Одностраничное приложение (SPA). По запросу http://localhost:8080/ сервер отдает index.htlm файл (лежит в папке build). Браузер парсит этот файл и подгружает подключенные javascript файлы и css стили.

Ключевые технологии

- Typescript - типизация для javascript
- React - библиотека для разработки пользовательских интерфейсов
- Redux - управление состоянием приложения
- Axios - библиотека, упрощающая получение данных с сервера
- Webpack - сборщик

Файловая структура проекта

package.json - основной файл проекта, в котором указаны все зависимости
build - файлы после сборки (для размещения на сервере)
public - для хранения статических файлов
ui-kit - библиотека ui компонентов
webpack - конфигурационные файлы сборщика
src - исходный код

- **index.tsx** - файл, который является точкой входа для сборщика

- **app.tsx** - файл с маршрутизацией (указано какой компонент использовать в зависимости от url)
- **pages** - компоненты страниц
- **store** - хранилище, состояние приложения.
- **server_api** - запросы на сервер
- **types** - typescript типы
- **helpers** - файлы с какими-то вспомогательными функциями
- **css** - файлы с общими css стилями
- **transform** - здесь хранится логика преобразования объектов (из того что приходит в то что нужно)
- **hooks** - переиспользуемые хуки

Архитектурная схема с указанием информационных потоков (на листочке от руки, без сильной детализации)

Стремлюсь пока придерживаться 3-х слойной архитектуры

1. DAL - слой доступа к данным (Data Access Layer)
2. BLL - слой предметной области (Business Logic Layer)
3. PI или UI - пользовательский интерфейс (Presentation Layer)

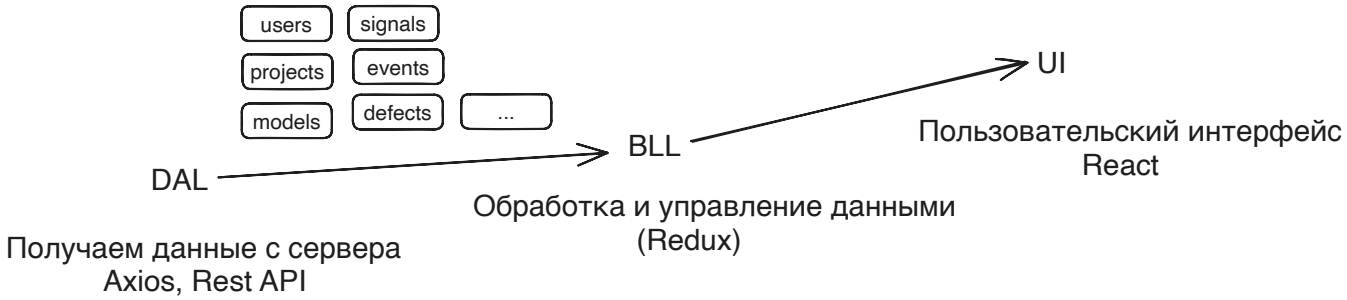

Схема хранения данных на стороне клиента, тоже очень общо, от руки

Данные хранятся на стороне клиента в нормализованном виде (подобно реляционной базе данных), то есть то, что приходит с сервера, хранится практически в неизменном виде с использованием идентификаторов для связи. Структуры данных, которые нужны только на клиенте хранятся отдельно (размеры окон, состояние открытости/закрытости окон и тд)

```
▶ auth (pin): { isLogin: true, isAuthError: false }
▶ profile (pin): { id: 4, isLogin: true, name: "orlov", … }
▶ users (pin): { ids: […], entities: {…} }
▶ projects (pin): { ids: […], entities: {…}, path: "", … }
▶ models (pin): { ids: […], entities: {…}, activeModelId: 0, … }
```