Архитектура программного средства «TaskMaster v1.0»

1. Выбранный тип архитектуры

Для проекта «TaskMaster» выбрана клиент-серверная архитектура с использованием REST API.

Это оптимальный выбор, так как:

1) позволяет разделить пользовательский интерфейс (Frontend) и бизнес-логику (Backend),
2) обеспечивает независимую разработку и масштабирование фронтенда и бэкенда,
3) упрощает интеграцию с мобильными приложениями в будущем,
4) соответствует современным стандартам веб-разработки.

2. Основные компоненты системы

Клиент (Frontend)

Одностраничное приложение (SPA).

Технологии: React.js (или Vue.js) + HTML/CSS/JS + библиотека для работы с HTTP (Axios/Fetch).

Отвечает за:

1) отображение интерфейса (списки проектов, задач, формы создания/редактирования),
2) взаимодействие с пользователем,
3) отправку запросов к серверу и отображение полученных данных.

Сервер (Backend)

Серверное приложение, реализующее REST API.

Технологии: Node.js + Express (или Python + FastAPI/Flask).

Отвечает за:

1) обработку запросов от клиента,
2) бизнес-логику (валидация, расчёты, права доступа),

3) взаимодействие с базой данных,
4) формирование ответов (JSON).

База данных (Database)

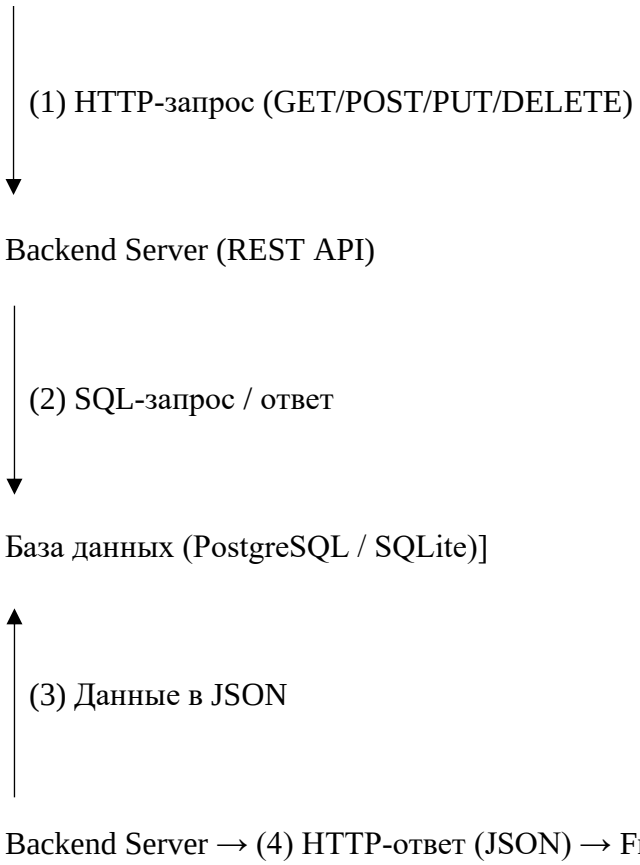
Реляционная СУБД.

Выбор: PostgreSQL (основной вариант) или SQLite (для прототипа и локальной разработки).

Хранит:

1) пользователей (id, email, хэш пароля, роль),
2) проекты (id, название, описание, владелец),
3) задачи (id, название, описание, статус, исполнитель, дедлайн, проект).

3. Схема взаимодействия компонентов



Описание потока:

1. Пользователь взаимодействует с интерфейсом (создаёт задачу).

2. Frontend отправляет HTTP-запрос на Backend (например POST /api/tasks).

3. Backend проверяет права, валидирует данные, обращается к БД.

4. БД возвращает результат → Backend формирует JSON-ответ.

5. Frontend получает ответ и обновляет интерфейс.

4. Примеры ключевых HTTP-endpoint'ов (REST API)

GET /api/projects

Описание: Получить список всех проектов текущего пользователя.

Параметры: — (аутентификация через токен в заголовке Authorization)

Ответ (200 OK):

```json
[
  { "id": 1, "title": "Сайт компании", "description": "Разработка лендинга", "ownerId": 5 },
  { "id": 2, "title": "Мобильное приложение", "description": "...", "ownerId": 5 }
]
```

POST /api/tasks

Описание: Создать новую задачу в указанном проекте.
Тело запроса (JSON):

```
{
  "projectId": 1,
  "title": "Сделать дизайн главной страницы",
  "description": "Использовать Figma, цвета бренда",
  "status": "Новая",
  "assigneeId": 7,
```

"deadline": "2026-02-20"

}

Ответ (201 Created):

{ "id": 42, "title": "Сделать дизайн...", ... }

PUT /api/tasks/{id}
Описание: Обновить задачу (например, изменить статус или исполнителя).
Путь: /api/tasks/42
Тело запроса (JSON, частичное обновление):

{

"status": "В работе",

"assigneeId": 8

}

Ответ (200 OK):

{ "id": 42, "title": "...", "status": "В работе", ... }

Дата составления архитектуры: 11 февраля 2026 г.

Ответственный: Залялетдинов Марат Артурович