

Паттерны проектирования и структура приложений с пользовательским интерфейсом, сервисами обработки и базой данных

Семинар 10





Что будет на уроке сегодня



Наши цели:

- вспомнить подход Domain Driven Design (DDD);
- вспомнить подход Feature-driven development (FDD);
- вспомнить подход Test Driven Development (TDD);
- вспомнить Cloud-native архитектуру;
- вспомнить Паттерны связывания компонент и уровней приложения:
API Gateway, Backends for Frontends, Repository, External Configuration, Self-checking.
- спроектировать облачное приложение (Блок 1).



Вопросы?

Вопросы
по лекции?

Вопросы?





Проговорим важные моменты



Вспомним

подход Domain Driven Design (DDD)

подход Feature-driven development (FDD)

подход Test Driven Development (TDD)





Вспомним

Cloud-native архитектуру





Вспомним

Паттерны связывания компонент и уровней приложения:

API Gateway, Backends for Frontends, Repository, External Configuration, Self-checking.





Практика



Цель задания: научиться проектировать облачное приложение.

Задание:

Спроектировать облачное приложение с интерфейсами в браузере и нативными интерфейсами в мобильных устройствах.



- Необходимо спроектировать облачный сервис домашнего робота пылесоса для уборки помещений.
- Результатом должны быть: компоненты интерфейсов, доменная модель, Use case, компонентные диаграммы, EDR, API контракты, тестовые сценарии.
- **Работа разделена на 3 семинара.**
- **Сегодня мы возьмем блок 1.**



- a. Спроектировать пользовательский интерфейс (web-SPA, native mobile), основные компоненты (подключение робота, управление помещениями, расписание работы, сервисное обслуживание робота, история уборок), <https://www.figma.com/> или <https://app.diagrams.net/>.
 - b. Спроектировать доменную модель, в виде текста Домен – атрибуты.
 - c. Спроектировать сценарии (Use case)(подключение, выбор помещения, программы уборки, настройка расписания, просмотр статистики..), в виде Актор – Прецедент (из первой лекции).
 - d. Спроектировать слой API Gateway (mobile, web), сформировать REST запросы: GET, POST, PUT, DELETE (<https://swagger.io>).
- ** (необязательно) Разработать REST контракты API между компонентами и сгенерировать (автоматически на ресурсе <https://swagger.io>) код на разных языках программирования.
- e. Спроектировать компоненты бизнес-логики и связать их API Gateway с применением паттерна BFF <https://app.diagrams.net/>.
 - f. Определить состав информации для кеширования на уровне приложения пользователя, API Gateway, уровня бизнес-логики и уровня репозитория. Список.
 - g. Спроектировать ER модель (<https://www.dbdesigner.net/>), запросы в БД и уровень хранения данных (СУБД).



Перерыв?

Голосуйте в чате



Домашнее задание



ДЗ

Доработать пункты, которые остались не сделанными на семинаре

Инструменты:

- <https://www.figma.com/>
- <https://app.diagrams.net/>
- <https://www.dbdesigner.net/>
- <https://swagger.io>



Вопросы?

Вопросы?



Вопросы?





Подведем итоги



Напишите 3 вещи в
комментариях, которым
вы научились сегодня.





Спасибо за работу!