

TDD на бекенде

Александр Шинкарев

Test-Driven Development (TDD) разработка через тестирование — это когда сначала тесты потом код. Всё так просто.....

Назад в 90-е



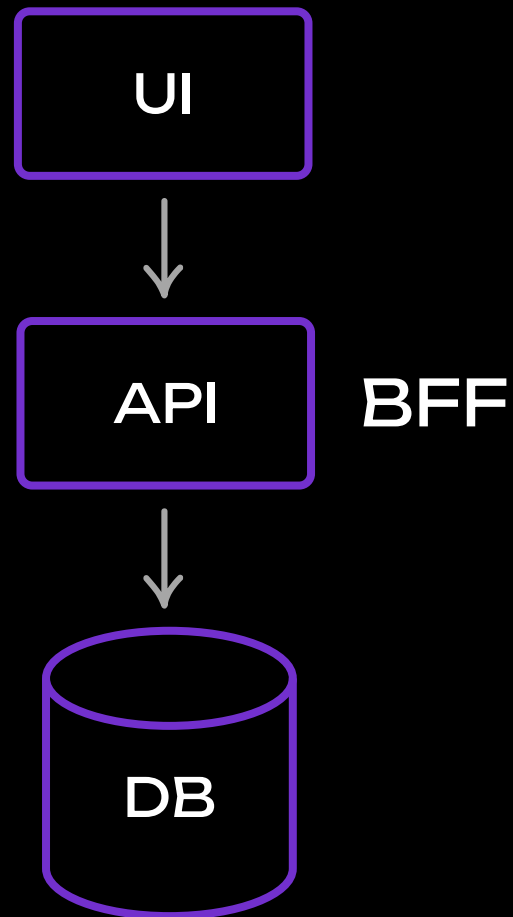
Кадр из к/ф «Идиократия» (Idiocracy), реж. Майк Джадж, 2006 г. Правообладатель: 20th Century Studios.

Любимое To Do приложение

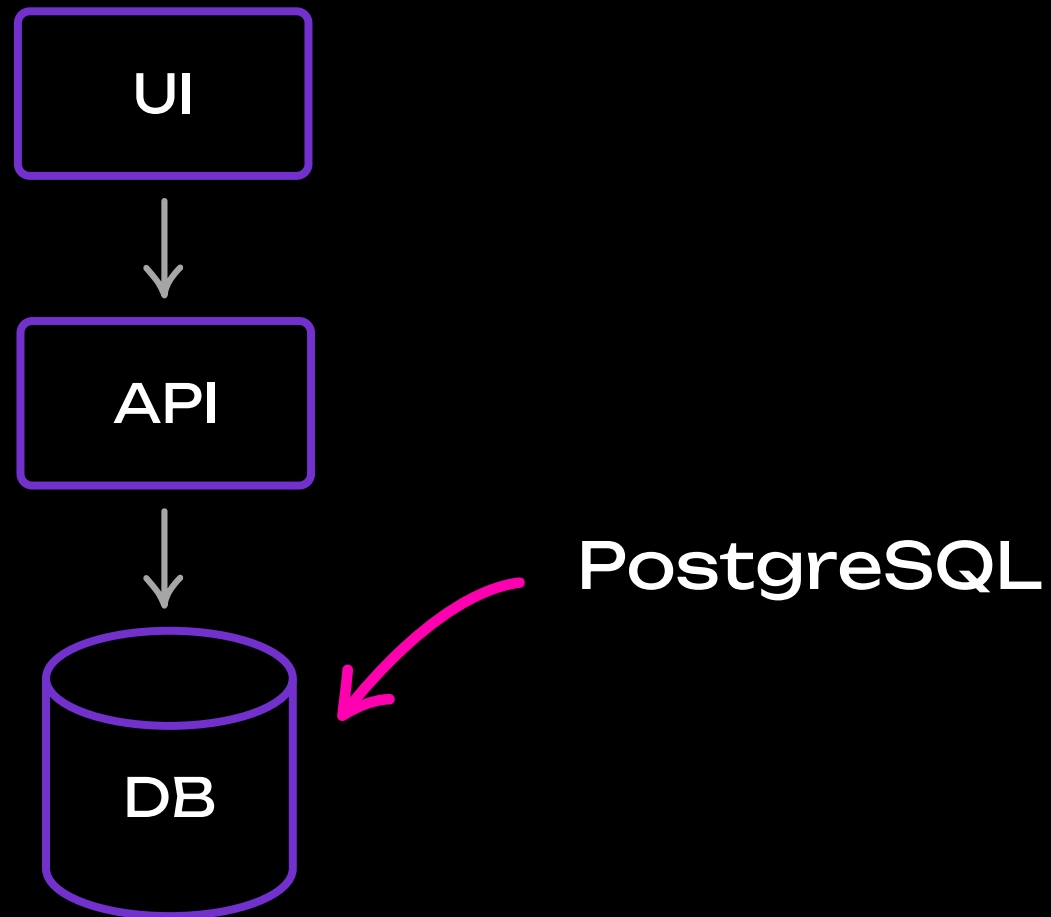
- ✓ Много простого кода на C#
- ✓ Большой контекст на удержание
- ✓ Авторский (спорный) подход
- ✓ Всё будет доступно в 2 открытых репозиториях



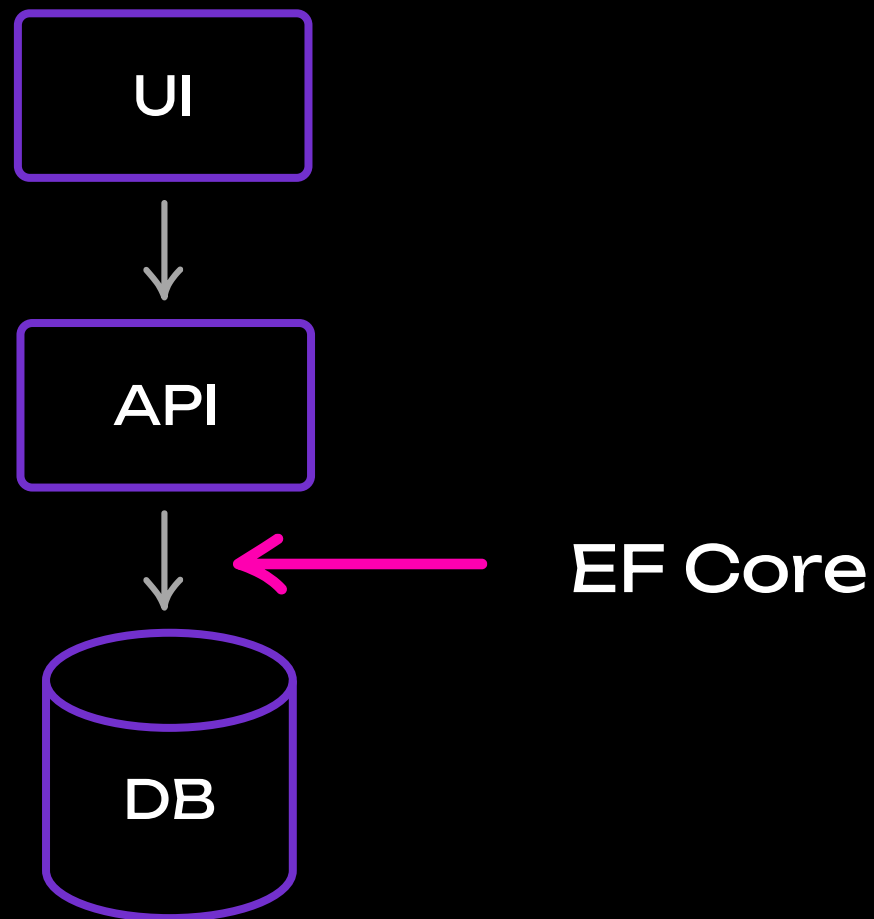
Архитектура и из чего сделаны наши бекенды



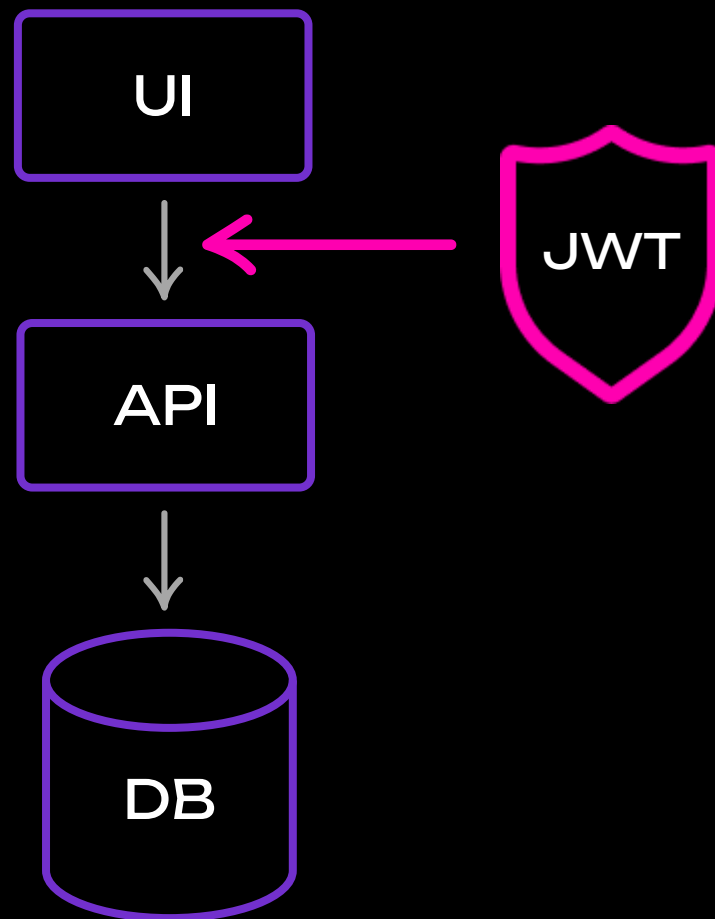
Архитектура и из чего сделаны наши бекенды



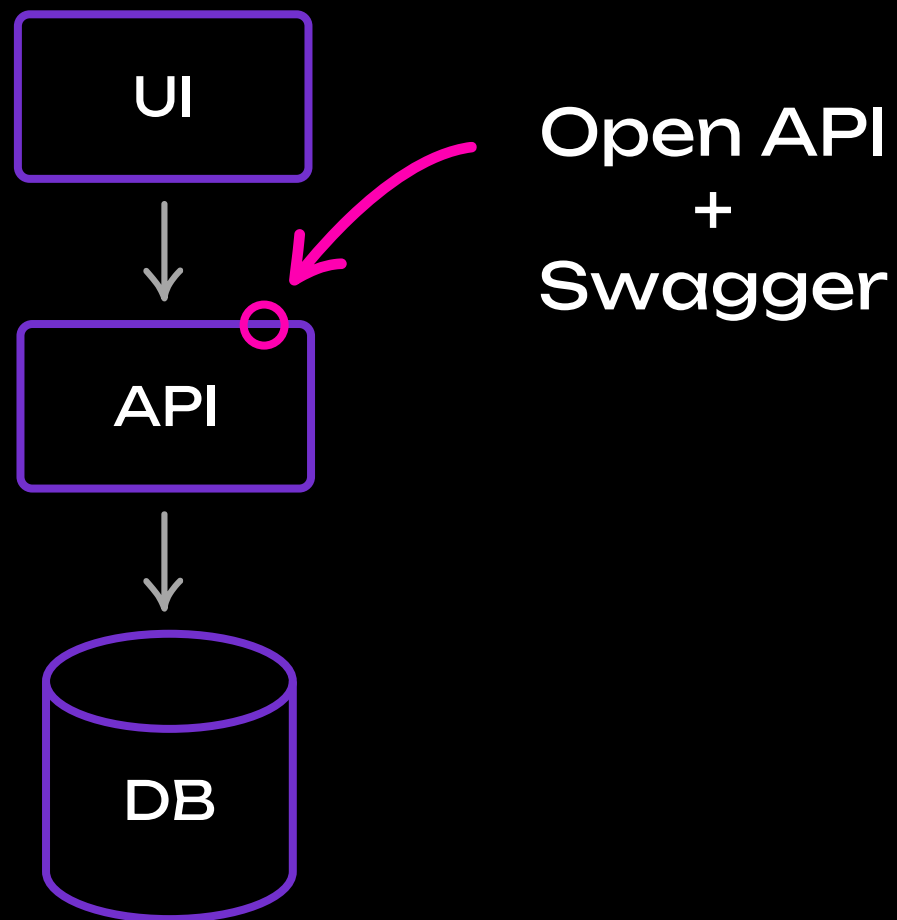
Архитектура и из чего сделаны наши бекенды



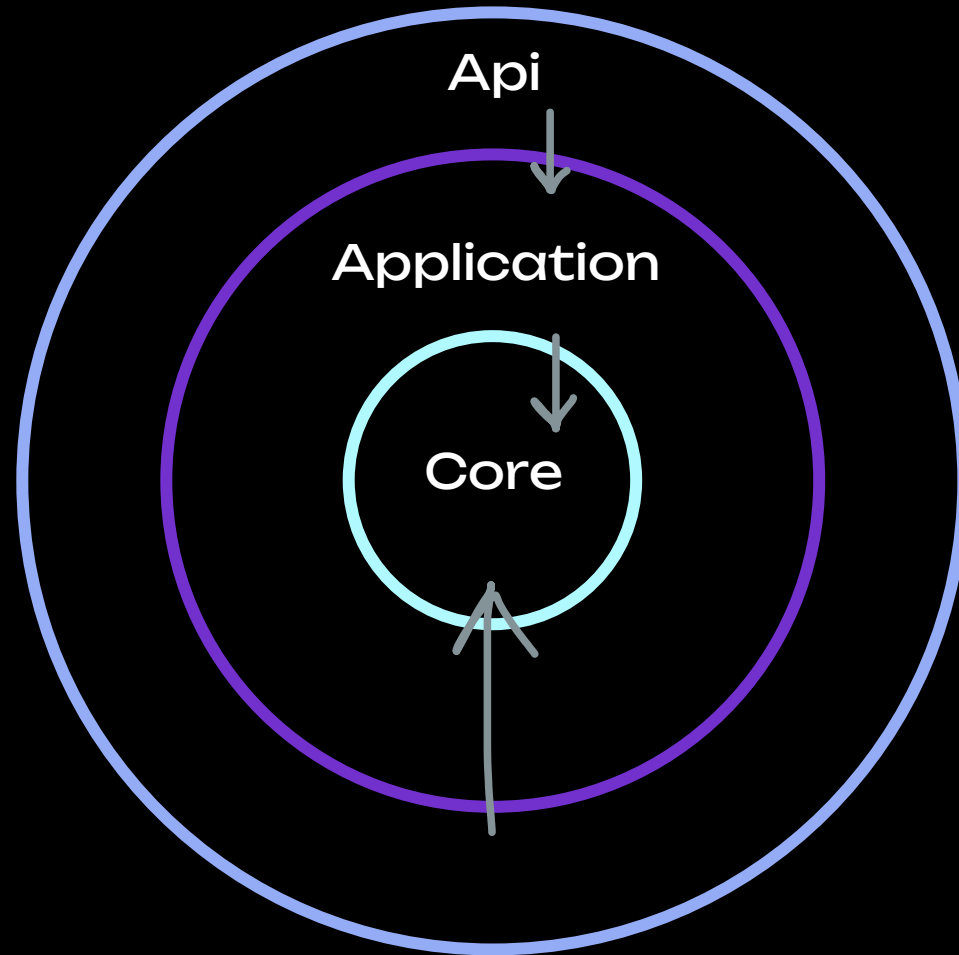
Архитектура и из чего сделаны наши бекенды



Архитектура и из чего сделаны наши бекенды



Слои



Процесс

User Story

Контракт

Тестовая
стратегия

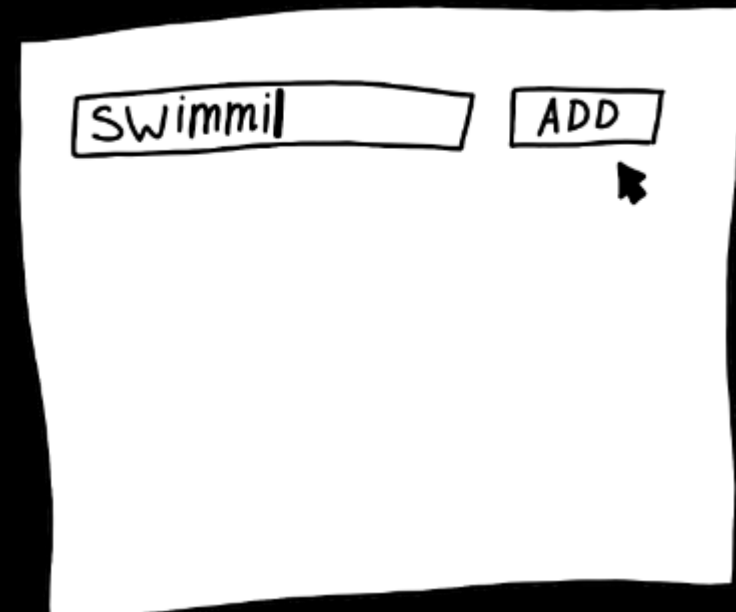
TDD

User Sory^t

Новая To Do

Хочу создавать новые To Do с именем

Чтобы я мог отслеживать свои дела



Список To Do

Хочу видеть мои To Do от самых новых до самых старых

Чтобы знать что ещё осталось сделать



Контракт на пальцах

ЭНДПОИНТ СОЗДАНИЯ

POST /to-dos

Request Body

```
{  
  "name": "Dishes"  
}
```

Response

```
{  
  "newToDoId": 2  
}
```

Эндпоинт на чтение

GET /to-dos

Response

```
{  
  "todos": [  
    {  
      "id": 1,  
      "name": "Test"  
    }  
  ]  
}
```

Эндпоинт удаления

DELETE /to-dos?toDold=2

Response

```
{  
  "isDeleted": true  
}
```

Контракт на C#

✓ Api
✓ Features/Todos
C# TodosController.cs U

```
[ApiController]
```

```
[Route("to-dos")]
```

```
public class TodosController : ControllerBase
```

Дихотомия добра и зла DTO

Request

- Тело запроса

Response

- Ответ сервера



Кадры из к/ф «Пираты Карибского моря: Сундук мертвеца» (Pirates of the Caribbean: Dead Man's Chest), реж. Гор Вербински, 2006 г. Правообладатель: Walt Disney Pictures.

- Api
 - Features/Todos
 -  TodosController.cs
- Application
 - Features/Todos/Handlers
 - CreateToDo
 -  CreateToDoRequest.cs U

```
public class CreateToDoRequest
{
    public required string Name { get; set; }
}
```

- ▼ Api
 - ▼ Features/Todos
 - 🔗 TodosController.cs
- ▼ Application
 - ▼ Features/Todos/Handlers
 - ▼ CreateToDo
 - 🔗 CreateToDoRequest.cs
 - 🔗 CreateToDoResponse.csU

```
public class CreateToDoResponse
{
    public required long NewToDoId { get; set; }
}
```

- Api
 - Features/Todos
 - ToDoController.cs** M
- Application
 - Features/Todos/Handlers
 - CreateToDo
 - CreateToDoRequest.cs
 - CreateToDoResponse.cs

[HttpPost]

```
public Task<CreateToDoResponse> CreateToDoAsync(  
    [Required] [FromBody] CreateToDoRequest createToDoRequest  
)  
{  
    throw new NotImplementedException();  
}
```



- ✓ Api
 - ✓ Features/Todos
 - 🔗 TodosController.cs
- ✓ Application
 - ✓ Features/Todos/Handlers
 - ✓ CreateToDo
 - 🔗 CreateToDoRequest.cs
 - 🔗 CreateToDoResponse.cs
 - ✓ GetTodos
 - 🔗 GetTodosResponse.cs U

```
public class GetTodosResponse
{
    public required List<ToDoDto> Todos { get; set; }
}
```

- ✓ Api
 - ✓ Features/Todos
 - 🔗 TodosController.cs
- ✓ Application
 - ✓ Features/Todos/Handlers
 - ✓ CreateToDo
 - 🔗 CreateToDoRequest.cs
 - 🔗 CreateToDoResponse.cs
 - ✓ GetTodos
 - 🔗 GetTodosResponse.cs U

```
public class ToDoDto
{
    public required long Id { get; set; }

    public required string Name { get; set; }
}
```

- ▼ Api
 - ▼ Features/Todos
 - ToDoController.cs M**
- ▼ Application
 - ▼ Features/Todos/Handlers
 - ▼ CreateToDo
 - CreateToDoRequest.cs
 - CreateToDoResponse.cs
 - ▼ GetToDo
 - GetToDoResponse.cs

[HttpGet]

```
public Task<GetToDoResponse> GetToDoAsync()  
{  
    throw new NotImplementedException();  
}
```

- ✓ Api
 - ✓ Features/Todos
 - 🔗 TodosController.cs
- ✓ Application
 - ✓ Features/Todos/Handlers
 - ✓ CreateToDo
 - 🔗 CreateToDoRequest.cs
 - 🔗 CreateToDoResponse.cs
 - ✓ DeleteToDo
 - 🔗 DeleteToDoResponse.cs
 - ✓ GetTodos
 - 🔗 GetTodosResponse.cs

```
public class DeleteToDoResponse
{
    public required bool IsDeleted { get; set; }
}
```

- ▼ Api
 - ▼ Features/Todos
 - ToDoController.cs M**
- ▼ Application
 - ▼ Features/Todos/Handlers
 - ▼ CreateToDo
 - CreateToDoRequest.cs
 - CreateToDoResponse.cs
 - ▼ DeleteToDo
 - DeleteToDoResponse.cs
 - ▼ GetTodos
 - GetTodosResponse.cs

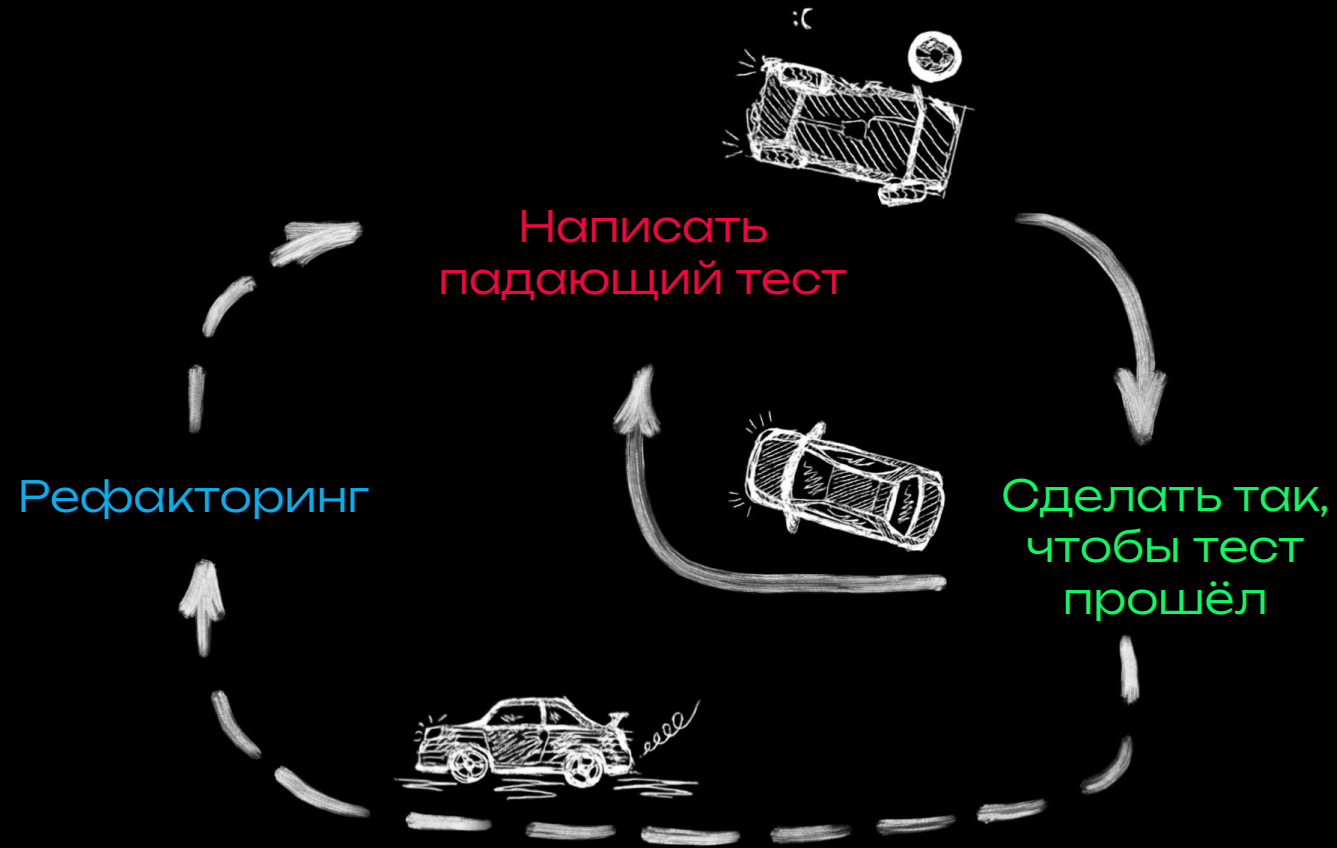
[HttpDelete]

```
public Task<DeleteToDoResponse> DeleteToDoAsync(  
    [Required] [FromQuery] long toDoId  
)  
{  
    throw new NotImplementedException();  
}
```

Тестовая стратегия

Памятка по TDD

Красный — падает
Зелёный — проходит
Синий — улучшить



E2E-тесты

// </> () <>

== <> {} ==

// </>

// == </>

<> {} == /

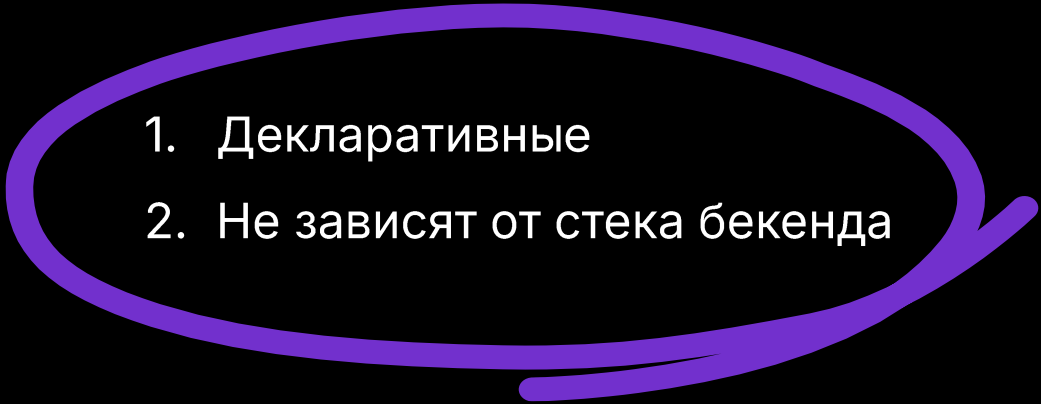
// </> () <> {}

Принципы наших E2E

- ✓ Покрывают только Happy Path
- ✓ Вызывают каждый эндпоинт хотя бы 1 раз
- ✓ Проверяют, что сохранённое также читается

E2E-тесты на Karate

```
Given url apiRootUrl
And path 'swagger/openapi.json'
When method GET
Then status 200
And match response.info ==
"""
{
  "title": "to-dos-api-dotnet"
}
"""
```

- 
1. Декларативные
 2. Не зависят от стека бекенда

Наш E2E-сценарий

- ✓ Создадим To Do
- ✓ Прочитаем его
- ✓ Удалим за собой
- ✓ Проверим, что удалилось



- Api
 - Features/Todos
 - ToDoController.cs
- Application
 - Features/Todos/Handlers
 - CreateToDo
 - CreateToDoRequest.cs
 - CreateToDoResponse.cs
 - DeleteToDo
 - DeleteToDoResponse.cs
 - GetTodos
 - GetTodosResponse.cs

- e2e
 - to-dos-happy-path.featureU

```
* def randomToDoName = '[API-E2E]-to-do-' + Math.random()
```

Given url apiRootUrl

And path 'to-dos'

And request

```
"""
```

```
{
```

```
  "name": "#(randomToDoName)",
```

```
}
```

```
"""
```

When method POST

Then status 200

```
* def newToDoId = response.newToDoId
```

- ✓ Api
 - ✓ Features/Todos
 - 🔗 TodosController.cs
- ✓ Application
 - ✓ Features/Todos/Handlers
 - ✓ CreateToDo
 - 🔗 CreateToDoRequest.cs
 - 🔗 CreateToDoResponse.cs
 - ✓ DeleteToDo
 - 🔗 DeleteToDoResponse.cs
 - ✓ GetTodos
 - 🔗 GetTodosResponse.cs
- ✓ e2e
 - ≡ to-dos-happy-path.featureU

Given url apiRootUrl

And path 'to-dos'

When method GET

Then match response.todos contains

"""

```
{  
  "id": "#(newToDoId)",  
  "name": "#(randomToDoName)",  
}
```

"""

- ✓ Api
 - ✓ Features/Todos
 - ⚙️ TodosController.cs
- ✓ Application
 - ✓ Features/Todos/Handlers
 - ✓ CreateToDo
 - ⚙️ CreateToDoRequest.cs
 - ⚙️ CreateToDoResponse.cs
 - ✓ DeleteToDo
 - ⚙️ DeleteToDoResponse.cs
 - ✓ GetTodos
 - ⚙️ GetTodosResponse.cs
- ✓ e2e
 - ≡ to-dos-happy-path.featureU

Given url apiRootUrl

Given path 'to-dos'

And params { toDoId: "#(newToDoId)" }

When method DELETE

Then status 200

And match response ==

"""

{

"isDeleted": true,

}

"""

- ✓ Api
 - ✓ Features/Todos
 - ⚙️ TodosController.cs
- ✓ Application
 - ✓ Features/Todos/Handlers
 - ✓ CreateToDo
 - ⚙️ CreateToDoRequest.cs
 - ⚙️ CreateToDoResponse.cs
 - ✓ DeleteToDo
 - ⚙️ DeleteToDoResponse.cs
 - ✓ GetTodos
 - ⚙️ GetTodosResponse.cs
- ✓ e2e
 - ≡ to-dos-happy-path.featureU

Given url apiRootUrl

Given path 'to-dos'

When method GET

Then status 200

And assert response.todos.every(x => x.id !== newToDoId)

// </> () <>

== <> { } ==

// </>

// == </>

<> { } == /

// </> () <> { }

● Тесты падают

Фейковая реализация (опционально)

Статья М. Фаулера Mocks Aren't Stubs (2007)

- ✓ **Fake** objects actually have working implementations, but usually take some shortcut which makes them not suitable for production (an [in memory database](#) is a good example).



[Статья М.
Фаулера
про моки
и заглушки](#)

- Api
 - Features/Todos
 - ToDoController.cs
- Application
 - Features/Todos/Handlers
 - CreateToDo
 - CreateToDoRequest.cs
 - CreateToDoResponse.cs
 - DeleteToDo
 - DeleteToDoResponse.cs
 - GetTodos
 - GetTodosResponse.cs
- Core
 - Features/Todos/Entities
 - ToDo.cs U
- e2e
 - to-dos-happy-path.feature

```
public class ToDo
{
    public long Id { get; set; }

    public required string Name { get; set; }
}
```

- ✓ Api
 - ✓ Features/Todos
 - 🔗 **ToDoController.cs** M
- ✓ Application
 - ✓ Features/Todos/Handlers
 - ✓ CreateToDo
 - 🔗 CreateToDoRequest.cs
 - 🔗 CreateToDoResponse.cs
 - ✓ DeleteToDo
 - 🔗 DeleteToDoResponse.cs
 - ✓ GetTodos
 - 🔗 GetTodosResponse.cs
- ✓ Core
 - ✓ Features/Todos/Entities
 - 🔗 ToDo.cs
- ✓ e2e
 - ☰ to-dos-happy-path.feature

```
public class ToDoController : ControllerBase
{
    private static long _nextToDoId = 0;
    private static readonly
        ConcurrentDictionary<long, ToDo> Todos = new();
}
```

- Api
 - Features/Todos
 - ToDoController.cs M**
- Application
 - Features/Todos/Handlers
 - CreateToDo
 - CreateToDoRequest.cs
 - CreateToDoResponse.cs
 - DeleteToDo
 - DeleteToDoResponse.cs
 - GetTodos
 - GetTodosResponse.cs
- Core
 - Features/Todos/Entities
 - ToDo.cs
- e2e
 - to-dos-happy-path.feature

[HttpPost]

```
public Task<CreateToDoResponse> CreateToDoAsync(
    [Required] [FromBody] CreateToDoRequest createToDoRequest
)
{
    var newToDo = new ToDo
    {
        Id = Interlocked.Increment(ref _nextToDoId),
        Name = createToDoRequest.Name,
    };
    Todos[newToDo.Id] = newToDo;

    return Task.FromResult(
        new CreateToDoResponse() { NewToDoId = newToDo.Id }
    );
}
```

- Api
 - Features/Todos
 - ToDoController.cs** M
- Application
 - Features/Todos/Handlers
 - CreateToDo
 - CreateToDoRequest.cs
 - CreateToDoResponse.cs
 - DeleteToDo
 - DeleteToDoResponse.cs
 - GetTodos
 - GetTodosResponse.cs
- Core
 - Features/Todos/Entities
 - ToDo.cs
- e2e
 - to-dos-happy-path.feature

[HttpGet]

```
public Task<GetTodosResponse> GetTodosAsync()
{
    return Task.FromResult(
        new GetTodosResponse
        {
            Todos = Todos
                .Values.Select(x => new ToDoDto {
                    Id = x.Id,
                    Name = x.Name,
                })
                .ToList(),
        }
    );
}
```

- ✓ Api
 - ✓ Features/Todos
 - 🔗 **TodosController.cs** M
- ✓ Application
 - ✓ Features/Todos/Handlers
 - ✓ CreateToDo
 - 🔗 CreateToDoRequest.cs
 - 🔗 CreateToDoResponse.cs
 - ✓ DeleteToDo
 - 🔗 DeleteToDoResponse.cs
 - ✓ GetTodos
 - 🔗 GetTodosResponse.cs
 - ✓ Core
 - ✓ Features/Todos/Entities
 - 🔗 ToDo.cs
 - ✓ e2e
 - ≡ to-dos-happy-path.feature

```
[HttpDelete]
public Task<DeleteToDoResponse> DeleteToDoAsync(
    [Required] [FromQuery] long toDoId
)
{
    return Task.FromResult(
        new DeleteToDoResponse {
            IsDeleted = Todos.Remove(toDoId, out _),
        }
    );
}
```

● Тесты проходят

// </> () <>

== <> {} ==

// </>

// == </>

<> {} == /

// </> () <> {}

Добавляем базу

- ✓ Api
 - ✓ Features/Todos
 - 🔗 TodosController.cs
- ✓ Application
 - ✓ Features/Todos/Handlers
 - ✓ CreateToDo
 - 🔗 CreateToDoRequest.cs
 - 🔗 CreateToDoResponse.cs
 - ✓ DeleteToDo
 - 🔗 DeleteToDoResponse.cs
 - ✓ GetTodos
 - 🔗 GetTodosResponse.cs
- ✓ Core
 - ✓ Features/Todos/Entities
 - 🔗 **ToDo.cs** M
- ✓ e2e
 - ≡ to-dos-happy-path.feature

```
public class ToDo
{
    public ToDo() { }

    public long Id { get; set; }

    public required string Name { get; set; }
}
```

- ▼ Api
 - ▼ Features/Todos
 - 🔗 TodosController.cs
- ▼ Application
 - ▼ EntityTypeConfigurations
 - 🔗 **ToDoEntityTypeConfigu..U**
 - ▼ Features/Todos/Handlers
 - ▼ CreateToDo
 - 🔗 CreateToDoRequest.cs
 - 🔗 CreateToDoResponse.cs
 - ▼ DeleteToDo
 - 🔗 DeleteToDoResponse.cs
 - ▼ GetTodos
 - 🔗 GetTodosResponse.cs
- ▼ Core
 - ▼ Features/Todos/Entities
 - 🔗 ToDo.cs
- ▼ e2e
 - ≡ to-dos-happy-path.feature

```
public class ToDoEntityTypeConfiguration
    : IEntityTypeConfiguration<ToDo>
{
    public void Configure(EntityTypeBuilder<ToDo> builder)
    {
        builder.HasIndex(x => x.Id).IsUnique();

        builder.Property(x => x.Name).IsRequired()
            .HasMaxLength(256);
    }
}
```

- ▼ Api
 - ▼ Features/Todos
 - 🔗 TodosController.cs
- ▼ Application
 - ▼ EntityTypeConfigurations
 - 🔗 ToDoEntityTypeConfigu...
 - ▼ Features/Todos/Handlers
 - ▼ CreateToDo
 - 🔗 CreateToDoRequest.cs
 - 🔗 CreateToDoResponse.cs
 - ▼ DeleteToDo
 - 🔗 DeleteToDoResponse.cs
 - ▼ GetTodos
 - 🔗 GetTodosResponse.cs
 - 🔗 AppDbContext.cs U
 - ▼ Core
 - ▼ Features/Todos/Entities
 - 🔗 ToDo.cs
 - ▼ e2e
 - ≡ to-dos-happy-path.feature

```
public class AppDbContext : DbContext
{
    public AppDbContext(
        DbContextOptions<AppDbContext> options
    ) : base(options) { }

    public AppDbContext() { }

    public virtual DbSet<ToDo> Todos { get; set; }
```

Создаём миграцию

```
dotnet ef migrations add AddToDos  
--startup-project ./Api/Api.csproj  
--project ./Application/Application.csproj  
--context AppDbContext  
--verbose
```

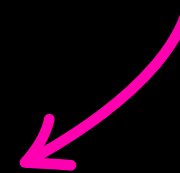
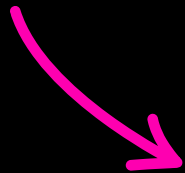
Немного не о CQRS

Command

- Меняет состояние в базе (пишет)
- Принимает Request

Query

- Читает из базы без модификаций
- Возвращает сущности из Core



Handler

- Орkestрирует команды и квери
- Маппит результат в Response

- Api
 - Features/Todos
 - ToDoController.cs
- Application
 - EntityTypeConfigurations
 - ToDoEntityTypeConfigu...
 - Features/Todos/Handlers
 - CreateToDo
 - CreateToDoCommand.cs U
 - CreateToDoRequest.cs
 - CreateToDoResponse.cs
 - DeleteToDo
 - DeleteToDoResponse.cs
 - GetTodos
 - GetTodosResponse.cs
 - AppDbContext.cs
- Core
 - Features/Todos/Entities
 - ToDo.cs
- e2e
 - to-dos-happy-path.feature

```

public class CreateToDoCommand(AppDbContext context)
{
    public async Task<long> ExecuteAsync(
        CreateToDoRequest createToDoRequest
    )
    {
        var newToDo = new ToDo {
            Name = createToDoRequest.Name,
        };

        await context.Todos.AddAsync(newToDo);

        await context.SaveChangesAsync();

        return newToDo.Id;
    }
}

```

- ▼ Api
 - ▼ Features/Todos
 - 🔗 TodosController.cs
- ▼ Application
 - ▼ EntityTypeConfigurations
 - 🔗 ToDoEntityTypeConfigu...
 - ▼ Features/Todos/Handlers
 - ▼ CreateToDo
 - 🔗 CreateToDoCommand.cs
 - 🔗 **CreateToDoHandler.cs U**
 - 🔗 CreateToDoRequest.cs
 - 🔗 CreateToDoResponse.cs
 - ▼ DeleteToDo
 - 🔗 DeleteToDoResponse.cs
 - ▼ GetTodos
 - 🔗 GetTodosResponse.cs
 - 🔗 AppDbContext.cs
- ▼ Core
 - ▼ Features/Todos/Entities
 - 🔗 ToDo.cs
- ▼ e2e
 - ≡ to-dos-happy-path.feature

```

public class CreateToDoHandler(
    CreateToDoCommand createToDoCommand
)
{
    public async Task<CreateToDoResponse> HandleAsync(
        CreateToDoRequest createToDoRequest
    )
    {
        var newToDoId = await createToDoCommand.ExecuteAsync(
            createToDoRequest
        );

        return new CreateToDoResponse { NewToDoId = newToDoId };
    }
}

```

- ▼ Api
 - ▼ Features/Todos
 - # TodosController.cs M**
- ▼ Application
 - ▼ EntityTypeConfigurations
 - # ToDoEntityTypeConfigu...
 - ▼ Features/Todos/Handlers
 - ▼ CreateToDo
 - # CreateToDoCommand.cs
 - # CreateToDoHandler.cs
 - # CreateToDoRequest.cs
 - # CreateToDoResponse.cs
 - ▼ DeleteToDo
 - # DeleteToDoResponse.cs
 - ▼ GetTodos
 - # GetTodosResponse.cs
 - # AppDbContext.cs
- ▼ Core
 - ▼ Features/Todos/Entities
 - # ToDo.cs
- ▼ e2e
 - ≡ to-dos-happy-path.feature

[HttpPost]

```
public Task<CreateToDoResponse> CreateToDoAsync(
    [Required] [FromBody] CreateToDoRequest createToDoRequest,
    [FromServices] CreateToDoHandler createToDoHandler
)
{
    return createToDoHandler.HandleAsync(createToDoRequest);
}
```

- Api
 - Features/Todos
 - ToDoController.cs
- Application
 - EntityTypeConfigurations
 - ToDoEntityTypeConfigu...
 - Features/Todos/Handlers
 - CreateToDo
 - CreateToDoCommand.cs
 - CreateToDoHandler.cs
 - CreateToDoRequest.cs
 - CreateToDoResponse.cs
 - DeleteToDo
 - DeleteToDoResponse.cs
 - GetTodos
 - GetTodosQuery.cs U
 - GetTodosResponse.cs
 - AppDbContext.cs
- Core
 - Features/Todos/Entities
 - ToDo.cs
- e2e
 - to-dos-happy-path.feature

```
public class GetTodosQuery(AppDbContext context)
{
    public Task<List<ToDo>> GetAsync()
    {
        return context
            .QueryableAsNoTracking<ToDo>()
            .ToListAsync();
    }
}
```

- Api
 - Features/Todos
 - ToDoController.cs
- Application
 - EntityTypeConfigurations
 - ToDoEntityTypeConfigu...
 - Features/Todos/Handlers
 - CreateToDo
 - CreateToDoCommand.cs
 - CreateToDoHandler.cs
 - CreateToDoRequest.cs
 - CreateToDoResponse.cs
 - DeleteToDo
 - DeleteToDoResponse.cs
 - GetTodos
 - GetToDoHandler.cs U
 - GetToDoQuery.cs
 - GetToDoResponse.cs
 - AppDbContext.cs
- Core
 - Features/Todos/Entities
 - ToDo.cs
- e2e
 - to-dos-happy-path.feature

```

public class GetTodosHandler(GetTodosQuery getTodosQuery)
{
    public async Task<GetTodosResponse> HandleAsync()
    {
        var todos = await getTodosQuery.GetAsync();
        return new GetTodosResponse
        {
            Todos = todos
                .Select(x => new ToDoDto {
                    Id = x.Id,
                    Name = x.Name,
                })
                .ToList(),
        };
    }
}

```

- ▼ Api
 - ▼ Features/Todos
 - # TodosController.cs M**
- ▼ Application
 - ▼ EntityTypeConfigurations
 - # ToDoEntityTypeConfigu...
 - ▼ Features/Todos/Handlers
 - ▼ CreateToDo
 - # CreateToDoCommand.cs
 - # CreateToDoHandler.cs
 - # CreateToDoRequest.cs
 - # CreateToDoResponse.cs
 - ▼ DeleteToDo
 - # DeleteToDoResponse.cs
 - ▼ GetTodos
 - # GetTodosHandler.cs
 - # GetTodosQuery.cs
 - # GetTodosResponse.cs
 - # AppDbContext.cs
- ▼ Core
 - ▼ Features/Todos/Entities
 - # ToDo.cs
- ▼ e2e
 - ≡ to-dos-happy-path.feature

[HttpGet]

```
public Task<GetTodosResponse> GetTodosAsync(  
    [FromServices] GetTodosHandler getTodosHandler  
)  
{  
    return getTodosHandler.HandleAsync();  
}
```

- ▼ Api
 - ▼ Features/Todos
 - 🔗 TodosController.cs
- ▼ Application
 - ▼ EntityTypeConfigurations
 - 🔗 ToDoEntityTypeConfigu...
 - ▼ Features/Todos/Handlers
 - ▼ CreateToDo
 - 🔗 CreateToDoCommand.cs
 - 🔗 CreateToDoHandler.cs
 - 🔗 CreateToDoRequest.cs
 - 🔗 CreateToDoResponse.cs
 - ▼ DeleteToDo
 - 🔗 DeleteToDoCommand.cs U
 - 🔗 DeleteToDoResponse.cs
 - ▼ GetTodos
 - 🔗 GetTodosHandler.cs
 - 🔗 GetTodosQuery.cs
 - 🔗 GetTodosResponse.cs
 - 🔗 AppDbContext.cs
- ▼ Core
 - ▼ Features/Todos/Entities
 - 🔗 ToDo.cs
- ▼ e2e
 - ≡ to-dos-happy-path.feature

```
public class DeleteToDoCommand(AppDbContext context)
{
    public async Task<bool> ExecuteAsync(long todoId)
    {
        var todoToDelete = await context
            .Queryable<ToDo>()
            .SingleAsync(x => x.Id == todoId);

        context.Set<ToDo>().Remove(todoToDelete);

        await context.SaveChangesAsync();

        return true;
    }
}
```

- ▼ Api
 - ▼ Features/Todos
 - 🔗 TodosController.cs
- ▼ Application
 - ▼ EntityTypeConfigurations
 - 🔗 ToDoEntityTypeConfigu...
 - ▼ Features/Todos/Handlers
 - ▼ CreateToDo
 - 🔗 CreateToDoCommand.cs
 - 🔗 CreateToDoHandler.cs
 - 🔗 CreateToDoRequest.cs
 - 🔗 CreateToDoResponse.cs
 - ▼ DeleteToDo
 - 🔗 DeleteToDoCommand.cs
 - 🔗 DeleteToDoHandler.cs U
 - 🔗 DeleteToDoResponse.cs
 - ▼ GetTodos
 - 🔗 GetTodosHandler.cs
 - 🔗 GetTodosQuery.cs
 - 🔗 GetTodosResponse.cs
 - 🔗 AppDbContext.cs
- ▼ Core
 - ▼ Features/Todos/Entities
 - 🔗 ToDo.cs
- ▼ e2e
 - ≡ to-dos-happy-path.feature

```

public class DeleteToDoHandler(
    DeleteToDoCommand deleteToDoCommand
)
{
    public async Task<DeleteToDoResponse> HandleAsync(
        long todoId
    )
    {
        return new DeleteToDoResponse
        {
            IsDeleted = await deleteToDoCommand.ExecuteAsync(
                todoId
            ),
        };
    }
}

```

- ▼ Api
 - ▼ Features/Todos
 - C# TodosController.cs M**
- ▼ Application
 - ▼ EntityTypeConfigurations
 - C# ToDoEntityTypeConfigu...
 - ▼ Features/Todos/Handlers
 - ▼ CreateToDo
 - C# CreateToDoCommand.cs
 - C# CreateToDoHandler.cs
 - C# CreateToDoRequest.cs
 - C# CreateToDoResponse.cs
 - ▼ DeleteToDo
 - C# DeleteToDoCommand.cs
 - C# DeleteToDoHandler.cs
 - C# DeleteToDoResponse.cs
 - ▼ GetTodos
 - C# GetTodosHandler.cs
 - C# GetTodosQuery.cs
 - C# GetTodosResponse.cs
 - C# AppDbContext.cs
- ▼ Core
 - ▼ Features/Todos/Entities
 - C# ToDo.cs
- ▼ e2e
 - ≡ to-dos-happy-path.feature

[HttpDelete]

```
public Task<DeleteToDoResponse> DeleteToDoAsync(  
    [Required] [FromQuery] long toDoId,  
    [FromServices] DeleteToDoHandler deleteToDoHandler  
)  
{  
    return deleteToDoHandler.HandleAsync(toDoId);  
}
```

● Тесты всё ещё проходят

Список To Do

Хочу видеть мои To Do от самых новых до самых старых

Чтобы знать что ещё осталось сделать



Тестим по настоящей базе

EF Core

EF Core itself
contains over 30,000
tests against SQL
Server alone

to-dos-api-dotnet

1 тест за 2 секунды
Пайплайн < 2 минут

inner-circle-time-api

106 тестов за 3 секунды
Пайплайн < 2 минут

Настоящестъ

Параллельность

Изолированность



[Дока
тестировани
я от EF Core](#)

- ✓ Api
 - ✓ Features/Todos
 - C# TodosController.cs
- ✓ Application
 - ✓ EntityTypeConfigurations
 - C# ToDoEntityTypeConfigu...
 - ✓ Features/Todos/Handlers
 - ✓ CreateToDo
 - C# CreateToDoCommand.cs
 - C# CreateToDoHandler.cs
 - C# CreateToDoRequest.cs
 - C# CreateToDoResponse.cs
 - ✓ DeleteToDo
 - C# DeleteToDoCommand.cs
 - C# DeleteToDoHandler.cs
 - C# DeleteToDoResponse.cs
 - ✓ GetTodos
 - C# GetTodosHandler.cs
 - C# GetTodosQuery.cs
 - C# GetTodosQueryTests.cs
 - C# GetTodosResponse.cs
 - C# AppDbContext.cs
- ✓ Core
 - ✓ Features/Todos/Entities
 - C# ToDo.cs
- ✓ e2e
 - ≡ to-dos-happy-path.feature

```
public class GetTodosQueryTests : IntegrationTestBase
{
    [Fact]
    public async Task GetAsync_ShouldOrderNewerTodosFirst()
    {
        var context = CreateAppDbContext();
```

- ▼ Api
 - ▼ Features/Todos
 - C# TodosController.cs
- ▼ Application
 - ▼ EntityTypeConfigurations
 - C# ToDoEntityTypeConfigu...
 - ▼ Features/Todos/Handlers
 - ▼ CreateToDo
 - C# CreateToDoCommand.cs
 - C# CreateToDoHandler.cs
 - C# CreateToDoRequest.cs
 - C# CreateToDoResponse.cs
 - ▼ DeleteToDo
 - C# DeleteToDoCommand.cs
 - C# DeleteToDoHandler.cs
 - C# DeleteToDoResponse.cs
 - ▼ GetTodos
 - C# GetTodosHandler.cs
 - C# GetTodosQuery.cs
 - C# GetTodosQueryTests.cs
 - C# GetTodosResponse.cs
 - C# AppDbContext.cs
 - C# IntegrationTestBaseTest...
- ▼ Core
 - ▼ Features/Todos/Entities
 - C# ToDo.cs
- ▼ e2e
 - ≡ to-dos-happy-path.feature

```
protected AppDbContext CreateAppDbContext()
{
    var context = new AppDbContext(_dbContextOptions);

    context.Database.UseTransaction(_dbTransaction);

    return context;
}
```

- ▼ Api
 - ▼ Features/Todos
 - C# TodosController.cs
- ▼ Application
 - ▼ EntityTypeConfigurations
 - C# ToDoEntityTypeConfigu...
 - ▼ Features/Todos/Handlers
 - ▼ CreateToDo
 - C# CreateToDoCommand.cs
 - C# CreateToDoHandler.cs
 - C# CreateToDoRequest.cs
 - C# CreateToDoResponse.cs
 - ▼ DeleteToDo
 - C# DeleteToDoCommand.cs
 - C# DeleteToDoHandler.cs
 - C# DeleteToDoResponse.cs
 - ▼ GetTodos
 - C# GetTodosHandler.cs
 - C# GetTodosQuery.cs
 - C# GetTodosQueryTests.cs
 - C# GetTodosResponse.cs
 - C# AppDbContext.cs
 - C# IntegrationTestBaseTest...
- ▼ Core
 - ▼ Features/Todos/Entities
 - C# ToDo.cs
- ▼ e2e
 - ≡ to-dos-happy-path.feature

```
_dbConnection = new NpgsqlConnection(connectionString);
```

```
await _dbConnection.OpenAsync();
```

```
await ApplyMigrationsAsync();
```

```
_dbTransaction = await _dbConnection.BeginTransactionAsync(  
    System.Data.IsolationLevel.Serializable  
);
```

- ▼ Api
 - ▼ Features/Todos
 - C# TodosController.cs
- ▼ Application
 - ▼ EntityTypeConfigurations
 - C# ToDoEntityTypeConfigu...
 - ▼ Features/Todos/Handlers
 - ▼ CreateToDo
 - C# CreateToDoCommand.cs
 - C# CreateToDoHandler.cs
 - C# CreateToDoRequest.cs
 - C# CreateToDoResponse.cs
 - ▼ DeleteToDo
 - C# DeleteToDoCommand.cs
 - C# DeleteToDoHandler.cs
 - C# DeleteToDoResponse.cs
 - ▼ GetTodos
 - C# GetTodosHandler.cs
 - C# GetTodosQuery.cs
 - C# GetTodosQueryTests.cs
 - C# GetTodosResponse.cs
 - C# AppDbContext.cs
 - C# IntegrationTestBaseTest...
- ▼ Core
 - ▼ Features/Todos/Entities
 - C# ToDo.cs
- ▼ e2e
 - ≡ to-dos-happy-path.feature

```

public async Task DisposeAsync()
{
    ...
    try
    {
        await _dbTransaction.RollbackAsync();
    }
    finally
    {
        await _dbTransaction.DisposeAsync();
    }
    ...
}

```

- ▼ Api
 - ▼ Features/Todos
 - C# TodosController.cs
- ▼ Application
 - ▼ EntityTypeConfigurations
 - C# ToDoEntityTypeConfigu...
 - ▼ Features/Todos/Handlers
 - ▼ CreateToDo
 - C# CreateToDoCommand.cs
 - C# CreateToDoHandler.cs
 - C# CreateToDoRequest.cs
 - C# CreateToDoResponse.cs
 - ▼ DeleteToDo
 - C# DeleteToDoCommand.cs
 - C# DeleteToDoHandler.cs
 - C# DeleteToDoResponse.cs
 - ▼ GetTodos
 - C# GetTodosHandler.cs
 - C# GetTodosQuery.cs
 - C# GetTodosQueryTests.csm
 - C# GetTodosResponse.cs
 - C# AppDbContext.cs
 - C# IntegrationTestBaseTest...
- ▼ Core
 - ▼ Features/Todos/Entities
 - C# ToDo.cs
- ▼ e2e
 - ≡ to-dos-happy-path.feature

```
public class GetTodosQueryTests : IntegrationTestBase
{
    [Fact]
    public async Task GetAsync_ShouldOrderNewerTodosFirst()
    {
        var context = CreateAppDbContext();
    }
}
```

- ✓ Api
 - ✓ Features/Todos
 - C# TodosController.cs
- ✓ Application
 - ✓ EntityTypeConfigurations
 - C# ToDoEntityTypeConfigu...
 - ✓ Features/Todos/Handlers
 - ✓ CreateToDo
 - C# CreateToDoCommand.cs
 - C# CreateToDoHandler.cs
 - C# CreateToDoRequest.cs
 - C# CreateToDoResponse.cs
 - ✓ DeleteToDo
 - C# DeleteToDoCommand.cs
 - C# DeleteToDoHandler.cs
 - C# DeleteToDoResponse.cs
 - ✓ GetTodos
 - C# GetTodosHandler.cs
 - C# GetTodosQuery.cs
 - C# GetTodosQueryTests.csm
 - C# GetTodosResponse.cs
 - C# AppDbContext.cs
 - C# IntegrationTestBaseTest...
- ✓ Core
 - ✓ Features/Todos/Entities
 - C# ToDo.cs
- ✓ e2e
 - ≡ to-dos-happy-path.feature

```
var firstToDo = new ToDo { Name = "First" };  
await AddEntityAndSaveAsync(context, firstToDo);
```

```
var secondToDo = new ToDo { Name = "Second" };  
await AddEntityAndSaveAsync(context, secondToDo);
```

```
var thirdToDo = new ToDo { Name = "Third" };  
await AddEntityAndSaveAsync(context, thirdToDo);
```

- ▼ Api
 - ▼ Features/Todos
 - C# TodosController.cs
- ▼ Application
 - ▼ EntityTypeConfigurations
 - C# TodoEntityTypeConfigu...
 - ▼ Features/Todos/Handlers
 - ▼ CreateToDo
 - C# CreateToDoCommand.cs
 - C# CreateToDoHandler.cs
 - C# CreateToDoRequest.cs
 - C# CreateToDoResponse.cs
 - ▼ DeleteToDo
 - C# DeleteToDoCommand.cs
 - C# DeleteToDoHandler.cs
 - C# DeleteToDoResponse.cs
 - ▼ GetTodos
 - C# GetTodosHandler.cs
 - C# GetTodosQuery.cs
 - C# GetTodosQueryTests.csm
 - C# GetTodosResponse.cs
 - C# AppDbContext.cs
 - C# IntegrationTestBaseTest...
- ▼ Core
 - ▼ Features/Todos/Entities
 - C# ToDo.cs
- ▼ e2e
 - ≡ to-dos-happy-path.feature

}

```
var getTodosQuery = new GetTodosQuery(context);
```

```
var todos = await getTodosQuery.GetAsync();
```

```
Assert.Equal(thirdToDo.Name, todos[0].Name);
```

```
Assert.Equal(secondToDo.Name, todos[1].Name);
```

```
Assert.Equal(firstToDo.Name, todos[2].Name);
```

// </> () <>

== <> { } ==

// </>

// == </>

<> { } == /

// </> () <> { }

● Тесты падают

- ▼ Api
 - ▼ Features/Todos
 - C# TodosController.cs
- ▼ Application
 - ▼ EntityTypeConfigurations
 - C# TodoEntityTypeConfigu...
 - ▼ Features/Todos/Handlers
 - ▼ CreateToDo
 - C# CreateToDoCommand.cs
 - C# CreateToDoHandler.cs
 - C# CreateToDoRequest.cs
 - C# CreateToDoResponse.cs
 - ▼ DeleteToDo
 - C# DeleteToDoCommand.cs
 - C# DeleteToDoHandler.cs
 - C# DeleteToDoResponse.cs
 - ▼ GetTodos
 - C# GetTodosHandler.cs
 - C# GetTodosQuery.cs
 - C# GetTodosQueryTests.cs
 - C# GetTodosResponse.cs
 - C# AppDbContext.cs
 - C# IntegrationTestBaseTest...
- ▼ Core
 - ▼ Features/Todos/Entities
 - C# ToDo.cs
- ▼ e2e
 - ≡ to-dos-happy-path.feature

```
public class GetTodosQuery(AppDbContext context)
{
    public Task<List<ToDo>> GetAsync()
    {
        return context
            .QueryableAsNoTracking<ToDo>()
            .ToListAsync();
    }
}
```

- ▼ Api
 - ▼ Features/Todos
 - C# TodosController.cs
- ▼ Application
 - ▼ EntityTypeConfigurations
 - C# TodoEntityTypeConfigu...
 - ▼ Features/Todos/Handlers
 - ▼ CreateToDo
 - C# CreateToDoCommand.cs
 - C# CreateToDoHandler.cs
 - C# CreateToDoRequest.cs
 - C# CreateToDoResponse.cs
 - ▼ DeleteToDo
 - C# DeleteToDoCommand.cs
 - C# DeleteToDoHandler.cs
 - C# DeleteToDoResponse.cs
 - ▼ GetTodos
 - C# GetTodosHandler.cs
 - C# GetTodosQuery.cs M
 - C# GetTodosQueryTests.cs
 - C# GetTodosResponse.cs
 - C# AppDbContext.cs
 - C# IntegrationTestBaseTest...
- ▼ Core
 - ▼ Features/Todos/Entities
 - C# ToDo.cs
- ▼ e2e
 - ≡ to-dos-happy-path.feature

```
public class GetTodosQuery(AppDbContext context)
{
    public Task<List<ToDo>> GetAsync()
    {
        return context
            .QueryableAsNoTracking<ToDo>()
            .OrderByDescending(x => x.Id)
            .ToListAsync();
    }
}
```

● Тесты проходят

// </> () <>

== <> {} ==

// </>

// == </>

<> {} == /

// </> () <> {}

- > Api
- ✓ Application
 - ✓ Commands
 - > Contracts
 - ☑ CreateToDoCommand.cs
 - ☑ DeleteToDoCommand.cs
 - ☑ SoftDeleteToDoCommand.cs
 - 📡 Application.csproj
- > Core
- ✓ Tests
 - ✓ Application
 - ✓ Commands
 - ☑ CreateToDoCommandTests.cs
 - ☑ DeleteToDoCommandTests.cs
 - ☑ SoftDeleteToDoCommandTests.cs
 - > Queries
 - 📡 Tests.csproj



JavaScript

- > Api
- ✓ Application
 - ✓ Commands
 - > Contracts
 - ☑ CreateToDoCommand.cs
 - ☑ CreateToDoCommandTests.cs U
 - ☑ DeleteToDoCommand.cs
 - ☑ DeleteToDoCommandTests.cs U
 - ☑ SoftDeleteToDoCommand.cs
 - ☑ SoftDeleteToDoCommandTest... U



Кадр из к/ф «Властелин колец: Возвращение короля» (The Lord of the Rings: The Return of the King), реж. Питер Джексон, 2003 г. Правообладатель: New Line Cinema / Warner Bros.



[Статья на Хабре](#)

Статус To Do

Хочу видеть статус To Do, зависящий от даты и времени создания

Чтобы отличать новые от старых и забытых и, соответственно, косвенно видеть важное и не важное

New — прошло меньше недели

Old — прошло от недели до четырёх

Forgotten — прошло больше четырёх недель



Эндпоинт на чтение со статусом

GET /to-dos

Response

```
{
  "toDos": [
    {
      "id": 1,
      "name": "Test",
      "status": "New" | "Old" | "Forgotten",
    }
  ]
}
```

- ▼ Api
 - ▼ Features/Todos
 - C# TodosController.cs
- ▼ Application
 - ▼ EntityTypeConfigurations
 - C# ToDoEntityTypeConfigu...
 - ▼ Features/Todos/Handlers
 - ▼ CreateToDo
 - C# CreateToDoCommand.cs
 - C# CreateToDoHandler.cs
 - C# CreateToDoRequest.cs
 - C# CreateToDoResponse.cs
 - ▼ DeleteToDo
 - C# DeleteToDoCommand.cs
 - C# DeleteToDoHandler.cs
 - C# DeleteToDoResponse.cs
 - ▼ GetTodos
 - C# GetTodosHandler.cs
 - C# GetTodosQuery.cs
 - C# GetTodosQueryTests.cs
 - C# GetTodosResponse.cs
 - C# AppDbContext.cs
 - C# IntegrationTestBaseTest...
- ▼ Core
 - ▼ Features/Todos/Entities
 - C# ToDo.cs
 - C# ToDoStatus.cs U
- ▼ e2e
 - ≡ to-dos-happy-path.feature

```
public enum ToDoStatus
{
    None = 0,
    New = 1,
    Old = 2,
    Forgotten = 3,
}
```

- ▼ Api
 - ▼ Features/Todos
 - C# TodosController.cs
- ▼ Application
 - ▼ EntityTypeConfigurations
 - C# ToDoEntityTypeConfigu...
 - ▼ Features/Todos/Handlers
 - ▼ CreateToDo
 - C# CreateToDoCommand.cs
 - C# CreateToDoHandler.cs
 - C# CreateToDoRequest.cs
 - C# CreateToDoResponse.cs
 - ▼ DeleteToDo
 - C# DeleteToDoCommand.cs
 - C# DeleteToDoHandler.cs
 - C# DeleteToDoResponse.cs
 - ▼ GetTodos
 - C# GetTodosHandler.cs
 - C# GetTodosQuery.cs
 - C# GetTodosQueryTests.cs
 - C# GetTodosResponse.cs M
 - C# AppDbContext.cs
 - C# IntegrationTestBaseTest...
- ▼ Core
 - ▼ Features/Todos/Entities
 - C# ToDo.cs
 - C# ToDoStatus.cs
- ▼ e2e
 - ≡ to-dos-happy-path.feature

```
public class ToDoDto
{
    public required long Id { get; set; }

    public required string Name { get; set; }

    public required ToDoStatus Status { get; set; }
}
```

- ✓ Api
 - ✓ Features/Todos
 - ⚙️ TodosController.cs
- ✓ Application
 - ✓ EntityTypeConfigurations
 - ⚙️ ToDoEntityTypeConfigu...
 - ✓ Features/Todos/Handlers
 - ✓ CreateToDo
 - ⚙️ CreateToDoCommand.cs
 - ⚙️ CreateToDoHandler.cs
 - ⚙️ CreateToDoRequest.cs
 - ⚙️ CreateToDoResponse.cs
 - ✓ DeleteToDo
 - ⚙️ DeleteToDoCommand.cs
 - ⚙️ DeleteToDoHandler.cs
 - ⚙️ DeleteToDoResponse.cs
 - ✓ GetTodos
 - ⚙️ GetTodosHandler.cs
 - ⚙️ GetTodosQuery.cs
 - ⚙️ GetTodosQueryTests.cs
 - ⚙️ GetTodosResponse.cs
 - ⚙️ AppDbContext.cs
 - ⚙️ IntegrationTestBaseTest...
- ✓ Core
 - ✓ Features/Todos/Entities
 - ⚙️ ToDo.cs
 - ⚙️ ToDoStatus.cs
- ✓ e2e
 - ☰ to-dos-happy-path.featureM

Given url apiRootUrl

And path 'to-dos'

When method GET

Then match response.todos contains

```

"""
{
  "id": "#(newToDoId)",
  "name": "#(randomToDoName)",
  "status": "New",
}
"""

```

// </> () <>

== <> {} ==

// </>

// == </>

<> {} == /

// </> () <> {}

● Тесты падают

- ✓ Api
 - ✓ Features/Todos
 - C# TodosController.cs
- ✓ Application
 - ✓ EntityTypeConfigurations
 - C# TodoEntityTypeConfigu...
 - ✓ Features/Todos/Handlers
 - ✓ CreateToDo
 - C# CreateToDoCommand.cs
 - C# CreateToDoHandler.cs
 - C# CreateToDoRequest.cs
 - C# CreateToDoResponse.cs
 - ✓ DeleteToDo
 - C# DeleteToDoCommand.cs
 - C# DeleteToDoHandler.cs
 - C# DeleteToDoResponse.cs
 - ✓ GetTodos
 - C# GetTodosHandler.cs M
 - C# GetTodosQuery.cs
 - C# GetTodosQueryTests.cs
 - C# GetTodosResponse.cs
 - C# AppDbContext.cs
 - C# IntegrationTestBaseTest...
- ✓ Core
 - ✓ Features/Todos/Entities
 - C# ToDo.cs
 - C# ToDoStatus.cs
- ✓ e2e
 - ≡ to-dos-happy-path.feature

```
return new GetTodosResponse
{
    Todos = todos
        .Select(x => new ToDoDto
        {
            Id = x.Id,
            Name = x.Name,
            Status = ToDoStatus.New,
        })
        .ToList(),
};
```

● Тесты проходят

// </> () <>

== <> {} ==

// </>

// == </>

<> {} == /

// </> () <> {}

- ▼ Api
 - ▼ Features/Todos
 - 📄 TodosController.cs
- ▼ Application
 - ▼ EntityTypeConfigurations
 - 📄 TodoEntityTypeConfigu...
 - ▼ Features/Todos/Handlers
 - ▼ CreateToDo
 - 📄 CreateToDoCommand.cs
 - 📄 CreateToDoHandler.cs
 - 📄 CreateToDoRequest.cs
 - 📄 CreateToDoResponse.cs
 - ▼ DeleteToDo
 - 📄 DeleteToDoCommand.cs
 - 📄 DeleteToDoHandler.cs
 - 📄 DeleteToDoResponse.cs
 - ▼ GetTodos
 - 📄 GetTodosHandler.cs
 - 📄 GetTodosQuery.cs
 - 📄 GetTodosQueryTests.cs
 - 📄 GetTodosResponse.cs
 - 📄 AppDbContext.cs
 - 📄 IntegrationTestBaseTest...
- ▼ Core
 - ▼ Features/Todos/Entities
 - 📄 ToDo.cs
 - 📄 ToDoStatus.cs
- ▼ e2e
 - ≡ to-dos-happy-path.feature

```
return new GetTodosResponse
{
    Todos = todos
        .Select(x => new ToDoDto
        {
            Id = x.Id,
            Name = x.Name,
            Status = ToDoStatus.New,
        })
        .ToList(),
};
```

Куда положить
логику с расчётом
статуса?

Здесь идёт
проектирование
из-за TDD!

- ✓ Api
 - ✓ Features/Todos
 - C# TodosController.cs
- ✓ Application
 - ✓ EntityTypeConfigurations
 - C# ToDoEntityTypeConfigu...
 - ✓ Features/Todos/Handlers
 - ✓ CreateToDo
 - C# CreateToDoCommand.cs
 - C# CreateToDoHandler.cs
 - C# CreateToDoRequest.cs
 - C# CreateToDoResponse.cs
 - ✓ DeleteToDo
 - C# DeleteToDoCommand.cs
 - C# DeleteToDoHandler.cs
 - C# DeleteToDoResponse.cs
 - ✓ GetTodos
 - C# GetTodosHandler.cs
 - C# GetTodosQuery.cs
 - C# GetTodosQueryTests.cs
 - C# GetTodosResponse.cs
 - C# AppDbContext.cs
 - C# IntegrationTestBaseTest...
- ✓ Core
 - ✓ Features/Todos/Entities
 - C# ToDo.cs
 - C# ToDoStatus.cs
 - C# ToDoTests.cs U
- ✓ e2e
 - ≡ to-dos-happy-path.feature

[Fact]

```
public void GetStatus_FiveDaysAgoIsNew()
{
    var dt = new DateTime(
        2026, 01, 02, 03, 04, 05, 678, DateTimeKind.Utc
    );

    var todo = new ToDo {
        Name = "To Do",
        CreatedAtUtc = dt,
    };
}
```

- Api
 - Features/Todos
 - ToDoController.cs
- Application
 - EntityTypeConfigurations
 - ToDoEntityTypeConfigu...
 - Features/Todos/Handlers
 - CreateToDo
 - CreateToDoCommand.cs
 - CreateToDoHandler.cs
 - CreateToDoRequest.cs
 - CreateToDoResponse.cs
 - DeleteToDo
 - DeleteToDoCommand.cs
 - DeleteToDoHandler.cs
 - DeleteToDoResponse.cs
 - GetTodos
 - GetTodosHandler.cs
 - GetTodosQuery.cs
 - GetTodosQueryTests.cs
 - GetTodosResponse.cs
 - AppDbContext.cs
 - IntegrationTestBaseTest...
- Core
 - Features/Todos/Entities
 - ToDo.cs
 - ToDoStatus.cs
 - ToDoTests.cs U
- e2e
 - to-dos-happy-path.feature

```
var dateTimeProviderMock = new Mock<IDateTimeProvider>();

dateTimeProviderMock.Setup(x => x.UtcNow)
    .Returns(dt.AddDays(5));

var status = todo.GetStatus(dateTimeProviderMock.Object);

Assert.Equal(ToDoStatus.New, status);
}
```

// </> () <>

== <> {} ==

// </>

// == </>

<> {} == /

// </> () <> {}

● Тесты падают

- ▼ Api
 - ▼ Features/Todos
 - C# TodosController.cs
- ▼ Application
 - ▼ EntityTypeConfigurations
 - C# TodoEntityTypeConfigu...
 - ▼ Features/Todos/Handlers
 - ▼ CreateToDo
 - C# CreateToDoCommand.cs
 - C# CreateToDoHandler.cs
 - C# CreateToDoRequest.cs
 - C# CreateToDoResponse.cs
 - ▼ DeleteToDo
 - C# DeleteToDoCommand.cs
 - C# DeleteToDoHandler.cs
 - C# DeleteToDoResponse.cs
 - ▼ GetTodos
 - C# GetTodosHandler.cs
 - C# GetTodosQuery.cs
 - C# GetTodosQueryTests.cs
 - C# GetTodosResponse.cs
 - C# AppDbContext.cs
 - C# IntegrationTestBaseTest...
- ▼ Core
 - ▼ Features/Todos/Entities
 - C# ToDo.cs
 - C# ToDoStatus.cs
 - C# ToDoTests.cs
 - C# IDateTimeProvider.cs U
- ▼ e2e
 - ≡ to-dos-happy-path.feature

```
public interface IDateTimeProvider
{
    public DateTime.UtcNow { get; }
}
```

- ▼ Api
 - ▼ Features/Todos
 - C# TodosController.cs
- ▼ Application
 - ▼ EntityTypeConfigurations
 - C# ToDoEntityTypeConfigu...
 - ▼ Features/Todos/Handlers
 - ▼ CreateToDo
 - C# CreateToDoCommand.cs
 - C# CreateToDoHandler.cs
 - C# CreateToDoRequest.cs
 - C# CreateToDoResponse.cs
 - ▼ DeleteToDo
 - C# DeleteToDoCommand.cs
 - C# DeleteToDoHandler.cs
 - C# DeleteToDoResponse.cs
 - ▼ GetTodos
 - C# GetTodosHandler.cs
 - C# GetTodosQuery.cs
 - C# GetTodosQueryTests.cs
 - C# GetTodosResponse.cs
 - C# AppDbContext.cs
 - C# IntegrationTestBaseTest...
- ▼ Core
 - ▼ Features/Todos/Entities
 - C# ToDo.cs M
 - C# ToDoStatus.cs
 - C# ToDoTests.cs
 - C# IDateTimeProvider.cs
- ▼ e2e
 - ≡ to-dos-happy-path.feature

```
public class ToDo
```

```
{
```

```
    public ToDo() { }
```

```
    public long Id { get; set; }
```

```
    public required string Name { get; set; }
```

```
    public DateTime CreatedAtUtc { get; set; }
```

- ▼ Api
 - ▼ Features/Todos
 - C# TodosController.cs
- ▼ Application
 - ▼ EntityTypeConfigurations
 - C# ToDoEntityTypeConfigu...
 - ▼ Features/Todos/Handlers
 - ▼ CreateToDo
 - C# CreateToDoCommand.cs
 - C# CreateToDoHandler.cs
 - C# CreateToDoRequest.cs
 - C# CreateToDoResponse.cs
 - ▼ DeleteToDo
 - C# DeleteToDoCommand.cs
 - C# DeleteToDoHandler.cs
 - C# DeleteToDoResponse.cs
 - ▼ GetTodos
 - C# GetTodosHandler.cs
 - C# GetTodosQuery.cs
 - C# GetTodosQueryTests.cs
 - C# GetTodosResponse.cs
 - C# AppDbContext.cs
 - C# IntegrationTestBaseTest...
- ▼ Core
 - ▼ Features/Todos/Entities
 - C# ToDo.cs M
 - C# ToDoStatus.cs
 - C# ToDoTests.cs
 - C# IDateTimeProvider.cs
- ▼ e2e
 - ≡ to-dos-happy-path.feature

```

public ToDoStatus GetStatus(
    IDateTimeProvider dateTimeProvider
)
{
    var utcNow = dateTimeProvider.UtcNow;

    if (CreatedAtUtc > utcNow.AddDays(-7)
        && CreatedAtUtc < utcNow)
    {
        return ToDoStatus.New;
    }

    throw new InvalidOperationException(
        $"Not expected path of {nameof(GetStatus)}"
    );
}

```

// </> () <>

== <> { } ==

// </>

// == </>

<> { } ==

// </> () <> { }

● Код компилируется,
но тесты падают

GetAsync_ShouldOrderNewerToDosFirst

FAILED: GetAsync_ShouldOrderNewerToDosFirst (141ms)
Microsoft.EntityFrameworkCore.DbUpdateException : An error occurred while saving the entity changes. See the inner exception for details. ----
Npgsql.PostgresException : 42703: column "created_at_utc" of relation "to_dos" does not exist

Создаём миграцию для CreatedAtUtc колонки

```
dotnet ef migrations add AddToDosCreatedAtUtc  
--startup-project ./Api/Api.csproj  
--project ./Application/Application.csproj  
--context AppDbContext  
--verbose
```

● Тесты проходят

// </> () <>

== <> {} ==

// </>

// == </>

<> {} == /

// </> () <> {}

- ▼ Api
 - ▼ Features/Todos
 - C# TodosController.cs
- ▼ Application
 - ▼ EntityTypeConfigurations
 - C# TodoEntityTypeConfigu...
 - ▼ Features/Todos/Handlers
 - ▼ CreateToDo
 - C# CreateToDoCommand.cs
 - C# CreateToDoHandler.cs
 - C# CreateToDoRequest.cs
 - C# CreateToDoResponse.cs
 - ▼ DeleteToDo
 - C# DeleteToDoCommand.cs
 - C# DeleteToDoHandler.cs
 - C# DeleteToDoResponse.cs
 - ▼ GetTodos
 - C# GetTodosHandler.cs
 - C# GetTodosQuery.cs
 - C# GetTodosQueryTests.cs
 - C# GetTodosResponse.cs
 - C# AppDbContext.cs
 - C# IntegrationTestBaseTest...
- ▼ Core
 - ▼ Features/Todos/Entities
 - C# ToDo.cs
 - C# ToDoStatus.cs
 - C# ToDoTests.cs M
 - C# IDateTimeProvider.cs
- ▼ e2e
 - ≡ to-dos-happy-path.feature

[Fact]

```
public void GetStatus_EightDaysAgoIsOld()
{
    var dt = new DateTime(
        2026, 01, 02, 03, 04, 05, 678, DateTimeKind.Utc
    );

    var todo = new ToDo {
        Name = "To Do",
        CreatedAtUtc = dt,
    };
}
```

- Api
 - Features/Todos
 - C# TodosController.cs
- Application
 - EntityTypeConfigurations
 - C# ToDoEntityTypeConfigu...
 - Features/Todos/Handlers
 - CreateToDo
 - C# CreateToDoCommand.cs
 - C# CreateToDoHandler.cs
 - C# CreateToDoRequest.cs
 - C# CreateToDoResponse.cs
 - DeleteToDo
 - C# DeleteToDoCommand.cs
 - C# DeleteToDoHandler.cs
 - C# DeleteToDoResponse.cs
 - GetTodos
 - C# GetTodosHandler.cs
 - C# GetTodosQuery.cs
 - C# GetTodosQueryTests.cs
 - C# GetTodosResponse.cs
 - C# AppDbContext.cs
 - C# IntegrationTestBaseTest...
- Core
 - Features/Todos/Entities
 - C# ToDo.cs
 - C# ToDoStatus.cs
 - C# ToDoTests.cs M
 - C# IDateTimeProvider.cs
- e2e
 - to-dos-happy-path.feature

```
var dateTimeProviderMock = new Mock<IDateTimeProvider>();

dateTimeProviderMock.Setup(x => x.UtcNow)
    .Returns(dt.AddDays(8));

var status = todo.GetStatus(dateTimeProviderMock.Object);

Assert.Equal(ToDoStatus.Old, status);

}
```

// </> () <>

== <> {} ==

// </>

// == </>

<> {} == /

// </> () <> {}

● Тесты падают

- ▼ Api
 - ▼ Features/Todos
 - C# TodosController.cs
- ▼ Application
 - ▼ EntityTypeConfigurations
 - C# TodoEntityTypeConfigu...
 - ▼ Features/Todos/Handlers
 - ▼ CreateToDo
 - C# CreateToDoCommand.cs
 - C# CreateToDoHandler.cs
 - C# CreateToDoRequest.cs
 - C# CreateToDoResponse.cs
 - ▼ DeleteToDo
 - C# DeleteToDoCommand.cs
 - C# DeleteToDoHandler.cs
 - C# DeleteToDoResponse.cs
 - ▼ GetTodos
 - C# GetTodosHandler.cs
 - C# GetTodosQuery.cs
 - C# GetTodosQueryTests.cs
 - C# GetTodosResponse.cs
 - C# AppDbContext.cs
 - C# IntegrationTestBaseTest...
- ▼ Core
 - ▼ Features/Todos/Entities
 - C# ToDo.cs M
 - C# ToDoStatus.cs
 - C# ToDoTests.cs
 - C# IDateTimeProvider.cs
- ▼ e2e
 - ≡ to-dos-happy-path.feature

GetStatus

```
if (CreatedAtUtc > utcNow.AddDays(-7)
    && CreatedAtUtc < utcNow)
{
    return ToDoStatus.New;
}
else if (CreatedAtUtc > utcNow.AddDays(-28)
    && CreatedAtUtc <= utcNow.AddDays(-7))
{
    return ToDoStatus.Old;
}
```

● Тесты проходят

// </> () <>

== <> {} ==

// </>

// == </>

<> {} == /

// </> () <> {}

```
public async Task GetStatus_FiveDaysAgoIsNew()
{
    var dt = new DateTime(2026, 01, 02, 03, 04, 05, 678, DateTimeKind.Utc);

    var todo = new Todo { Name = "To Do", CreatedAtUtc = dt };

    var dateTimeProviderMock = new Mock<IDateTimeProvider>();

    dateTimeProviderMock.Setup(x => x.UtcNow).Returns(dt.AddDays(5));

    var status = todo.GetStatus(dateTimeProviderMock.Object);

    Assert.Equal(TodoStatus.New, status);
}
```

```
public async Task GetStatus_EightDaysAgoIsOld()
{
    var dt = new DateTime(2026, 01, 02, 03, 04, 05, 678, DateTimeKind.Utc);

    var todo = new Todo { Name = "To Do", CreatedAtUtc = dt };

    var dateTimeProviderMock = new Mock<IDateTimeProvider>();

    dateTimeProviderMock.Setup(x => x.UtcNow).Returns(dt.AddDays(8));

    var status = todo.GetStatus(dateTimeProviderMock.Object);

    Assert.Equal(TodoStatus.Old, status);
}
```

```

public async Task GetStatus_FiveDaysAgoIsNew()
public async Task GetStatus_EightDaysAgoIsOld()
{
    var dt = new DateTime(2026, 01, 02, 03, 04, 05, 678, DateTimeKind.Utc);

    var todo = new Todo { Name = "To Do", CreatedAtUtc = dt };

    var dateTimeProviderMock = new Mock<IDateTimeProvider>();

    dateTimeProviderMock.Setup(x => x.UtcNow).Returns(dt.AddDays(5));
    dateTimeProviderMock.Setup(x => x.UtcNow).Returns(dt.AddDays(8));

    var status = todo.GetStatus(dateTimeProviderMock.Object);

    Assert.Equal(TodoStatus.New, status);
    Assert.Equal(TodoStatus.Old, status);
}

```

- ▼ Api
 - ▼ Features/Todos
 - C# TodosController.cs
- ▼ Application
 - ▼ EntityTypeConfigurations
 - C# ToDoEntityTypeConfigu...
 - ▼ Features/Todos/Handlers
 - ▼ CreateToDo
 - C# CreateToDoCommand.cs
 - C# CreateToDoHandler.cs
 - C# CreateToDoRequest.cs
 - C# CreateToDoResponse.cs
 - ▼ DeleteToDo
 - C# DeleteToDoCommand.cs
 - C# DeleteToDoHandler.cs
 - C# DeleteToDoResponse.cs
 - ▼ GetTodos
 - C# GetTodosHandler.cs
 - C# GetTodosQuery.cs
 - C# GetTodosQueryTests.cs
 - C# GetTodosResponse.cs
 - C# AppDbContext.cs
 - C# IntegrationTestBaseTest...
- ▼ Core
 - ▼ Features/Todos/Entities
 - C# ToDo.cs
 - C# ToDoStatus.cs
 - C# ToDoTests.cs M
 - C# IDateTimeProvider.cs
- ▼ e2e
 - ≡ to-dos-happy-path.feature

```

public static IEnumerable<object[]> GetStatusTestCases =>
    [
        [
            new DateTime(
                2026, 01, 07, 03, 04, 05, 678, DateTimeKind.Utc
            ),
            ToDoStatus.New,
        ],
        [
            new DateTime(
                2026, 01, 10, 03, 04, 05, 678, DateTimeKind.Utc
            ),
            ToDoStatus.Old,
        ],
    ]
  
```

- Api
 - Features/Todos
 - ToDoController.cs
- Application
 - EntityTypeConfigurations
 - ToDoEntityTypeConfigu...
 - Features/Todos/Handlers
 - CreateToDo
 - CreateToDoCommand.cs
 - CreateToDoHandler.cs
 - CreateToDoRequest.cs
 - CreateToDoResponse.cs
 - DeleteToDo
 - DeleteToDoCommand.cs
 - DeleteToDoHandler.cs
 - DeleteToDoResponse.cs
 - GetTodos
 - GetTodosHandler.cs
 - GetTodosQuery.cs
 - GetTodosQueryTests.cs
 - GetTodosResponse.cs
 - AppDbContext.cs
 - IntegrationTestBaseTest...
- Core
 - Features/Todos/Entities
 - ToDo.cs
 - ToDoStatus.cs
 - ToDoTests.cs
 - IDateTimeProvider.cs
- e2e
 - to-dos-happy-path.feature

```
[  
    new DateTime(  
        2026, 01, 30, 03, 04, 05, 678, DateTimeKind.Utc  
    ),  
    ToDoStatus.Forgotten,  
],  
];
```

- ▼ Api
 - ▼ Features/Todos
 - C# TodosController.cs
- ▼ Application
 - ▼ EntityTypeConfigurations
 - C# TodoEntityTypeConfigu...
 - ▼ Features/Todos/Handlers
 - ▼ CreateToDo
 - C# CreateToDoCommand.cs
 - C# CreateToDoHandler.cs
 - C# CreateToDoRequest.cs
 - C# CreateToDoResponse.cs
 - ▼ DeleteToDo
 - C# DeleteToDoCommand.cs
 - C# DeleteToDoHandler.cs
 - C# DeleteToDoResponse.cs
 - ▼ GetTodos
 - C# GetTodosHandler.cs
 - C# GetTodosQuery.cs
 - C# GetTodosQueryTests.cs
 - C# GetTodosResponse.cs
 - C# AppDbContext.cs
 - C# IntegrationTestBaseTest...
- ▼ Core
 - ▼ Features/Todos/Entities
 - C# ToDo.cs
 - C# ToDoStatus.cs
 - C# ToDoTests.cs M
 - C# IDateTimeProvider.cs
- ▼ e2e
 - ≡ to-dos-happy-path.feature

[Theory]

[MemberData(nameof(GetStatusTestCases))]

```
public void GetStatus(
    DateTime utcNow,
    ToDoStatus expectedStatus
)
{
    var dt = new DateTime(
        2026, 01, 02, 03, 04, 05, 678, DateTimeKind.Utc
    );

    var todo = new ToDo {
        Name = "To Do",
        CreatedAtUtc = dt,
    };
}
```

- Api
 - Features/Todos
 - C# TodosController.cs
- Application
 - EntityTypeConfigurations
 - C# ToDoEntityTypeConfigu...
 - Features/Todos/Handlers
 - CreateToDo
 - C# CreateToDoCommand.cs
 - C# CreateToDoHandler.cs
 - C# CreateToDoRequest.cs
 - C# CreateToDoResponse.cs
 - DeleteToDo
 - C# DeleteToDoCommand.cs
 - C# DeleteToDoHandler.cs
 - C# DeleteToDoResponse.cs
 - GetTodos
 - C# GetTodosHandler.cs
 - C# GetTodosQuery.cs
 - C# GetTodosQueryTests.cs
 - C# GetTodosResponse.cs
 - C# AppDbContext.cs
 - C# IntegrationTestBaseTest...
- Core
 - Features/Todos/Entities
 - C# ToDo.cs
 - C# ToDoStatus.cs
 - C# ToDoTests.cs M
 - C# IDateTimeProvider.cs
- e2e
 - to-dos-happy-path.feature

```
var dateTimeProviderMock = new Mock<IDateTimeProvider>();

dateTimeProviderMock.Setup(x => x.UtcNow)
    .Returns(utcNow);

var status = todo.GetStatus(dateTimeProviderMock.Object);

Assert.Equal(expectedStatus, status);
}
```



Статья про
ЭТИ
ТЕХНИКИ

// </> () <>

== <> {} ==

// </>

// == </>

<> {} == /

// </> () <> {}

● Тесты падают

- ✓ Api
 - ✓ Features/Todos
 - C# TodosController.cs
- ✓ Application
 - ✓ EntityTypeConfigurations
 - C# TodoEntityTypeConfigu...
 - ✓ Features/Todos/Handlers
 - ✓ CreateToDo
 - C# CreateToDoCommand.cs
 - C# CreateToDoHandler.cs
 - C# CreateToDoRequest.cs
 - C# CreateToDoResponse.cs
 - ✓ DeleteToDo
 - C# DeleteToDoCommand.cs
 - C# DeleteToDoHandler.cs
 - C# DeleteToDoResponse.cs
 - ✓ GetTodos
 - C# GetTodosHandler.cs
 - C# GetTodosQuery.cs
 - C# GetTodosQueryTests.cs
 - C# GetTodosResponse.cs
 - C# AppDbContext.cs
 - C# IntegrationTestBaseTest...
- ✓ Core
 - ✓ Features/Todos/Entities
 - C# ToDo.cs M
 - C# ToDoStatus.cs
 - C# ToDoTests.cs
 - C# IDateTimeProvider.cs
- ✓ e2e
 - ≡ to-dos-happy-path.feature

GetStatus

```
if (CreatedAtUtc > utcNow.AddDays(-7)
    && CreatedAtUtc < utcNow)
{
    return ToDoStatus.New;
}
else if (CreatedAtUtc > utcNow.AddDays(-28)
    && CreatedAtUtc <= utcNow.AddDays(-7))
{
    return ToDoStatus.Old;
}
else if (CreatedAtUtc <= utcNow.AddDays(-28))
{
    return ToDoStatus.Forgotten;
}
```

● Тесты проходят

// </> () <>

== <> {} ==

// </>

// == </>

<> {} == /

// </> () <> {}

- ✓ Api
 - ✓ Features/Todos
 - C# TodosController.cs
- ✓ Application
 - ✓ EntityTypeConfigurations
 - C# ToDoEntityTypeConfigu...
 - ✓ Features/Todos/Handlers
 - ✓ CreateToDo
 - C# CreateToDoCommand.cs
 - C# CreateToDoHandler.cs
 - C# CreateToDoRequest.cs
 - C# CreateToDoResponse.cs
 - ✓ DeleteToDo
 - C# DeleteToDoCommand.cs
 - C# DeleteToDoHandler.cs
 - C# DeleteToDoResponse.cs
 - ✓ GetTodos
 - C# GetTodosHandler.cs
 - C# GetTodosQuery.cs
 - C# GetTodosQueryTests.cs
 - C# GetTodosResponse.cs
 - C# AppDbContext.cs
 - C# IntegrationTestBaseTest...
- ✓ Core
 - ✓ Features/Todos/Entities
 - C# ToDo.cs
 - C# ToDoStatus.cs
 - C# ToDoTests.cs
 - C# IDateTimeProvider.cs
- ✓ e2e
 - ≡ to-dos-happy-path.feature

GetStatus

```
if (CreatedAtUtc > utcNow.AddDays(-7)
    && CreatedAtUtc < utcNow)
{
    return ToDoStatus.New;
}
else if (CreatedAtUtc > utcNow.AddDays(-28)
    && CreatedAtUtc <= utcNow.AddDays(-7))
{
    return ToDoStatus.Old;
}
else if (CreatedAtUtc <= utcNow.AddDays(-28))
{
    return ToDoStatus.Forgotten;
}
```



- ▼ Api
 - ▼ Features/Todos
 - C# TodosController.cs
- ▼ Application
 - ▼ EntityTypeConfigurations
 - C# ToDoEntityTypeConfigu...
 - ▼ Features/Todos/Handlers
 - ▼ CreateToDo
 - C# CreateToDoCommand.cs
 - C# CreateToDoHandler.cs
 - C# CreateToDoRequest.cs
 - C# CreateToDoResponse.cs
 - ▼ DeleteToDo
 - C# DeleteToDoCommand.cs
 - C# DeleteToDoHandler.cs
 - C# DeleteToDoResponse.cs
 - ▼ GetTodos
 - C# GetTodosHandler.cs
 - C# GetTodosQuery.cs
 - C# GetTodosQueryTests.cs
 - C# GetTodosResponse.cs
 - C# AppDbContext.cs
 - C# IntegrationTestBaseTest...
- ▼ Core
 - ▼ Features/Todos/Entities
 - C# ToDo.cs M
 - C# ToDoStatus.cs
 - C# ToDoTests.cs
 - C# IDateTimeProvider.cs
- ▼ e2e
 - ≡ to-dos-happy-path.feature

```

public ToDoStatus GetStatus(
    IDateTimeProvider dateTimeProvider
) {
    const int newForDays = 7;
    const int forgottenAfterDays = 28;

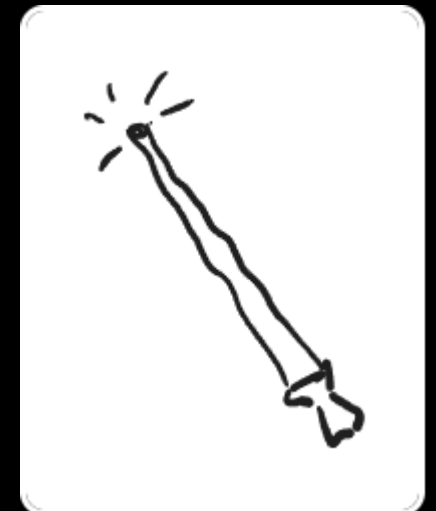
    var ageInDays = (dateTimeProvider.UtcNow - CreatedAtUtc)
        .TotalDays;

    if (ageInDays < newForDays)
        return ToDoStatus.New;

    if (ageInDays < forgottenAfterDays)
        return ToDoStatus.Old;

    return ToDoStatus.Forgotten;
}

```



● Тесты всё ещё проходят

Техники тест-дизайна



Шеф, почему тестов 3?



Минимально необходимое количество

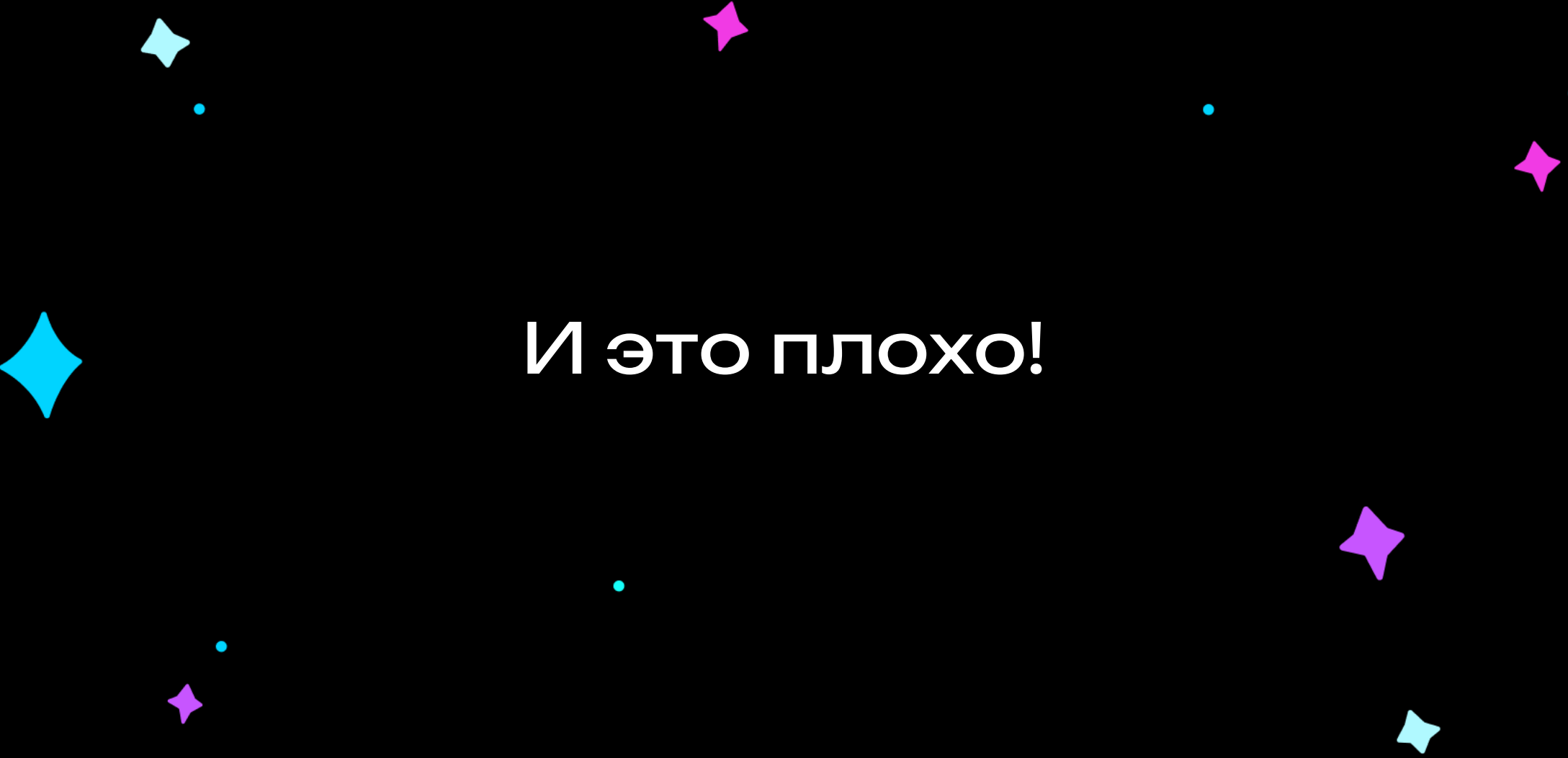
А сколько нужно?



- ▼ Api
 - ▼ Features/Todos
 - 🔗 TodosController.cs
- ▼ Application
 - ▼ EntityTypeConfigurations
 - 🔗 TodoEntityTypeConfigu...
 - ▼ Features/Todos/Handlers
 - ▼ CreateToDo
 - 🔗 CreateToDoCommand.cs
 - 🔗 CreateToDoHandler.cs
 - 🔗 CreateToDoRequest.cs
 - 🔗 CreateToDoResponse.cs
 - ▼ DeleteToDo
 - 🔗 DeleteToDoCommand.cs
 - 🔗 DeleteToDoHandler.cs
 - 🔗 DeleteToDoResponse.cs
 - ▼ GetTodos
 - 🔗 GetTodosHandler.cs
 - 🔗 GetTodosQuery.cs
 - 🔗 GetTodosQueryTests.cs
 - 🔗 GetTodosResponse.cs
 - 🔗 AppDbContext.cs
 - 🔗 DateTimeProvider.cs U
 - 🔗 IntegrationTestBaseTest...
 - ▼ Core
 - ▼ Features/Todos/Entities
 - 🔗 ToDo.cs
 - 🔗 ToDoStatus.cs
 - 🔗 ToDoTests.cs
 - 🔗 IDateTimeProvider.cs
 - ▼ e2e
 - ≡ to-dos-happy-path.feature

```
public class DateTimeProvider : IDateTimeProvider
{
    public DateTime.UtcNow => DateTime.UtcNow;
}
```

● E2E-тесты проходят

The background is black and decorated with several stylized stars and dots. There are four small cyan stars, four small magenta stars, and four small cyan dots scattered across the frame. A large cyan four-pointed star is positioned on the left side of the image.

И это плохо!

- ✓ Api
 - ✓ Features/Todos
 - TodosController.cs
- ✓ Application
 - ✓ EntityTypeConfigurations
 - TodoEntityTypeConfigu...
 - ✓ Features/Todos/Handlers
 - ✓ CreateToDo
 - CreateToDoCommand.cs
 - CreateToDoHandler.cs
 - CreateToDoRequest.cs
 - CreateToDoResponse.cs
 - ✓ DeleteToDo
 - DeleteToDoCommand.cs
 - DeleteToDoHandler.cs
 - DeleteToDoResponse.cs
 - ✓ GetTodos
 - GetTodosHandler.cs
 - GetTodosQuery.cs
 - GetTodosQueryTests.cs
 - GetTodosResponse.cs
 - AppDbContext.cs
 - DateTimeProvider.cs
 - IntegrationTestBaseTest...
- ✓ Core
 - ✓ Features/Todos/Entities
 - ToDo.cs
 - ToDoStatus.cs
 - ToDoTests.cs
 - IDateTimeProvider.cs
- ✓ e2e
 - ≡ to-dos-happy-path.feature

```
return new GetTodosResponse
{
    Todos = todos
        .Select(x => new ToDoDto
        {
            Id = x.Id,
            Name = x.Name,
            Status = ToDoStatus.New,
        })
        .ToList(),
};
```

Надо было всё
ломать по TDD
через None!

- ✓ Api
 - ✓ Features/Todos
 - ☛ TodosController.cs
- ✓ Application
 - ✓ EntityTypeConfigurations
 - ☛ ToDoEntityTypeConfigu...
 - ✓ Features/Todos/Handlers
 - ✓ CreateToDo
 - ☛ CreateToDoCommand.cs
 - ☛ CreateToDoHandler.cs
 - ☛ CreateToDoRequest.cs
 - ☛ CreateToDoResponse.cs
 - ✓ DeleteToDo
 - ☛ DeleteToDoCommand.cs
 - ☛ DeleteToDoHandler.cs
 - ☛ DeleteToDoResponse.cs
 - ✓ GetTodos
 - ☛ GetTodosHandler.cs M
 - ☛ GetTodosQuery.cs
 - ☛ GetTodosQueryTests.cs
 - ☛ GetTodosResponse.cs
 - ☛ AppDbContext.cs
 - ☛ DateTimeProvider.cs
 - ☛ IntegrationTestBaseTest...
- ✓ Core
 - ✓ Features/Todos/Entities
 - ☛ ToDo.cs
 - ☛ ToDoStatus.cs
 - ☛ ToDoTests.cs
 - ☛ IDateTimeProvider.cs
- ✓ e2e
 - ≡ to-dos-happy-path.feature

```
return new GetTodosResponse
{
    Todos = todos
        .Select(x => new ToDoDto
        {
            Id = x.Id,
            Name = x.Name,
            Status = x.GetStatus(dateTimeProvider),
        })
        .ToList(),
};
```

// </> () <>

== <> { } ==

// </>

// == </>

<> { } == /

// </> () <> { }

● E2E-тесты падают

И это хорошо!

Karate вывод

```
{"id":80,"name":"[API-E2E]-to-do-0.0021082630084587883","status":"Forgotten"}
```

```
{"id":80,"name":"[API-E2E]-to-do-0.0021082630084587883","status":"New"}
```

```
$(0).status | not equal (STRING:STRING)
```

```
'Forgotten'
```

```
'New'
```

Не понятное

```
{  
  "id": 80,  
  "name": "[API-E2E]-to-do-0.0021082630084587883",  
  "status": "Forgotten"  
}
```

Что внутри?

CreatedAtUtc содержит DateTime {01/01/0001 00:00:00}

В базе created_at_utc timestamp with timezone

А там лежит -infinity

- ▼ Api
 - ▼ Features/Todos
 - 📄 TodosController.cs
- ▼ Application
 - ▼ EntityTypeConfigurations
 - 📄 ToDoEntityTypeConfigu...
 - ▼ Features/Todos/Handlers
 - ▼ CreateToDo
 - 📄 CreateToDoCommand.cs
 - 📄 CreateToDoHandler.cs
 - 📄 CreateToDoRequest.cs
 - 📄 CreateToDoResponse.cs
 - ▼ DeleteToDo
 - 📄 DeleteToDoCommand.cs
 - 📄 DeleteToDoHandler.cs
 - 📄 DeleteToDoResponse.cs
 - ▼ GetTodos
 - 📄 GetTodosHandler.cs
 - 📄 GetTodosQuery.cs
 - 📄 GetTodosQueryTests.cs
 - 📄 GetTodosResponse.cs
 - 📄 AppDbContext.cs
 - 📄 DateTimeProvider.cs
 - 📄 IntegrationTestBaseTest...
- ▼ Core
 - ▼ Features/Todos/Entities
 - 📄 ToDo.cs
 - 📄 ToDoStatus.cs
 - 📄 ToDoTests.cs
 - 📄 IDateTimeProvider.cs
- ▼ e2e
 - ≡ to-dos-happy-path.feature

```

public class CreateToDoCommand(AppDbContext context)
{
    public async Task<long> ExecuteAsync(
        CreateToDoRequest createToDoRequest
    )
    {
        var newToDo = new ToDo {
            Name = createToDoRequest.Name,
        };

        await context.Todos.AddAsync(newToDo);

        await context.SaveChangesAsync();

        return newToDo.Id;
    }
}

```

- ✓ Api
 - ✓ Features/Todos
 - 🔗 TodosController.cs
- ✓ Application
 - ✓ EntityTypeConfigurations
 - 🔗 TodoEntityTypeConfigu...
 - ✓ Features/Todos/Handlers
 - ✓ CreateToDo
 - 🔗 CreateToDoCommand.cs
 - 🔗 CreateToDoHandler.cs
 - 🔗 CreateToDoRequest.cs
 - 🔗 CreateToDoResponse.cs
 - ✓ DeleteToDo
 - 🔗 DeleteToDoCommand.cs
 - 🔗 DeleteToDoHandler.cs
 - 🔗 DeleteToDoResponse.cs
 - ✓ GetTodos
 - 🔗 GetTodosHandler.cs
 - 🔗 GetTodosQuery.cs
 - 🔗 GetTodosQueryTests.cs
 - 🔗 GetTodosResponse.cs
 - 🔗 AppDbContext.cs
 - 🔗 DateTimeProvider.cs
 - 🔗 IntegrationTestBaseTest...
- ✓ Core
 - ✓ Features/Todos/Entities
 - 🔗 **ToDo.cs**
 - 🔗 ToDoStatus.cs
 - 🔗 ToDoTests.cs
 - 🔗 IDateTimeProvider.cs
- ✓ e2e
 - ≡ to-dos-happy-path.feature

```
public class ToDo
{
    public ToDo() { }

    public long Id { get; set; }

    public required string Name { get; set; }

    public DateTime CreatedAtUtc { get; set; }
}
```

- ✓ Api
 - ✓ Features/Todos
 - 🔗 TodosController.cs
- ✓ Application
 - ✓ EntityTypeConfigurations
 - 🔗 ToDoEntityTypeConfigu...
 - ✓ Features/Todos/Handlers
 - ✓ CreateToDo
 - 🔗 CreateToDoCommand.cs
 - 🔗 CreateToDoHandler.cs
 - 🔗 CreateToDoRequest.cs
 - 🔗 CreateToDoResponse.cs
 - ✓ DeleteToDo
 - 🔗 DeleteToDoCommand.cs
 - 🔗 DeleteToDoHandler.cs
 - 🔗 DeleteToDoResponse.cs
 - ✓ GetTodos
 - 🔗 GetTodosHandler.cs
 - 🔗 GetTodosQuery.cs
 - 🔗 GetTodosQueryTests.cs
 - 🔗 GetTodosResponse.cs
 - 🔗 AppDbContext.cs
 - 🔗 DateTimeProvider.cs
 - 🔗 IntegrationTestBaseTest...
- ✓ Core
 - ✓ Features/Todos/Entities
 - 🔗 **ToDo.cs** M
 - 🔗 ToDoStatus.cs
 - 🔗 ToDoTests.cs
 - 🔗 IDateTimeProvider.cs
- ✓ e2e
 - ≡ to-dos-happy-path.feature

```
public class ToDo
{
    public ToDo() { }

    public long Id { get; set; }

    public required string Name { get; set; }

    public required DateTime CreatedAtUtc { get; set; }
}
```

- ▼ Api
 - ▼ Features/Todos
 - 🔗 TodosController.cs
- ▼ Application
 - ▼ EntityTypeConfigurations
 - 🔗 ToDoEntityTypeConfigu...
 - ▼ Features/Todos/Handlers
 - ▼ CreateToDo
 - 🔗 CreateToDoCommand.cs
 - 🔗 CreateToDoHandler.cs
 - 🔗 CreateToDoRequest.cs
 - 🔗 CreateToDoResponse.cs
 - ▼ DeleteToDo
 - 🔗 DeleteToDoCommand.cs
 - 🔗 DeleteToDoHandler.cs
 - 🔗 DeleteToDoResponse.cs
 - ▼ GetTodos
 - 🔗 GetTodosHandler.cs
 - 🔗 GetTodosQuery.cs
 - 🔗 GetTodosQueryTests.cs
 - 🔗 GetTodosResponse.cs
 - 🔗 AppDbContext.cs
 - 🔗 DateTimeProvider.cs
 - 🔗 IntegrationTestBaseTest...
- ▼ Core
 - ▼ Features/Todos/Entities
 - 🔗 ToDo.cs
 - 🔗 ToDoStatus.cs
 - 🔗 ToDoTests.cs
 - 🔗 IDateTimeProvider.cs
- ▼ e2e
 - ≡ to-dos-happy-path.feature

```
public async Task<long> ExecuteAsync(  
    CreateToDoRequest createToDoRequest  
)  
{  
    var newToDo = new ToDo  
    {  
        Name = createToDoRequest.Name,  
    };  
  
    await context.Todos.AddAsync(newToDo);  
  
    await context.SaveChangesAsync();  
  
    return newToDo.Id;  
}
```

- ▼ Api
 - ▼ Features/Todos
 - 🔗 TodosController.cs
- ▼ Application
 - ▼ EntityTypeConfigurations
 - 🔗 ToDoEntityTypeConfigu...
 - ▼ Features/Todos/Handlers
 - ▼ CreateToDo
 - 🔗 CreateToDoCommand.cs M
 - 🔗 CreateToDoHandler.cs
 - 🔗 CreateToDoRequest.cs
 - 🔗 CreateToDoResponse.cs
 - ▼ DeleteToDo
 - 🔗 DeleteToDoCommand.cs
 - 🔗 DeleteToDoHandler.cs
 - 🔗 DeleteToDoResponse.cs
 - ▼ GetTodos
 - 🔗 GetTodosHandler.cs
 - 🔗 GetTodosQuery.cs
 - 🔗 GetTodosQueryTests.cs
 - 🔗 GetTodosResponse.cs
 - 🔗 AppDbContext.cs
 - 🔗 DateTimeProvider.cs
 - 🔗 IntegrationTestBaseTest...
- ▼ Core
 - ▼ Features/Todos/Entities
 - 🔗 ToDo.cs
 - 🔗 ToDoStatus.cs
 - 🔗 ToDoTests.cs
 - 🔗 IDateTimeProvider.cs
- ▼ e2e
 - ≡ to-dos-happy-path.feature

```

public async Task<long> ExecuteAsync(
    CreateToDoRequest createToDoRequest
)
{
    var newToDo = new ToDo
    {
        Name = createToDoRequest.Name,
        CreatedAtUtc = dateTimeProvider.UtcNow,
    };

    await context.Todos.AddAsync(newToDo);

    await context.SaveChangesAsync();

    return newToDo.Id;
}

```

- Api
 - Features/Todos
 - ToDoController.cs
- Application
 - EntityTypeConfigurations
 - ToDoEntityTypeConfigu...
 - Features/Todos/Handlers
 - CreateToDo
 - CreateToDoCommand.cs
 - CreateToDoHandler.cs
 - CreateToDoRequest.cs
 - CreateToDoResponse.cs
 - DeleteToDo
 - DeleteToDoCommand.cs
 - DeleteToDoHandler.cs
 - DeleteToDoResponse.cs
 - GetTodos
 - GetTodosHandler.cs
 - GetTodosQuery.cs
 - GetTodosQueryTests.csm**
 - GetTodosResponse.cs
 - AppDbContext.cs
 - DateTimeProvider.cs
 - IntegrationTestBaseTest...
- Core
 - Features/Todos/Entities
 - ToDo.cs
 - ToDoStatus.cs
 - ToDoTests.cs
 - IDateTimeProvider.cs
- e2e
 - to-dos-happy-path.feature

GetAsync_ShouldOrderNewerTodosFirst

```
var dt = new DateTime(  
    2026, 01, 02, 03, 04, 05, 678, DateTimeKind.Utc  
);
```

```
var firstToDo = new ToDo  
{  
    Name = "First",  
    CreatedAtUtc = dt.AddHours(1),  
};
```

```
await AddEntityAndSaveAsync(context, firstToDo);
```

- ✓ Api
 - ✓ Features/Todos
 - 🔗 TodosController.cs
- ✓ Application
 - ✓ EntityTypeConfigurations
 - 🔗 TodoEntityTypeConfigu...
 - ✓ Features/Todos/Handlers
 - ✓ CreateToDo
 - 🔗 CreateToDoCommand.cs
 - 🔗 CreateToDoHandler.cs
 - 🔗 CreateToDoRequest.cs
 - 🔗 CreateToDoResponse.cs
 - ✓ DeleteToDo
 - 🔗 DeleteToDoCommand.cs
 - 🔗 DeleteToDoHandler.cs
 - 🔗 DeleteToDoResponse.cs
 - ✓ GetTodos
 - 🔗 GetTodosHandler.cs
 - 🔗 GetTodosQuery.cs
 - 🔗 GetTodosQueryTests.csm
 - 🔗 GetTodosResponse.cs
 - 🔗 AppDbContext.cs
 - 🔗 DateTimeProvider.cs
 - 🔗 IntegrationTestBaseTest...
- ✓ Core
 - ✓ Features/Todos/Entities
 - 🔗 ToDo.cs
 - 🔗 ToDoStatus.cs
 - 🔗 ToDoTests.cs
 - 🔗 IDateTimeProvider.cs
- ✓ e2e
 - ≡ to-dos-happy-path.feature

GetAsync_ShouldOrderNewerTodosFirst

```
var secondToDo = new ToDo
{
    Name = "Second",
    CreatedAtUtc = dt.AddHours(2),
};
await AddEntityAndSaveAsync(context, secondToDo);
```

- ▼ Api
 - ▼ Features/Todos
 - 🔗 TodosController.cs
- ▼ Application
 - ▼ EntityTypeConfigurations
 - 🔗 TodoEntityTypeConfigu...
 - ▼ Features/Todos/Handlers
 - ▼ CreateToDo
 - 🔗 CreateToDoCommand.cs
 - 🔗 CreateToDoHandler.cs
 - 🔗 CreateToDoRequest.cs
 - 🔗 CreateToDoResponse.cs
 - ▼ DeleteToDo
 - 🔗 DeleteToDoCommand.cs
 - 🔗 DeleteToDoHandler.cs
 - 🔗 DeleteToDoResponse.cs
 - ▼ GetTodos
 - 🔗 GetTodosHandler.cs
 - 🔗 GetTodosQuery.cs
 - 🔗 GetTodosQueryTests.csm
 - 🔗 GetTodosResponse.cs
 - 🔗 AppDbContext.cs
 - 🔗 DateTimeProvider.cs
 - 🔗 IntegrationTestBaseTest...
- ▼ Core
 - ▼ Features/Todos/Entities
 - 🔗 ToDo.cs
 - 🔗 ToDoStatus.cs
 - 🔗 ToDoTests.cs
 - 🔗 IDateTimeProvider.cs
- ▼ e2e
 - ≡ to-dos-happy-path.feature

GetAsync_ShouldOrderNewerTodosFirst

```
var thirdToDo = new ToDo
{
    Name = "Third",
    CreatedAtUtc = dt.AddHours(3),
};
await AddEntityAndSaveAsync(context, thirdToDo);
```

● Тесты проходят

// </> () <>

== <> {} ==

// </>

// == </>

<> {} == /

// </> () <> {}

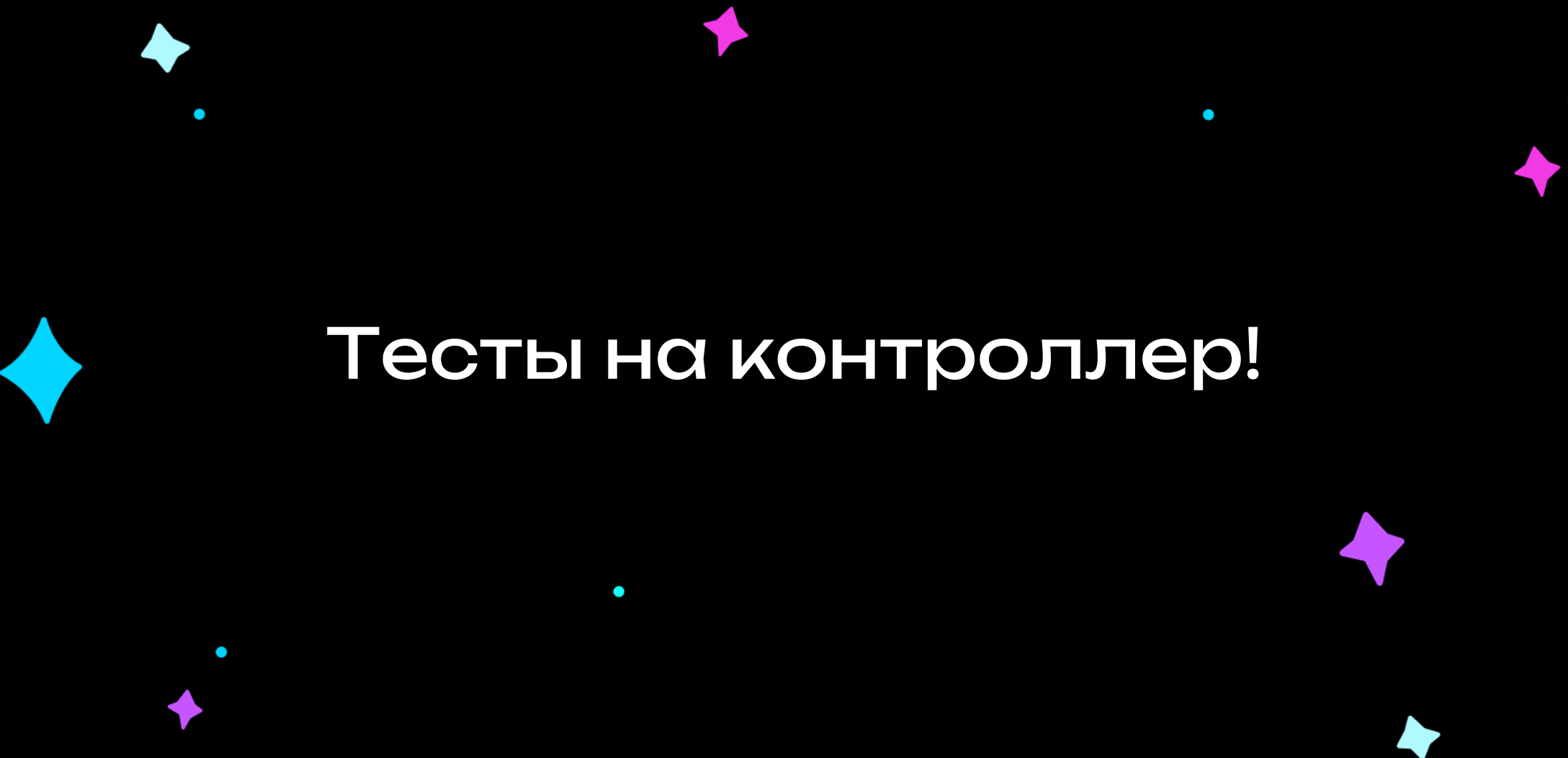
- ▼ Api
 - ▼ Features/Todos
 - 🔗 TodosController.cs
- ▼ Application
 - ▼ EntityTypeConfigurations
 - 🔗 ToDoEntityTypeConfigu...
 - ▼ Features/Todos/Handlers
 - ▼ CreateToDo
 - 🔗 CreateToDoCommand.cs
 - 🔗 CreateToDoHandler.cs
 - 🔗 CreateToDoRequest.cs
 - 🔗 CreateToDoResponse.cs
 - ▼ DeleteToDo
 - 🔗 DeleteToDoCommand.cs
 - 🔗 DeleteToDoHandler.cs
 - 🔗 DeleteToDoResponse.cs
 - ▼ GetTodos
 - 🔗 GetTodosHandler.cs
 - 🔗 GetTodosQuery.cs
 - 🔗 GetTodosQueryTests.cs
 - 🔗 GetTodosResponse.cs
 - 🔗 AppDbContext.cs
 - 🔗 DateTimeProvider.cs
 - 🔗 IntegrationTestBaseTest...
- ▼ Core
 - ▼ Features/Todos/Entities
 - 🔗 ToDo.cs
 - 🔗 ToDoStatus.cs
 - 🔗 ToDoTests.cs
 - 🔗 IDateTimeProvider.cs
- ▼ e2e
 - ≡ to-dos-happy-path.feature

```
public class GetTodosQuery(AppDbContext context)
{
    public Task<List<ToDo>> GetAsync()
    {
        return context
            .QueryableAsNoTracking<ToDo>()
            .OrderByDescending(x => x.Id)
            .ToListAsync();
    }
}
```

- ▼ Api
 - ▼ Features/Todos
 - 🔗 TodosController.cs
- ▼ Application
 - ▼ EntityTypeConfigurations
 - 🔗 TodoEntityTypeConfigu...
 - ▼ Features/Todos/Handlers
 - ▼ CreateToDo
 - 🔗 CreateToDoCommand.cs
 - 🔗 CreateToDoHandler.cs
 - 🔗 CreateToDoRequest.cs
 - 🔗 CreateToDoResponse.cs
 - ▼ DeleteToDo
 - 🔗 DeleteToDoCommand.cs
 - 🔗 DeleteToDoHandler.cs
 - 🔗 DeleteToDoResponse.cs
 - ▼ GetTodos
 - 🔗 GetTodosHandler.cs
 - 🔗 GetTodosQuery.cs M
 - 🔗 GetTodosQueryTests.cs
 - 🔗 GetTodosResponse.cs
 - 🔗 AppDbContext.cs
 - 🔗 DateTimeProvider.cs
 - 🔗 IntegrationTestBaseTest...
- ▼ Core
 - ▼ Features/Todos/Entities
 - 🔗 ToDo.cs
 - 🔗 ToDoStatus.cs
 - 🔗 ToDoTests.cs
 - 🔗 IDateTimeProvider.cs
- ▼ e2e
 - ≡ to-dos-happy-path.feature

```
public class GetTodosQuery(AppDbContext context)
{
    public Task<List<ToDo>> GetAsync()
    {
        return context
            .QueryableAsNoTracking<ToDo>()
            .OrderByDescending(x => x.CreatedAtUtc)
            .ToListAsync();
    }
}
```

● Тесты всё ещё проходят

The background is black and decorated with several stylized stars and dots. There are four light blue stars of varying sizes, four pink stars, and several small light blue dots scattered across the slide. A large, bright blue four-pointed star is positioned to the left of the title.

Тесты на контроллер!

- ▼ Api
 - ▼ Features/Todos
 - 🔗 TodosController.cs
- ▼ Application
 - ▼ EntityTypeConfigurations
 - 🔗 ToDoEntityTypeConfigu...
 - ▼ Features/Todos/Handlers
 - ▼ CreateToDo
 - 🔗 CreateToDoCommand.cs
 - 🔗 CreateToDoHandler.cs
 - 🔗 CreateToDoRequest.cs
 - 🔗 CreateToDoResponse.cs
 - ▼ DeleteToDo
 - 🔗 DeleteToDoCommand.cs
 - 🔗 DeleteToDoHandler.cs
 - 🔗 DeleteToDoResponse.cs
 - ▼ GetTodos
 - 🔗 GetTodosHandler.cs
 - 🔗 GetTodosQuery.cs
 - 🔗 GetTodosQueryTests.cs
 - 🔗 GetTodosResponse.cs
 - 🔗 AppDbContext.cs
 - 🔗 DateTimeProvider.cs
 - 🔗 IntegrationTestBaseTest...
- ▼ Core
 - ▼ Features/Todos/Entities
 - 🔗 ToDo.cs
 - 🔗 ToDoStatus.cs
 - 🔗 ToDoTests.cs
 - 🔗 IDateTimeProvider.cs
- ▼ e2e
 - ≡ to-dos-happy-path.feature

```
public class ToDoEntityTypeConfiguration
    : IEntityTypeConfiguration<ToDo>
{
    public void Configure(EntityTypeBuilder<ToDo> builder)
    {
        builder.HasIndex(x => x.Id).IsUnique();

        builder.Property(x => x.Name).IsRequired()
            .HasMaxLength(256);
    }
}
```

- ▼ Api
 - ▼ Features/Todos
 - C# TodosController.cs
 - C# TodosControllerValida.ü.
- ▼ Application
 - ▼ EntityTypeConfigurations
 - C# ToDoEntityTypeConfigu...
 - ▼ Features/Todos/Handlers
 - ▼ CreateToDo
 - C# CreateToDoCommand.cs
 - C# CreateToDoHandler.cs
 - C# CreateToDoRequest.cs
 - C# CreateToDoResponse.cs
 - ▼ DeleteToDo
 - C# DeleteToDoCommand.cs
 - C# DeleteToDoHandler.cs
 - C# DeleteToDoResponse.cs
 - ▼ GetTodos
 - C# GetTodosHandler.cs
 - C# GetTodosQuery.cs
 - C# GetTodosQueryTests.cs
 - C# GetTodosResponse.cs
 - C# AppDbContext.cs
 - C# DateTimeProvider.cs
 - C# IntegrationTestBaseTest...
- ▼ Core
 - ▼ Features/Todos/Entities
 - C# ToDo.cs
 - C# ToDoStatus.cs
 - C# ToDoTests.cs
 - C# IDateTimeProvider.cs
- ▼ e2e
 - ≡ to-dos-happy-path.feature

```
public static IEnumerable<object[]>
  CreateToDoRequestInvalidNameTestCases =>
  [
    [null!],
    [""],
    [new string('x', 257)],
  ];
```

- ▼ Api
 - ▼ Features/Todos
 - C# TodosController.cs
 - C# TodosControllerValida.u.
- ▼ Application
 - ▼ EntityTypeConfigurations
 - C# ToDoEntityTypeConfigu...
 - ▼ Features/Todos/Handlers
 - ▼ CreateToDo
 - C# CreateToDoCommand.cs
 - C# CreateToDoHandler.cs
 - C# CreateToDoRequest.cs
 - C# CreateToDoResponse.cs
 - ▼ DeleteToDo
 - C# DeleteToDoCommand.cs
 - C# DeleteToDoHandler.cs
 - C# DeleteToDoResponse.cs
 - ▼ GetTodos
 - C# GetTodosHandler.cs
 - C# GetTodosQuery.cs
 - C# GetTodosQueryTests.cs
 - C# GetTodosResponse.cs
 - C# AppDbContext.cs
 - C# DateTimeProvider.cs
 - C# IntegrationTestBaseTest...
- ▼ Core
 - ▼ Features/Todos/Entities
 - C# ToDo.cs
 - C# ToDoStatus.cs
 - C# ToDoTests.cs
 - C# IDateTimeProvider.cs
- ▼ e2e
 - ≡ to-dos-happy-path.feature

```

public class TodosControllerValidationTests(
    WebApplicationFactory<Program> factory
) : ControllerValidationTestsBase(factory)
{
    [Theory]
    [MemberData(nameof(
        CreateToDoRequestInvalidNameTestCases
    ))]
    public async Task
        CreateToDoAsync_ShouldThrowBadRequestIfNameIsInvalid(
        string name
    )
    {

```

- ▼ Api
 - ▼ Features/Todos
 - C# TodosController.cs
 - C# TodosControllerValida.U.
- ▼ Application
 - ▼ EntityTypeConfigurations
 - C# ToDoEntityTypeConfigu...
 - ▼ Features/Todos/Handlers
 - ▼ CreateToDo
 - C# CreateToDoCommand.cs
 - C# CreateToDoHandler.cs
 - C# CreateToDoRequest.cs
 - C# CreateToDoResponse.cs
 - ▼ DeleteToDo
 - C# DeleteToDoCommand.cs
 - C# DeleteToDoHandler.cs
 - C# DeleteToDoResponse.cs
 - ▼ GetTodos
 - C# GetTodosHandler.cs
 - C# GetTodosQuery.cs
 - C# GetTodosQueryTests.cs
 - C# GetTodosResponse.cs
 - C# AppDbContext.cs
 - C# DateTimeProvider.cs
 - C# IntegrationTestBaseTest...
- ▼ Core
 - ▼ Features/Todos/Entities
 - C# ToDo.cs
 - C# ToDoStatus.cs
 - C# ToDoTests.cs
 - C# IDateTimeProvider.cs
- ▼ e2e
 - ≡ to-dos-happy-path.feature

```

var createToDoRequest = new CreateToDoRequest {
    Name = name,
};

var response = await _httpClient.PostAsJsonAsync(
    "/to-dos",
    createToDoRequest
);

Assert.NotNull(response);
Assert.Equal(
    StatusCodes.Status400BadRequest,
    (int)response.StatusCode
);
}

```

- ▼ Api
 - ▼ Features/Todos
 - ⚡ # TodosController.cs
 - ⚡ # TodosControllerValida...
 - ⚡ # ControllerValidationTes...
- ▼ Application
 - ▼ EntityTypeConfigurations
 - ⚡ # ToDoEntityTypeConfigu...
 - ▼ Features/Todos/Handlers
 - ▼ CreateToDo
 - ⚡ # CreateToDoCommand.cs
 - ⚡ # CreateToDoHandler.cs
 - ⚡ # CreateToDoRequest.cs
 - ⚡ # CreateToDoResponse.cs
 - ▼ DeleteToDo
 - ⚡ # DeleteToDoCommand.cs
 - ⚡ # DeleteToDoHandler.cs
 - ⚡ # DeleteToDoResponse.cs
 - ▼ GetTodos
 - ⚡ # GetTodosHandler.cs
 - ⚡ # GetTodosQuery.cs
 - ⚡ # GetTodosQueryTests.cs
 - ⚡ # GetTodosResponse.cs
 - ⚡ # AppDbContext.cs
 - ⚡ # DateTimeProvider.cs
 - ⚡ # IntegrationTestBaseTest...
- ▼ Core
 - ▼ Features/Todos/Entities
 - ⚡ # ToDo.cs
 - ⚡ # ToDoStatus.cs
 - ⚡ # ToDoTests.cs
 - ⚡ # IDateTimeProvider.cs
- ▼ e2e
 - ≡ to-dos-happy-path.feature

```

public class ControllerValidationTestsBase(
    WebApplicationFactory<Program> factory
) : IClassFixture<WebApplicationFactory<Program>>,
    IAsyncLifetime
{
    protected HttpClient _httpClient = null!;

    public async Task InitializeAsync()
    {
        _httpClient = factory.CreateClient();

        factory = factory.WithWebHostBuilder(builder =>
        {

```

- Api
 - Features/Todos
 - ToDoController.cs
 - ToDoControllerValida...
 - ControllerValidationTes...**
- Application
 - EntityTypeConfigurations
 - ToDoEntityTypeConfigu...
 - Features/Todos/Handlers
 - CreateToDo
 - CreateToDoCommand.cs
 - CreateToDoHandler.cs
 - CreateToDoRequest.cs
 - CreateToDoResponse.cs
 - DeleteToDo
 - DeleteToDoCommand.cs
 - DeleteToDoHandler.cs
 - DeleteToDoResponse.cs
 - GetTodos
 - GetTodosHandler.cs
 - GetTodosQuery.cs
 - GetTodosQueryTests.cs
 - GetTodosResponse.cs
 - AppDbContext.cs
 - DateTimeProvider.cs
 - IntegrationTestBaseTest...
- Core
 - Features/Todos/Entities
 - ToDo.cs
 - ToDoStatus.cs
 - ToDoTests.cs
 - IDateTimeProvider.cs
- e2e
 - to-dos-happy-path.feature

```

services.Remove(
    services.Single(x =>
        x.ServiceType
        == typeof(
            IDbContextOptionsConfiguration<AppDbContext>
        )
    )
);

services.AddDbContext<AppDbContext>(options =>
    options.UseInMemoryDatabase(
        databaseName: new Random().Next().ToString()
    )
);

```

.NET красавка!

// </> () <>

== <> {} ==

// </>

// == </>

<> {} == /

// </> () <> {}

● Тесты падают

- ▼ Api
 - ▼ Features/Todos
 - C# TodosController.cs
 - C# TodosControllerValida...
 - C# ControllerValidationTes...
- ▼ Application
 - ▼ EntityTypeConfigurations
 - C# ToDoEntityTypeConfigu...
 - ▼ Features/Todos/Handlers
 - ▼ CreateToDo
 - C# CreateToDoCommand.cs
 - C# CreateToDoHandler.cs
 - C# CreateToDoRequest.cs
 - C# CreateToDoResponse.cs
 - ▼ DeleteToDo
 - C# DeleteToDoCommand.cs
 - C# DeleteToDoHandler.cs
 - C# DeleteToDoResponse.cs
 - ▼ GetTodos
 - C# GetTodosHandler.cs
 - C# GetTodosQuery.cs
 - C# GetTodosQueryTests.cs
 - C# GetTodosResponse.cs
 - C# AppDbContext.cs
 - C# DateTimeProvider.cs
 - C# IntegrationTestBaseTest...
- ▼ Core
 - ▼ Features/Todos/Entities
 - C# ToDo.cs
 - C# ToDoStatus.cs
 - C# ToDoTests.cs
 - C# IDateTimeProvider.cs
- ▼ e2e
 - ≡ to-dos-happy-path.feature

```
public class CreateToDoRequest
{
    public required string Name { get; set; }
}
```

- ▼ Api
 - ▼ Features/Todos
 - C# TodosController.cs
 - C# TodosControllerValida...
 - C# ControllerValidationTes...
- ▼ Application
 - ▼ EntityTypeConfigurations
 - C# ToDoEntityTypeConfigu...
 - ▼ Features/Todos/Handlers
 - ▼ CreateToDo
 - C# CreateToDoCommand.cs
 - C# CreateToDoHandler.cs
 - C# CreateToDoRequest.cs M
 - C# CreateToDoResponse.cs
 - ▼ DeleteToDo
 - C# DeleteToDoCommand.cs
 - C# DeleteToDoHandler.cs
 - C# DeleteToDoResponse.cs
 - ▼ GetTodos
 - C# GetTodosHandler.cs
 - C# GetTodosQuery.cs
 - C# GetTodosQueryTests.cs
 - C# GetTodosResponse.cs
 - C# AppDbContext.cs
 - C# DateTimeProvider.cs
 - C# IntegrationTestBaseTest...
- ▼ Core
 - ▼ Features/Todos/Entities
 - C# ToDo.cs
 - C# ToDoStatus.cs
 - C# ToDoTests.cs
 - C# IDateTimeProvider.cs
- ▼ e2e
 - ≡ to-dos-happy-path.feature

```
public class CreateToDoRequest
{
    [Required]
    [MaxLength(256)]
    public required string Name { get; set; }
}
```

● Тесты проходят

// </> () <>

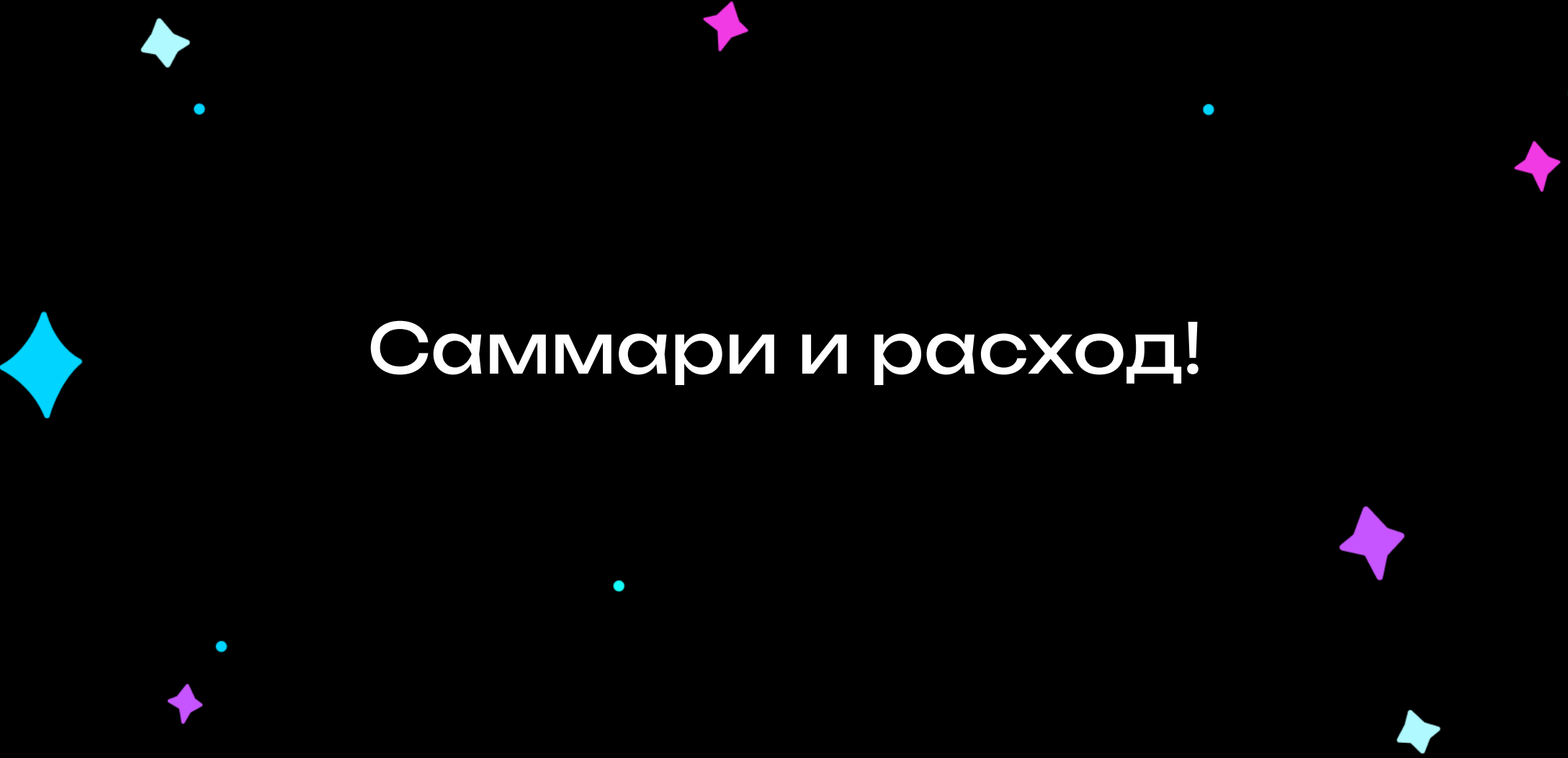
== <> {} ==

// </>

// == </>

<> {} == /

// </> () <> {}

The background is black and decorated with several stylized stars and dots. There are four small cyan stars, four small magenta stars, and five small cyan dots scattered across the slide. A large cyan four-pointed star is positioned to the left of the title.

Саммари и расход!

Тесты кто, где, с кем, зачем?

E2E

Happy Path тесты.
Минимум краевых случаев.
Реальная база.

Integrational

Сортировка,
фильтрация, краевые случаи, валидации на уровне базы.
Реальная база.

Modular

Бизнес-правила, расчёты, не зависящие от ORM.

Controller

Валидация входа.
In-memory база.

Handler

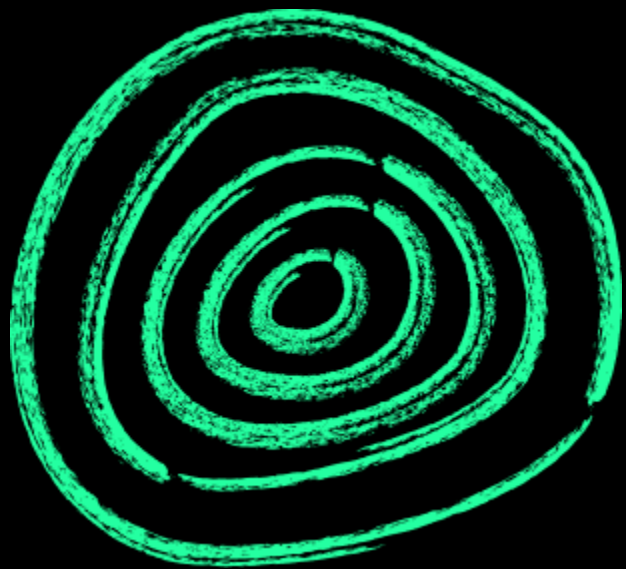
Тестирование флоу вызовов команд, квери, мапперов, калькуляторов и т.д.

Новая папка

Здесь будет ваша придумка.

Выводы / Тезисы / Мысли

- ✓ Можно придумать как протестировать
- ✓ Код тестов как код сервиса
- ✓ TDD не сложно, если архитектура и инфраструктура его поддерживают
- ✓ TDD влияет на тактический дизайн кода
- ✓ TDD не влияет на дизайн кода, если есть архитектура и опыт в ней для типовых задач





.NET бэкенд по TDD

**С НУЛЯ ДО
PRODUCTION-READY** ♦

Анонс серии видео



Здесь будут видео,
как сделать такое
с нуля



[Ссылка на плейлист](#)

Основной репозиторий
на .NET 10



[to-dos-api-dotnet](#)

Production-ready пример
на .NET 9



[inner-circle-time-api](#)

Нюансы DexExpr



DevExpr .NET-бекенда:
Dev Container и пайплайны
на все случаи жизни

Александр Шинкарев

Сетап проекта,
пайплайнов и всего
DevExpr



[Ссылка на видео](#)

Журнал «Турмалин Код»

Почитать
электронную версию
и следить за следующими
выпусками:



Оставить фидбек

