**Техническое задание (структура с чек-листами) – ООО «СистемаКонтроля» по работе с строительными объектами**

Цель разработки — создать монолитное веб-приложение для централизованного управления дефектами на строительных объектах. Система должна обеспечить полный цикл работы: от регистрации дефекта и назначения исполнителя до контроля статусов и формирования отчётности для руководства.

Система предназначена для:

* инженеров (регистрация дефектов, обновление информации);
* менеджеров (назначение задач, контроль сроков, формирование отчётов);
* руководителей и заказчиков (просмотр прогресса и отчётности).

**1. Анализ**

На первом этапе требуется провести полный анализ предметной области. Важна фиксация бизнес-целей, определение ролей пользователей и описание типовых сценариев. Система должна учитывать потребности инженеров, менеджеров и руководителей, поэтому необходимо собрать как функциональные, так и нефункциональные требования. Итогом этапа станет формализованное техническое задание (SRS), а также диаграммы вариантов использования и набор пользовательских историй. Это обеспечит прозрачность и исключит спорные моменты на следующих стадиях.

|  |  |
| --- | --- |
| **Пункт** | **Почему важно** |
| Определение функциональных требований | Без этого разработка может пойти в неверном направлении и система не решит бизнес-задачи. |
| Определение нефункциональных требований | Позволяет избежать проблем с производительностью и совместимостью. |
| Согласование ролей пользователей | Исключает конфликт интересов и обеспечивает правильное разграничение доступа. |
| Разработка Use Case | Дает возможность проверить, что сценарии работы понятны и полны. |
| Подготовка User Stories | Фокусирует разработку на реальных потребностях пользователей. |
| Создание SRS | Обеспечивает документированную базу для дальнейших этапов. |
| Утверждение результатов анализа | Защищает от изменений требований в последний момент. |

### Функциональные требования

1. Регистрация пользователей и аутентификация.
2. Разграничение прав доступа (менеджер, инженер, наблюдатель).
3. Управление проектами/объектами и их этапами.
4. Создание и редактирование дефектов (заголовок, описание, приоритет, исполнитель, сроки, вложения).
5. Управление статусами дефектов: Новая → В работе → На проверке → Закрыта/Отменена.
6. Ведение комментариев и истории изменений.
7. Поиск, сортировка и фильтрация дефектов.
8. Экспорт отчётности в CSV/Excel.
9. Просмотр аналитических отчётов (графики, статистика).

### Нефункциональные требования

* Время отклика страницы ≤ 1 секунды (для 50 активных пользователей).
* Обеспечить резервное копирование БД раз в сутки.
* Интерфейс на русском языке, адаптивный под ПК/планшеты.
* Совместимость с Chrome/Firefox/Edge последних версий.
* Пароли хранить с использованием bcrypt или argon2.
* Защита от SQL-инъекций, XSS и CSRF.

**2. Проектирование**

После утверждения требований нужно перейти к проектированию. Здесь формируется архитектура приложения и структура базы данных, разрабатываются прототипы интерфейсов. Также подготавливается декомпозиция работ и сетевой график. Это позволит связать цели заказчика с конкретными техническими решениями и убедиться, что будущая система органично встанет на наши серверы.

Система предназначена для:

* регистрации и классификации дефектов на строительных объектах;
* назначения ответственных лиц и сроков устранения;
* отслеживания статусов исправления дефектов;
* формирования аналитической отчётности для руководства и заказчика.

Система должна сократить потери информации, повысить прозрачность работы, а также обеспечить единый инструмент взаимодействия между инженерами, менеджерами и заказчиками.

|  |  |
| --- | --- |
| **Пункт** | **Почему важно** |
| Разработка архитектурной схемы | Обеспечивает понимание структуры приложения и упрощает его поддержку. |
| Подготовка ER-диаграммы | Позволяет организовать данные и избежать дублирования. |
| Создание прототипов интерфейсов | Снижает риск, что пользователи не смогут работать в системе. |
| Утверждение проектных решений | Исключает переделки на поздних этапах. |
| Согласование с заказчиком | Даёт гарантию, что решение соответствует бизнес-целям. |

**3. Разработка**

Разработка реализует проектные решения и превращает их в работающий код. Необходимо последовательно создать модули пользователей и авторизации, проекты, дефекты и отчёты. Каждый модуль должен быть интегрирован в общую архитектуру и доступен для проверки. Код размещается в репозитории, проходит сборку и готовится к развертыванию на инфраструктуре заказчика.

|  |  |
| --- | --- |
| **Пункт** | **Почему важно** |
| Реализация Users/Auth | Безопасный доступ и разграничение прав. |
| Реализация Projects | Упорядочивает данные по объектам и этапам. |
| Реализация Defects | Решает ключевую бизнес-задачу фиксации и контроля дефектов. |
| Реализация Reports | Даёт руководству инструмент анализа и контроля. |
| Поддержка вложений (фото, документы) | Повышает достоверность информации о дефектах. |
| Выкладка кода в репозиторий | Обеспечивает прозрачность и командную работу. |
| **Гарантия, что система может быть развернута на сервере.** | |

**4. Безопасность**

Особое внимание уделяется безопасности. Система будет хранить данные заказчика и пользователей, поэтому требуется защита на всех уровнях. Пароли должны храниться в зашифрованном виде, права доступа разграничены, уязвимости исключены. Логирование действий и резервное копирование обеспечивают контроль и восстановление в случае сбоев.

|  |  |
| --- | --- |
| **Пункт** | **Почему важно** |
| Хранение паролей в bcrypt/argon2 | Исключает утечки данных пользователей. |
| Проверка защиты от SQL/XSS/CSRF | Минимизирует угрозы взлома системы. |
| Настройка логирования | Позволяет отслеживать действия и расследовать инциденты. |
| Резервное копирование БД | Обеспечивает восстановление при авариях. |
| Контроль доступа к логам | Защищает служебную информацию от злоупотреблений. |

**5. Тестирование**

Перед внедрением необходимо убедиться в стабильности и соответствии системы заявленным требованиям. Для этого готовится план тестирования, выполняются модульные и интеграционные проверки, проводится нагрузочное тестирование. Все пользовательские сценарии должны быть выполнены без ошибок, а производительность должна соответствовать ожиданиям.

|  |  |
| --- | --- |
| **Пункт** | **Почему важно** |
| Подготовка плана тестирования | Системный подход к проверке. |
| Реализация юнит-тестов | Контроль качества отдельных модулей. |
| Реализация интеграционных тестов | Проверка целостности цепочек действий. |
| Проверка User Stories | Подтверждение выполнения требований заказчика. |
| Нагрузочное тестирование | Гарантия работы при реальном числе пользователей. |
| Регрессионное тестирование | Исключение повторного появления ошибок. |
| Контроль покрытия кода тестами | Показатель качества разработки. |

Предоставить:

• Подготовлен план тестирования.

• Реализовано ≥ 5 юнит-тестов.

• Реализовано ≥ 2 интеграционных сценария.

• Пройдены тесты по критериям приёмки User Stories.

• Проведено нагрузочное тестирование (отклик ≤ 1 сек).

**6. Введение в эксплуатацию**

Заключительный этап предполагает установку системы на серверы заказчика, настройку окружения и проверку работы в боевых условиях. Параллельно готовится документация и проводится обучение пользователей. Завершается этап приёмкой системы и запуском её в опытную эксплуатацию.

**Контрольные точки (чек-лист):**

• Подготовлено окружение для деплоя.

• Система развёрнута и протестирована на сервере (виртуальном).

**Выставление оценок**

|  |  |  |  |
| --- | --- | --- | --- |
| **Этап** | **Оценка «3» (минимум)** | **Оценка «4» (хорошо)** | **Оценка «5» (отлично)** |
| **1. Анализ** | Перечислены базовые функциональные требования (FR) и роли пользователей. | Полный перечень FR и NFR, Use Case/диаграммы, User Stories. | Подробное SRS, анализ рисков требований, приоритезация FR/NFR, трассировка требований к задачам и тестам. |
| **2. Проектирование** | Элементарная архитектурная схема и список таблиц БД. | Архитектурная схема с описанием слоёв, ER-диаграмма, WBS, прототип интерфейсов. | Завершённая архитектура с вариантами масштабирования, согласованный прототип UX, диаграмма Ганта, описание выбора фреймворка и библиотек. |
| **3. Разработка** | Реализован 1–2 модуля (например, Users/Auth и Defects), базовый CRUD. | Все ключевые модули реализованы (Users, Projects, Defects, Reports), код в репозитории, система запускается локально. | Полностью рабочий прототип, поддержка вложений и фильтрации, документация по коду и API, CI/CD пайплайн. |
| **4. Безопасность** | Реализована базовая аутентификация, пароли хранятся в хэше. | Разграничение доступа по ролям, базовые проверки от SQL/XSS/CSRF, логирование. | Полный набор мер: резервное копирование, контроль доступа к логам, аудит безопасности по чек-листу (OWASP Top 10). |
| **5. Тестирование** | 2–3 юнит-теста, проверка запуска приложения. | План тестирования, ≥5 юнит-тестов, ≥2 интеграционных, проверка User Stories. | Нагрузочное тестирование, регрессия, покрытие кода ≥50%, результаты тестов документированы. |
| **6. Введение в эксплуатацию** | Система запускается локально, есть короткая инструкция. | Развёртывание на тестовом сервере, пользовательская документация, обучение. | Полноценный деплой на сервер заказчика, мониторинг и план сопровождения, акт приёмки. |