better for you and everyone who uses it. We take steps to protect your privacy. [Learn more](#)

Delete all chats

Delete all

Copilot config



Suggestions matching public code

Allowed ▾

Copilot can allow or block suggestions matching public code. Learn more about [code suggestions](#)

Allow GitHub to use my code snippets from the code editor for product improvements

Allow GitHub, its affiliates and third parties to use my code snippets to research and improve GitHub product features. More information in [About GitHub Copilot privacy](#).

✓ **Allowed**

GitHub Copilot will show suggestions matching public code.

Blocked

GitHub Copilot won't show suggestions matching public code.

For more information about the data your organization receives regarding your use of GitHub Copilot, please review [GitHub's Privacy Statement](#).

Summary



- Don't Repeat Yourself
- Avoid mock overuse
- Test contracts, not implementation internals
- Don't test for the sake of testing
- Distribute your tests across the test pyramid
- Let AI help you

*To me, legacy code is
simply code...
without tests*

Michael Feathers
Working Effectively
With Legacy Code

Qinshift 



Questions?



Thanks!

Atanas Gardev

Software Architect

atanas.gardev@musala.com

Petyo Dimitrov

Senior Software Architect

petyo.dimitrov@musala.com

Qinshift