

# Техническое задание

Проект: Онлайн-дашборд управления процессами предприятия

## 1. Цель работы

Разработать веб-приложение (дашборд) для контроля процессов предприятия в режиме близком к реальному времени: заявки, задачи, склад/выдача, статусы работ и ключевые показатели (KPI).

## 2. Модель предприятия (упрощённая)

В рамках учебного проекта предприятие моделируется упрощённо: система оперирует заявками и задачами, а также (опционально) складскими операциями. Данные могут генерироваться, вводиться вручную или поступать через API.

## 3. Роли пользователей

- Администратор: управление пользователями, справочниками, настройками дашборда.
- Менеджер: просмотр показателей, контроль заявок/задач, изменение статусов.
- Исполнитель: просмотр назначенных задач, обновление статусов и комментариев.

## 4. Основные сущности (данные)

- Пользователи: ФИО, роль, логин/пароль.
- Заявки: номер, инициатор, тип, приоритет, статус, дата, описание.
- Задачи: заявка\_id, исполнитель, срок, статус, комментарии.
- События/лог: кто и когда менял статусы/поля.
- Справочники (минимум): типы заявок, статусы, приоритеты.

## 5. Функциональные требования (дашборд)

- Главная страница дашборда с виджетами KPI: количество новых заявок, в работе, просроченных, закрытых.
- Графики/диаграммы (минимум 2): динамика заявок по дням, распределение по статусам/типам.
- Таблица «Последние заявки» с фильтрами (статус, приоритет, период) и поиском.
- Страница заявки: детали, история изменений, связанные задачи, комментарии.

## 6. Обновление данных в реальном времени

- Дашборд должен обновляться без перезагрузки страницы.

- Способ обновления: WebSocket (предпочтительно) или периодический опрос API (polling) каждые N секунд.
- При изменении заявки/задачи пользователями — обновления должны отображаться у других пользователей.

## **7. Функциональные требования (управление процессами)**

- Создание заявки (форма).
- Изменение статуса заявки и приоритета.
- Назначение исполнителя и создание задач по заявке.
- Добавление комментариев.
- Фиксация всех изменений в журнале событий (audit log).

## **8. Административная панель (минимум)**

- Авторизация.
- CRUD пользователей и ролей.
- CRUD справочников (статусы, типы заявок, приоритеты).
- Просмотр журнала событий и ошибок.

## **9. Требования к базе данных (MySQL, минимум)**

- users (id, name, role, login, password\_hash, created\_at).
- tickets (id, number, creator\_id, type\_id, priority\_id, status\_id, title, description, created\_at, updated\_at, due\_at).
- tasks (id, ticket\_id, assignee\_id, status\_id, title, comment, due\_at, created\_at, updated\_at).
- comments (id, ticket\_id, author\_id, text, created\_at).
- audit\_log (id, entity, entity\_id, action, before\_json, after\_json, author\_id, created\_at).
- dictionaries (types/statuses/priorities).

## **10. Технические требования**

- Frontend: React или HTML/CSS/JS.
- Backend: Node.js (Express/Fastify) или PHP.
- Обмен данными: REST API + WebSocket.
- Обработка ошибок и валидация входных данных.
- Адаптивный интерфейс (Desktop / Tablet / Mobile).
- README с инструкцией запуска (backend, frontend, миграции БД).

## **11. Ограничения**

- Можно использовать тестовые данные или генератор данных.

- Интеграция с реальными системами предприятия (1С и т.п.) не требуется.
- Достаточно одного «модуля процессов» (заявки/задачи). Склад — опционально.

## **12. Результат работы**

- Рабочее веб-приложение с ролями и авторизацией.
- Дашборд с KPI и графиками, обновляющийся без перезагрузки.
- Демонстрация изменения статуса заявки с отражением изменений у другого пользователя (через WebSocket/polling).
- Исходный код и база данных.

## **13. Критерии оценки**

- Наличие ключевых сущностей (заявки/задачи) и корректная бизнес-логика статусов.
- Наглядные KPI и графики, корректные фильтры/поиск.
- Реализация обновлений без перезагрузки (WebSocket или polling).
- Логирование изменений (audit log).
- Качество UI и адаптивность.