# План создания курсов по React + ASP.NET Core

## Что нужно сделать в целом

1. Презентация для сопровождения видосов
2. Создать новое мини-приложение с тематикой «сайта недвижимости» - ReactRealtyCourse.
3. Спланировать видеоуроки – что записываю, что буду говорить
4. Записать видео, как создаю приложение. Видимо, придется делать еще один проект «рядом» с ReactRealtyCourse и воссоздавать его постепенно (чтобы в итоге получился тот же ReactRealtyCourse).

## ГЛАВА 1 – Подготовка проекта

### Создание проекта

Что должно быть установлено на ПК: VS2019, Node.js, Net Core framework.

Разделяем на сборки Business, Frontend, DAL, Tests

### Интеграция React в проект

Создаем папку App в Frontend, смотрим package.json и webpack.config.json.

Настраиваем Middleware, чтобы работал запуск проекта с одновременным созданием bundle.js.

Выводим helloworld в самом начальном компоненте App.

### Добавляем Роутинг и компоненты

Нужны компоненты apartmentIndex, houseIndex, apartmentRead и houseRead с папками для каждого из них.

## ГЛАВА 2 – работа с бэкендом

### Моделируем предметную область

Добавляем базовый класс IdInfo.

Добавляем в Business модельки House и Apartment.

### Добавляем контроллеры с API и роутингом.

Нужны методы взятия 1 квартиры, 1 дома, всех квартир и всех домов.

Пока что пусть возвращают данные не из БД (а модельки уже готовы!)

Тестируем – просто вызываем GET-методы в браузере

## ГЛАВА 3 – подключение Redux

### Немного лекций по Redux

Говорим, что у каждого компонента будет состояние, редьюсер для перехода между состояниями и экшены, реализующие переход между состояниями.

### Проектируем состояние для компонентов

Начинаем на примере компонента houseRead. Добавляем houseReadCoinstants, houseReadReducer, houseReadActions.

Показываем готовые состояния для остальных компонентов.

### Объединение редьюсеров в один

Импортируем функцию. Импортируем 4 редьюсера, делаем 1 общий.

### Используем thunk для загрузки данных с сервера

Добавляем в состояния поле isLoading.

Добавляем экшен с загрузкой данных

Настраиваем условное отображение данных на примере houseRead.

Показываем отображение для остальных компонентов.

## ГЛАВА 4 – База данных и связь с ней

### Создаем БД по предметной области в MS SQL

Добавляем папку SQL и скрипт All\_tables\_creation в проект DAL. Создаем таблицы House и Apartment.

Задаем FK и Constraint между ними.

Заполняем тестовыми данными.

### Подключение Entity Framework

Добавляем EF Core

Создаем контекст с DbSet’ами для 2 моделек.

Добавяем юнит-тест коннекшена к БД, заигнориваем его.

### Создание репозиториев

Создаем 2 одинаковых репозитория для 2 моделек.

Рефакторим в один дженерик-репозиторий.

Добавляем юнит-тест выборки из БД 2 сущностей, заигнориваем.

### Добавляем выборку реальных данных в контроллеры

Внедрение зависимостей, чтобы были доступны репозитории

## ГЛАВА 5 - Доработка фронтенда

### Подключение AntDesign

Устанавливаем AntD. Рендерим инпут для теста.

Рендерим подложку для сайта

### Работа с Description’ами

Украшаем компоненты houseRead и apartmentRead

### Работа с гридом

Подключаем грид к компонентам apartmentIndex и houseIndex

### Добавляем доп. стили в Site.css

Делаем выравнивание в таблицах.

Уменьшаем отступы в Description’ах.

### Добавляем кнопки «Вернуться к списку домов/квартир»

Говорим про хук с навигацией

### Разные названия вкладок браузера на разных страницах

Добавляем названия «RealtyCourseApp - Apartments» и т.п.

## ГЛАВА 6 – доработка бэкенда

### Добавляем пейджинг в гриды

Заполняем БД, чтобы было больше тестовых данных

Пробрасываем номер страницы в экшен и в колнтроллер. Делаем на примере компонента houseIndex, показываем, что в компоненте apartmentIndex.

### Запоминаем последний номер страницы в состоянии

Чтобы при нажатии на кнопку «Вернуться» в гриде была выбрана та же страница, что и раньше.

### Добавляем фильтрацию в грид

Для примера – по году постройки и кол-ву этажей у домов.

### Добавляем сортировку в грид

По дате создания и по цене для квартир.