

# Персональный трекер освоения технологий

---

**Цель проекта:** - разработать одностраничное приложение (SPA) на React, которое поможет пользователю систематизировать и отслеживать свой прогресс в изучении новой технологии или направления по готовой дорожной карте (roadmap).

Представьте, что вы решили изучить, например, фреймворк React. В интернете вы находите структурированную дорожную карту в формате JSON, которая содержит все ключевые темы для освоения: от основ JSX до работы с состоянием и маршрутизацией. Ваше приложение позволит не просто просматривать эту карту, а активно с ней работать: отмечать пройденные темы, записывать свои заметки, планировать сроки и визуально видеть свой прогресс.

Вдохновлено сервисом: <https://roadmap.sh/>

---

## Функциональные требования

### 1. Работа с дорожной картой

Приложение должно уметь загружать дорожную карту из стандартизированного JSON-файла. Это означает, что пользователь может найти карту по интересующему его направлению и импортировать её в вашу программу. Структура файла должна быть простой и включать в себя название карты, её описание и массив пунктов для изучения. Каждый пункт, в свою очередь, содержит уникальный идентификатор, название, описание и может иметь ссылку на внешние ресурсы.

Не менее важной является возможность экспорта. После того как пользователь провел работу с картой - поставил статусы, добавил заметки - он должен иметь возможность сохранить результат своего труда. Экспортированный файл должен содержать как исходную структуру карты, так и все данные, введенные пользователем.

### 2. Визуальное представление карты

Главный экран приложения должен отображать дорожную карту в виде интерактивной сетки или списка карточек. Визуальный дизайн должен быть чистым и понятным. Каждая карточка пункта должна нести базовую информацию, такую как название, но самое главное - она должна визуальным образом отображать свой текущий статус. Например, цвет карточки или специальная иконка будут сразу показывать, выполнена ли тема, находится ли она в работе или еще не начата. Это дает пользователю возможность с первого взгляда оценить общую картину своего прогресса.

### 3. Детальный просмотр и редактирование пунктов

Когда пользователь кликает по карточке, он переходит на страницу с детальной информацией по этому пункту. На этой странице отображается всё, что известно о пункте: его название, развернутое описание и полезные ссылки.

Ключевой элемент здесь - возможность персонализации. Пользователь должен иметь возможность добавить к пункту свою текстовую заметку. Это может быть конспект, полезная команда, ссылка на решенную задачу или любые другие мысли, связанные с изучением. Эту заметку можно должно быть не только создать, но и в дальнейшем отредактировать или удалить.

Кроме того, здесь же пользователь управляет статусом пункта и сроками. Он может переключать статус между вариантами «Не начат», «В работе» и «Выполнено». Для более точного планирования следует реализовать возможность установки желаемой даты завершения работы над темой.

#### 4. Система статусов и планирования

Статусы являются ядром трекера, превращая статичную карту в динамичный инструмент. Они позволяют контролировать прогресс. На основе этих статусов можно рассчитывать и отображать общий прогресс в процентах, например, в виде прогресс-бара в шапке приложения.

Функция установки сроков добавляет цели в процесс обучения. Она помогает пользователю оставаться сфокусированным и дисциплинированным, превращая абстрактное желание «изучить React» в конкретный план действий с дедлайнами.

---

### Критерии оценивания

Оценка выставляется по 100-процентной шкале, где каждая функциональность имеет свой вес. Максимальный балл в БРС за эту работу - **7 баллов**.

#### Работа с дорожной картой (25%)

**10% - корректная загрузка и парсинг JSON-файла с картой**, отображение его структуры в приложении.

**10% - функция экспорта текущего состояния карты** (всех пользовательских данных) в новый JSON-файл.

**5% - обработка ошибок:** приложение адекватно реагирует на попытку загрузить файл неверного формата (выводит сообщение для пользователя, не ломается).

#### Интерфейс и маршрутизация (30%)

**10% - главная страница:** карточки пунктов отображаются в виде аккуратной сетки, каждая карточка визуально отражает свой статус (разным цветом или иконкой).

**10% - детальная страница:** реализована с помощью React Router, отображает полную информацию о выбранном пункте. Наличие кнопки или ссылки для возврата на главную страницу.

**10% - удобство использования:** интерфейс интуитивно понятен, нет критических ошибок в логике работы (например, невозможность закрыть модальное окно), данные на детальной странице соответствуют выбранной карточке.

#### Функциональность заметок и статусов (35%)

**15% - работа с заметками:** на детальной странице есть текстовое поле (или форма) для добавления и редактирования пользовательской заметки. Заметка сохраняется и отображается при повторном открытии пункта.

**10% - управление статусами:** реализован механизм переключения статуса пункта (минимум 3 состояния). Текущий статус отображается как на карточке на главной странице, так и на детальной.

**10% - расчет и отображение прогресса:** в шапке приложения или на главной странице отображается общий прогресс пользователя по карте в виде процентов или прогресс-бара, который пересчитывается при изменении любого статуса.

#### **Качество кода (10%)**

**5% - код читаем**, логично структурирован на компоненты, отсутствуют грубые нарушения базовых принципов React (например, мутация состояния).

**5% - проект запускается без ошибок** стандартными командами `npm install` и `npm start`.

Работа сдаётся в виде ссылки на git-репозиторий, клонирование которого в локальную среду и запуск команд `npm install` и `npm start` приведут к запуску проекта. Демонстрация работоспособности проекта производится непосредственно на практическом занятии.