

ЭСКИЗНЫЙ ПРОЕКТ

на создание автоматизированной системы
“Курсы по повышению квалификации”

Оглавление

Ведомость эскизного проекта	3
Пояснительная записка к эскизному проекту	3
Общие положения	3
Основные технические решения	3
Решения по структуре системы	3
Общая структура базы данных	4
Решения по режимам функционирования системы	4
Решения по численности, квалификации и функциям персонала	5
Решения по составу программных средств, языкам деятельности, алгоритмам процедур и операций и методам их реализации	5
Источники разработки	6

Ведомость эскизного проекта

На предыдущих стадиях разработки программного модуля «Курсы по повышению квалификации» были составлены и утверждены следующие документы:

- Техническое задание на создание программного модуля «Курсы по повышению квалификации», разработанное на основании ГОСТ 34.602—2020 на написание ТЗ на автоматизированные системы управления от г.2020

Пояснительная записка к эскизному проекту

Общие положения

Данный документ является эскизным проектом на создание программного модуля «Курсы повышения квалификации» (далее система).

Перечень организаций, участвующих в разработке системы, сроки и стадии разработки, а также ее цели и назначение указаны в техническом задании на создание автоматизированной системы.

Основные технические решения

Решения по структуре системы

Программный модуль «Курсы по повышению квалификации» будет представлять собой подсистему, интегрированную в существующую автоматизированную систему учета обучения Заказчика. Система будет построена на основе трехуровневой архитектуры “клиент-сервер”, что обеспечивает масштабируемость, надежность и безопасность данных:

- **Уровень данных (СУБД):** В качестве системы управления базами данных используется реляционная СУБД PostgreSQL. На этом уровне хранятся все данные системы: информация об обучающихся, курсах, группах, процессе обучения, а также сгенерированные свидетельства в формате PDF (в виде ссылок на файловое хранилище или BLOB-объектов). Доступ к данным осуществляется через сервер приложений.
- **Уровень логики (Сервер приложений):** Реализован с использованием платформы Node.js. На этом уровне сосредоточена вся бизнес-логика системы: алгоритмы регистрации и учета, правила генерации свидетельств, методы расчета статистики для аналитических справок, а также API для интеграции с существующими автоматизированными системами Заказчика.
- **Уровень представления (Веб-интерфейс):** Разрабатывается с использованием библиотеки React. Обеспечивает русскоязычный, интуитивно понятный интерфейс для взаимодействия пользователей с системой через веб-браузер. Интерфейс выполнен в нейтральной цветовой схеме с использованием шрифта 12pt, содержит сообщения об ошибках с рекомендациями по их устранению.

Общая структура базы данных

База данных системы будет спроектирована в соответствии с принципами нормализации для обеспечения непротиворечивости и целостности данных. Основные сущности и их атрибуты включают:

- **Информация об обучающихся:**
 - Фамилия, имя, отчество (при наличии)
 - Контактные данные (телефон, email)
 - Место работы, должность
 - Дата регистрации в системе
 - Примечания
- **Информация о курсах повышения квалификации:**
 - Наименование курса
 - Краткое описание
 - Продолжительность (в часах)
 - Учебный план (кратко)
- **Информация о группах и процессе обучения:**
 - Связь с курсом
 - Список обучающихся (связь с сущностью “Обучающиеся”)
 - Даты начала и окончания обучения по группе
 - Результаты (оценки, статус “завершил”/“не завершил”)
 - Связь со сгенерированным свидетельством (путь к PDF-файлу)
- **Информация для аналитики:**
 - Хранимые процедуры и представления (views) для агрегации данных по курсам, группам, периодам обучения для последующего формирования отчетов в Excel/PDF.

Решения по режимам функционирования системы

Автоматизированная система будет функционировать в круглосуточном режиме, обеспечивая доступность 99% (за исключением времени регламентных работ). Основные режимы работы:

- **Штатный режим:** Круглосуточная работа системы, выполнение всех функций согласно бизнес-процессам Заказчика. Авторизованные пользователи (до 20 человек) работают с системой через веб-интерфейс в соответствии со своими ролями (просмотр, редактирование).
- **Режим администрирования:** Выполняется администратором БД и администратором интерфейса. Включает в себя настройку системы, управление ролями и правами доступа, мониторинг производительности, проведение регламентных работ.
- **Режим восстановления после сбоя:** Включает процедуры автоматического или ручного восстановления работоспособности системы из резервных копий. Время восстановления не должно превышать 4 часов.
- **Режим резервного копирования:** Ежедневное автоматическое создание резервной копии базы данных с глубиной хранения 7 дней.

Система должна выполнять следующие функции в реальном времени:

- Регистрация обучающихся и зачисление на курсы (время отклика <1 сек);

- Учет прохождения курсов (фиксация дат, часов, оценок);
- Генерация и хранение свидетельств (PDF);
- Формирование аналитических справок по курсам и группам (время формирования отчета не более 5 сек).

Решения по численности, квалификации и функциям персонала

Указанные требования должны удовлетворять требованиям, зафиксированным в техническом задании на разработку системы:

- **Администратор базы данных (1 чел.):** Должен обладать знанием SQL, принципов администрирования PostgreSQL, обеспечения резервного копирования и восстановления.
- **Администратор интерфейса (1 чел.):** Должен обладать знанием React и Node.js для настройки и поддержки веб-интерфейса и сервера приложений.
- **Пользователи (до 20 чел.):** Сотрудники Центра повышения квалификации и Отдела отчетности. Должны обладать базовыми навыками работы с ПК и веб-браузером.

Режим работы персонала: 8-часовой рабочий день, 5 дней в неделю.

Решения по составу программных средств, языкам деятельности, алгоритмам процедур и операций и методам их реализации

- **Системное и сетевое ПО:** Операционная система сервера (Linux Ubuntu Server, NixOS), СУБД PostgreSQL, веб-сервер (Nginx).
- **Средства разработки:**
 - Среда разработки: VS Code.
 - Языки программирования: JavaScript (ES6+), SQL-92.
 - Фреймворки и библиотеки: React для клиентской части, Node.js для серверной части.
- **Инструменты для генерации отчетов:** Библиотеки для работы с Excel и PDF на Node.js, typst.

Алгоритм генерации свидетельства:

1. Пользователь через веб-интерфейс инициирует генерацию свидетельства для конкретного обучающегося по завершенному курсу.
2. Сервер приложений (Node.js) получает запрос и извлекает из базы данных необходимую информацию: ФИО обучающегося, наименование курса, даты начала и окончания обучения, количество часов, оценку.
3. Сервер выбирает соответствующий шаблон свидетельства (например, в формате HTML или DOCX).
4. В шаблон подставляются данные из базы данных.
5. С помощью специализированной библиотеки (например, pdfkit или puppeteer) сформированный документ конвертируется в формат PDF.
6. Готовый PDF-файл сохраняется в файловом хранилище, а ссылка на него записывается в базу данных в карточку обучающегося или группы.
7. Пользователю в интерфейсе отображается ссылка для скачивания готового свидетельства.

Алгоритм формирования аналитической справки по курсу:

1. Пользователь выбирает в интерфейсе интересующий курс и период.
2. Сервер приложений формирует SQL-запрос к базе данных для агрегации данных: общее количество обучающихся на курсе за период, количество успешно завершивших, средний балл, распределение по группам и т.д.
3. Полученные данные структурируются и могут быть представлены в виде таблиц и графиков непосредственно в веб-интерфейсе.
4. При необходимости экспорта, данные форматируются и выгружаются в файл Excel или PDF.

Источники разработки

Данный документ разрабатывался на основании ТЗ, написанного по ГОСТ 34.602—2020 от 9 февраля 2026 г.

СОТАВИЛИ

Должность исполнителя

Подпись

Фамилия, Имя, Отчество

Дата «__» ____ 2026 г.