ИНТЕНСИВ

СЛЁРМ

День 2:
Структура проекта и «Чистая архитектура»

Спикер:
Николай Колядко

Вопросы

Таков путь...

День 1:

1. Создание структуры проекта

День 2:

2. Подключение к БД
3. Чистая архитектура:
   - 3.1. Структура папок по чистой архитектуре
   - 3.2. Наполнение – Domain
   - 3.3. Интерфейс – Use Case
   - 3.4. Интерфейс – Repository
   - 3.5. Конструкторы слоёв
   - 3.6. Инициализация слоёв на main
   - 3.7. Реализация интерфейса – Use Case

День 3:

   - 3.8. Реализация интерфейса – Repository
   - 3.9. Реализация – Delivery
4. Использование – Context
5. Добавить логи
6. Добавить трассировку
7. Тестирование**

Подключение к БД (30 мин)

Ветка с примером прошлого задания: 001-day-1-project-layout

1. Создайте свою ветку в git из ветки kolyadkons-01. Правила наименования ФамилияИнициалы-02 Например, kolyadkons-02

2. В папке pkg создать подпапку store/postgres

3. Подключится к БД PostgreSQL
    a. Метод поподключения должен принимать параметры: host, port, user, password, dbname. Также иметь возможность легко их расширять
    b. При успешном подключении должен вернуть работающий коннект к БД иначе вернуть ошибку

4. Вызвать подключение в БД из функции main, расположенную по пути services/contact/cmd

5. Запустить проект

6. Залейте свою ветку в git

Результаты

Успели ли выполнить задание? (опрос)

- ❑ Да
- ❑ Успел на 75%
- ❑ Успел на 50%
- ❑ Успел на 25%
- ❑ Нет

Демонстрация

Структура папок по чистой архитектур (40 мин)

Ветка с примером прошлого задания: 002-day-2-conn-pg

1. Создайте свою ветку в git из ветки kolyadkons-02. Правила наименования ФамилияИнициалы-03 Например, kolyadkons-03

2. В папке services создать подпапку services/contact/internal

3. В internal реализовать структуру проекта по чистой архитектуре
   a. Domain
   b. Repository
   c. Delivery
   d. Use Case

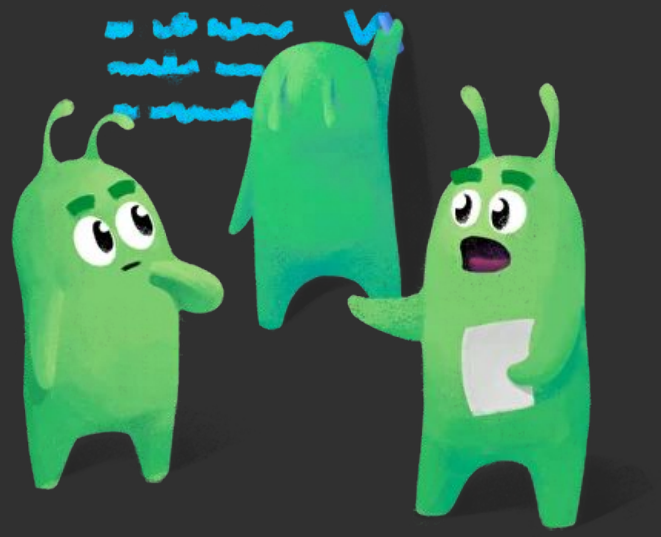
4. Подготовить файлы для интерфейсов
   a. Use Case
   b. Adapters

5. Запустить проект
6. Залейте свою ветку в git

Результаты

Успели ли выполнить задание? (опрос)

- Да
- Успел на 75%
- Успел на 50%
- Успел на 25%
- Нет



Демонстрация

Перерыв 10 мин

Наполнение – Domain (30 мин)

Ветка с примером прошлого задания: 003-day-2-folders-clean-architecture

1. Создайте свою ветку в git из ветки kolyadkons-03. Правила наименования ФамилияИнициалы-04 Например, kolyadkons-04

2. В папке services/contact/internal/domain:
   a. Создать структуру Контакта
      i. Поля: идентификатор, ФИО, номер телефона
      ii. ФИО – заполняется из составных частей (фамилии имени и отчества)
      iii. ФИО – доступно только для чтения
      iv. Номер телефона – должен иметь только числа

   b. Создать структуру Группы
      i. Поля: идентификатор, название
      ii. Название – максимальная длина строки 250 символов

3. Запустить проект

4. Залейте свою ветку в git

Результаты

Успели ли выполнить задание? (опрос)

- ❑ Да
- ❑ Успел на 75%
- ❑ Успел на 50%
- ❑ Успел на 25%
- ❑ Нет

Демонстрация

Интерфейс – Use Case (40 мин)

Ветка с примером прошлого задания: 004-day-2-domain

1. Создайте свою ветку в git из ветки kolyadkons-04. Правила наименования ФамилияИнициалы-05
   Например, kolyadkons-05

2. В папке services/contact/internal/useCase:
   a. Создание интерфейса для Use Case
      i. CRUD контакта
      ii. Создать группу
      iii. Чтение группы
      iv. Создание и добавление контакта в группу

3. Запустить проект

4. Залейте свою ветку в git

Результаты

Успели ли выполнить задание? (опрос)

- ❑ Да
- ❑ Успел на 75%
- ❑ Успел на 50%
- ❑ Успел на 25%
- ❑ Нет

Демонстрация

Обед 60 мин

Интерфейс – Repository (40 мин)

Ветка с примером прошлого задания: 005-day-2-interface-use-case

1. Создайте свою ветку в git из ветки kolyadkons-05. Правила наименования ФамилияИнициалы-06
   Например, kolyadkons-06

2. В папке services/contact/internal/repository:
   a. Создание интерфейса для repository. Интерфейс репозитория должен быть продиктован требованиями от Use Case

3. Запустить проект

4. Залейте свою ветку в git

Результаты

Успели ли выполнить задание? (опрос)

- ❑ Да
- ❑ Успел на 75%
- ❑ Успел на 50%
- ❑ Успел на 25%
- ❑ Нет

Демонстрация

Наполнение слоёв – Конструкторы (30 мин)

Ветка с примером прошлого задания: 006-day-2-interface-repository

1. Создайте свою ветку в git из ветки kolyadkons-06. Правила наименования ФамилияИнициалы-07
   Например, kolyadkons-07

2. В папке services/contact/internal:
   a. Создать конструкторы слоёв. Repository/UseCase/Delivery. Каждый слой должен иметь конструктор

3. Запустить проект

4. Залейте свою ветку в git

Результаты

Успели ли выполнить задание? (опрос)

- ❑ Да
- ❑ Успел на 75%
- ❑ Успел на 50%
- ❑ Успел на 25%
- ❑ Нет

Демонстрация

Перерыв 10 мин

Инициализация слоёв на main (20 мин)

Ветка с примером прошлого задания: 007-day-2-layers-constructor

1. Создайте свою ветку в git из ветки kolyadkons-07. Правила наименования ФамилияИнициалы-08
   Например, kolyadkons-08

2. В папке services/contact/:
   a. Добавить вызовы созданных конструкторов и запустить вечный цикл для прослушивания REST запросов

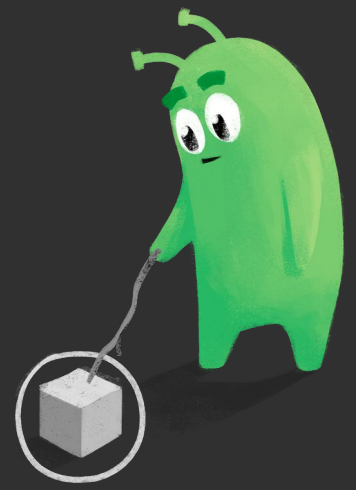
3. Запустить проект

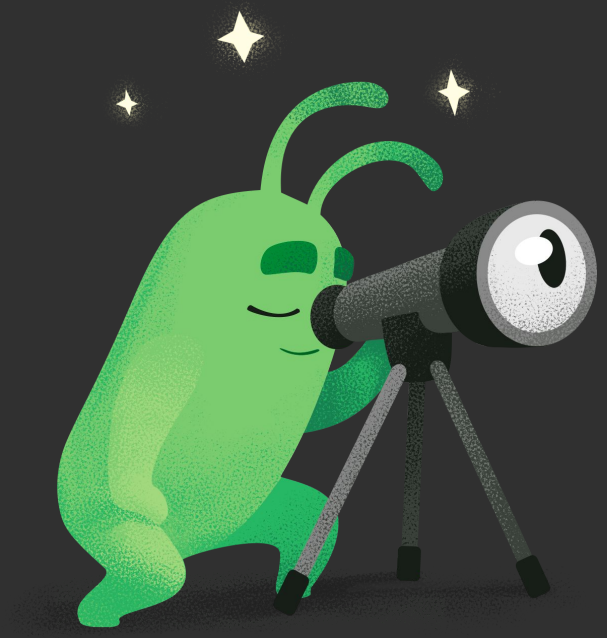
4. Залейте свою ветку в git

Результаты

Успели ли выполнить задание? (опрос)

- ❑ Да
- ❑ Успел на 75%
- ❑ Успел на 50%
- ❑ Успел на 25%
- ❑ Нет





Демонстрация

Реализация интерфейса Use Case (40 мин)

Ветка с примером прошлого задания: 008-day-2-init-main

1. Создайте свою ветку в git из ветки kolyadkons-08. Правила наименования ФамилияИнициалы-09
   Например, kolyadkons-09

2. В папке services/contact/useCase:
   a. CRUD контакта
   b. Создание группы
   c. Чтение группы
   d. Создание и добавление контакта в группу

3. Запустить проект

4. Залейте свою ветку в git

Результаты

Успели ли выполнить задание? (опрос)

- Да
- Успел на 75%
- Успел на 50%
- Успел на 25%
- Нет

Демонстрация

Таков путь...

День 1:

1. Создание структуры проекта

День 2:

2. Подключение к БД
3. Чистая архитектура:
   - 3.1. Структура папок по чистой архитектуре
   - 3.2. Наполнение – Domain
   - 3.3. Интерфейс – Use Case
   - 3.4. Интерфейс – Repository
   - 3.5. Конструкторы слоёв
   - 3.6. Инициализация слоёв на main
   - 3.7. Реализация интерфейса – Use Case

День 3:

   - 3.8. Реализация интерфейса – Repository
   - 3.9. Реализация – Delivery
4. Использование – Context
5. Добавить логи
6. Добавить трассировку
7. Тестирование**

Вопросы