# ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

## Тема: "Разработка веб-приложения «Студенческий планировщик» для организации учебного процесса"

### Оглавление

1. Введение

2. Аналитический раздел

2.1. Анализ предметной области

2.2. Обзор существующих аналогов

2.3. Выбор технологий разработки

2.4. Требования к системе

3. Проектный раздел

3.1. Архитектура приложения

3.2. Проектирование базы данных

3.3. Проектирование интерфейса пользователя

3.4. Проектирование модулей системы

4. Технологический раздел

4.1. Разработка серверной части

4.2. Разработка клиентской части

4.3. Интеграция компонентов

4.4. Тестирование

5. Руководство пользователя

6. Заключение

7. Список литературы

8. Приложения

## 1. Введение

В современном образовательном процессе студенты сталкиваются с необходимостью эффективно организовывать своё время, управлять заданиями по различным дисциплинам и отслеживать прогресс обучения. Традиционные методы планирования учебной деятельности часто не соответствуют динамике современного образовательного процесса.

Целью данной выпускной квалификационной работы является разработка веб-приложения «Студенческий планировщик», которое поможет студентам эффективно организовать учебный процесс, управлять расписанием занятий, отслеживать задания и контролировать выполнение учебных задач.

Для достижения поставленной цели необходимо решить следующие задачи:

1. Провести анализ требований и потребностей студентов в организации учебного процесса

2. Изучить существующие решения в данной области

3. Спроектировать архитектуру приложения и структуру базы данных

4. Разработать клиентскую и серверную части приложения

5. Обеспечить удобный интерфейс для взаимодействия пользователя с системой

6. Провести тестирование разработанного приложения

Актуальность работы обусловлена растущей потребностью в специализированных инструментах для организации учебного процесса, которые учитывают особенности образовательной деятельности студентов и предлагают удобные способы планирования и отслеживания заданий.

Практическая значимость работы заключается в создании инструмента, который может быть внедрен в образовательный процесс для повышения эффективности обучения студентов.

## 2. Аналитический раздел

### 2.1. Анализ предметной области

Организация учебного процесса для студентов включает в себя несколько ключевых аспектов:

- Управление расписанием занятий

- Отслеживание домашних заданий и проектов

- Планирование подготовки к экзаменам и зачетам

- Приоритезация задач в условиях ограниченного времени

Эффективная организация учебного процесса требует систематического подхода к управлению этими аспектами. Веб-приложение для планирования учебной деятельности должно предоставлять инструменты для работы с каждым из них, а также их интеграцию в единую систему.

### 2.2. Обзор существующих аналогов

На рынке существует несколько решений для планирования и организации деятельности, которые могут быть адаптированы для учебного процесса:

1. \*\*Google Calendar\*\* - календарь с возможностью планирования событий, но с ограниченным функционалом для учебных задач

2. \*\*Trello\*\* - система управления задачами с помощью карточек и досок, не специализированная для учебного процесса

3. \*\*Microsoft Todo\*\* - простой планировщик задач без специфических функций для студентов

4. \*\*MyStudyLife\*\* - специализированное приложение для студентов с ограниченными возможностями управления заданиями

Анализ аналогов показал, что существующие решения либо обладают избыточным общим функционалом, не ориентированным на студентов, либо имеют недостаточный набор возможностей для полноценной организации учебного процесса.

### 2.3. Выбор технологий разработки

Для разработки приложения были выбраны следующие технологии:

\*\*Серверная часть\*\*:

- \*\*Next.js с API Routes\*\* - современный фреймворк для разработки серверной части приложения

- \*\*PostgreSQL\*\* - надежная реляционная база данных для хранения информации о пользователях, занятиях и заданиях

- \*\*JSON Web Tokens (JWT)\*\* - для аутентификации пользователей

- \*\*bcrypt\*\* - для безопасного хранения паролей

\*\*Клиентская часть\*\*:

- \*\*React\*\* - библиотека для построения пользовательских интерфейсов

- \*\*TypeScript\*\* - типизированный JavaScript для повышения качества кода

- \*\*Material-UI\*\* - библиотека компонентов для создания удобного интерфейса

- \*\*FullCalendar\*\* - компонент календаря с богатым функционалом

- \*\*Recharts\*\* - библиотека для создания графиков и диаграмм

Выбор данных технологий обусловлен их современностью, широкой поддержкой сообщества, хорошей документацией и соответствием требованиям проекта.

### 2.4. Требования к системе

\*\*Функциональные требования\*\*:

1. Регистрация и аутентификация пользователей

2. Управление расписанием занятий в различных режимах просмотра (день/неделя/месяц)

3. Создание и категоризация предметов с цветовой маркировкой

4. Добавление и отслеживание заданий, связанных с занятиями

5. Управление списком заданий с возможностью отметки о выполнении и фильтрации

6. Визуализация учебной нагрузки в виде графиков

7. Управление личными задачами с возможностью приоритезации

\*\*Нефункциональные требования\*\*:

1. Интуитивно понятный интерфейс

2. Быстрая загрузка и отзывчивость приложения

3. Безопасное хранение пользовательских данных

4. Масштабируемость системы

5. Поддержка различных устройств и браузеров

## 3. Проектный раздел

### 3.1. Архитектура приложения

Приложение построено на основе клиент-серверной архитектуры с использованием фреймворка Next.js, который объединяет серверную и клиентскую части в единое приложение.

\*\*Компоненты архитектуры\*\*:

- \*\*Клиентская часть\*\* - React-приложение с компонентным подходом

- \*\*Серверная часть\*\* - API Routes в Next.js

- \*\*База данных\*\* - PostgreSQL

- \*\*Аутентификация\*\* - на основе JWT-токенов

- \*\*Middleware\*\* - для проверки аутентификации и управления маршрутизацией

Взаимодействие между клиентом и сервером осуществляется через REST API, что обеспечивает гибкость и масштабируемость системы.

### 3.2. Проектирование базы данных

Структура базы данных включает следующие основные таблицы:

1. \*\*users\*\* - информация о пользователях системы

2. \*\*categories\*\* - категории предметов или занятий

3. \*\*lessons\*\* - информация о занятиях

4. \*\*assignments\*\* - задания, связанные с занятиями

5. \*\*todos\*\* - личные задачи пользователя

Между таблицами установлены следующие связи:

- users (1) --- (\*) categories

- users (1) --- (\*) lessons

- users (1) --- (\*) assignments

- users (1) --- (\*) todos

- categories (1) --- (\*) lessons

- lessons (1) --- (\*) assignments

Для оптимизации запросов созданы индексы:

- `idx\_categories\_user\_id`

- `idx\_lessons\_user\_id`

- `idx\_lessons\_category\_id`

- `idx\_assignments\_user\_id`

- `idx\_assignments\_lesson\_id`

- `idx\_todos\_user\_id`

### 3.3. Проектирование интерфейса пользователя

Интерфейс пользователя спроектирован с учетом принципов UX/UI дизайна и включает следующие основные экраны:

1. \*\*Страница входа и регистрации\*\* - форма для авторизации и создания аккаунта

2. \*\*Дашборд\*\* - главная страница с сводной информацией о занятиях, заданиях и задачах

3. \*\*Календарь\*\* - экран для просмотра и управления расписанием занятий

4. \*\*Список заданий\*\* - интерфейс для управления учебными заданиями с возможностью фильтрации и сортировки

5. \*\*Задачи\*\* - список личных задач с возможностью приоритезации

6. \*\*Категории\*\* - интерфейс управления категориями предметов

Интерфейс разработан с использованием компонентов Material-UI, что обеспечивает современный внешний вид и отзывчивость на различных устройствах.

### 3.4. Проектирование модулей системы

Система разделена на следующие функциональные модули:

1. \*\*Модуль аутентификации\*\* - управление пользователями, регистрация, вход в систему

2. \*\*Модуль календаря\*\* - отображение и управление занятиями, интеграция с FullCalendar

3. \*\*Модуль заданий\*\* - создание, редактирование и отслеживание заданий

4. \*\*Модуль списка задач\*\* - организация, фильтрация и управление статусами заданий

5. \*\*Модуль личных задач\*\* - управление личными задачами пользователя

6. \*\*Модуль аналитики\*\* - визуализация учебной нагрузки и прогресса

Модули взаимодействуют между собой через API и общую базу данных, что обеспечивает целостность данных и эффективное взаимодействие компонентов.

## 4. Технологический раздел

### 4.1. Разработка серверной части

Серверная часть приложения реализована с использованием Next.js API Routes, что позволяет создавать серверные эндпоинты в рамках того же приложения, что и клиентская часть.

Основные компоненты серверной части:

- \*\*Конфигурация базы данных\*\* - настройка подключения к PostgreSQL

- \*\*Репозитории данных\*\* - функции для работы с таблицами базы данных

- \*\*Аутентификация\*\* - обработка регистрации, входа и проверки пользователей

- \*\*API эндпоинты\*\* - обработчики запросов клиентской части

- \*\*Middleware\*\* - промежуточное ПО для проверки авторизации

Серверная часть обеспечивает безопасность данных, проверку входных параметров и валидацию запросов.

### 4.2. Разработка клиентской части

Клиентская часть приложения разработана с использованием React и TypeScript. Компоненты организованы в соответствии с функциональными модулями системы.

Основные компоненты клиентской части:

- \*\*Компоненты аутентификации\*\* - формы входа и регистрации

- \*\*Компоненты календаря\*\* - интеграция с FullCalendar и управление занятиями

- \*\*Компоненты списка заданий\*\* - интерфейс для просмотра и управления заданиями

- \*\*Компоненты задач\*\* - интерфейс для управления личными задачами

- \*\*Компоненты аналитики\*\* - графики и диаграммы на основе Recharts

Клиентская часть обеспечивает интуитивно понятный интерфейс и отзывчивость приложения.

### 4.3. Интеграция компонентов

Интеграция серверной и клиентской частей осуществляется через REST API. Клиентская часть отправляет запросы к серверным эндпоинтам, которые обрабатывают эти запросы и взаимодействуют с базой данных.

Для управления состоянием приложения используется React Context API, что позволяет эффективно передавать данные между компонентами без избыточной передачи через пропсы.

### 4.4. Тестирование

Тестирование приложения включало следующие виды:

- \*\*Функциональное тестирование\*\* - проверка работоспособности основных функций приложения

- \*\*Интеграционное тестирование\*\* - проверка взаимодействия между компонентами

- \*\*Пользовательское тестирование\*\* - тестирование интерфейса реальными пользователями

- \*\*Нагрузочное тестирование\*\* - проверка работы системы при различных уровнях нагрузки

В результате тестирования были выявлены и исправлены ошибки, а также улучшен пользовательский интерфейс на основе обратной связи от пользователей.

## 5. Руководство пользователя

### Регистрация и вход

1. Перейдите на страницу регистрации `/register`

2. Заполните форму, указав имя, email и пароль

3. После регистрации вы будете перенаправлены на страницу входа

4. Введите ваш email и пароль для входа в систему

### Работа с дашбордом

После авторизации вы попадаете на дашборд, который содержит:

- Сводку предстоящих занятий

- Список заданий на выполнение

- График учебной нагрузки

- Список задач и их статус

### Управление категориями предметов

1. Перейдите в раздел "Категории" через боковое меню

2. Нажмите "Добавить категорию"

3. Введите название категории и выберите цвет

4. Для редактирования категории нажмите на иконку редактирования

5. Для удаления категории нажмите на иконку корзины

### Работа с календарем

1. Перейдите в раздел "Календарь" через боковое меню

2. Выберите подходящий вид календаря (день/неделя/месяц)

3. Для добавления занятия кликните по свободному месту в календаре

4. Заполните форму создания занятия (название, время, категория, место проведения)

5. Для редактирования занятия кликните на него в календаре

6. Для удаления занятия откройте его и нажмите кнопку "Удалить"

### Управление заданиями

1. Перейдите в раздел "Задания" или кликните на занятие в календаре

2. Нажмите "Добавить задание"

3. Заполните форму создания задания (название, описание, срок выполнения)

4. Задания можно отмечать как выполненные

5. Используйте фильтры для отображения заданий по категориям или статусу выполнения

### Работа со списком заданий

1. В списке заданий отображаются все учебные задания

2. Используйте фильтры для просмотра заданий по статусу выполнения или категориям

3. Нажмите на задание, чтобы просмотреть его подробности

4. Отмечайте задания как выполненные с помощью чекбокса

5. Для редактирования задания нажмите на иконку редактирования

### Управление личными задачами

1. В разделе "Задачи" отображается список ваших личных задач

2. Добавьте новую задачу, указав название и приоритет

3. Отмечайте задачи как выполненные

4. Удаляйте ненужные задачи

## 6. Заключение

В результате выполнения выпускной квалификационной работы было разработано веб-приложение «Студенческий планировщик», которое предоставляет студентам инструменты для эффективной организации учебного процесса, управления расписанием занятий, отслеживания заданий и контроля выполнения учебных задач.

Основные результаты работы:

1. Проведен анализ предметной области и существующих аналогов

2. Спроектирована архитектура приложения и структура базы данных

3. Разработана серверная часть приложения на основе Next.js и PostgreSQL

4. Реализована клиентская часть с использованием React и Material-UI

5. Создан удобный интерфейс для управления учебным процессом

6. Проведено тестирование приложения и исправлены выявленные ошибки

Приложение может быть использовано студентами различных учебных заведений для повышения эффективности организации учебного процесса.

Перспективы дальнейшего развития проекта включают:

- Разработку мобильного приложения

- Интеграцию с системами управления обучением (LMS)

- Добавление социальных функций для совместной работы студентов

- Расширение аналитических возможностей для оценки прогресса обучения

## 7. Список литературы

1. React – A JavaScript library for building user interfaces. [Электронный ресурс]. – Режим доступа: https://reactjs.org/

2. Next.js by Vercel - The React Framework for the Web. [Электронный ресурс]. – Режим доступа: https://nextjs.org/

3. PostgreSQL: The world's most advanced open source database. [Электронный ресурс]. – Режим доступа: https://www.postgresql.org/

4. Material-UI: A popular React UI framework. [Электронный ресурс]. – Режим доступа: https://mui.com/

5. TypeScript - JavaScript with syntax for types. [Электронный ресурс]. – Режим доступа: https://www.typescriptlang.org/

6. FullCalendar - JavaScript Calendar. [Электронный ресурс]. – Режим доступа: https://fullcalendar.io/

7. Nielsen, J. (2000). Designing Web Usability: The Practice of Simplicity. New Riders Publishing.

8. Fowler, M. (2002). Patterns of Enterprise Application Architecture. Addison-Wesley Professional.

9. Richardson, L. & Ruby, S. (2007). RESTful Web Services. O'Reilly Media.

10. Gamma, E., Helm, R., Johnson, R., & Vlissides, J. (1994). Design Patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley Professional.

## 8. Приложения

### Приложение А. Структура базы данных

#### Таблица users

| Поле | Тип | Описание |

|------|-----|----------|

| id | SERIAL | Первичный ключ |

| name | VARCHAR(255) | Имя пользователя |

| email | VARCHAR(255) | Email (уникальный) |

| password | VARCHAR(255) | Хешированный пароль |

| created\_at | TIMESTAMP | Дата создания аккаунта |

#### Таблица categories

| Поле | Тип | Описание |

|------|-----|----------|

| id | SERIAL | Первичный ключ |

| name | VARCHAR(255) | Название категории |

| color | VARCHAR(50) | Цвет категории (HEX) |

| user\_id | INTEGER | Внешний ключ к users |

| created\_at | TIMESTAMP | Дата создания категории |

#### Таблица lessons

| Поле | Тип | Описание |

|------|-----|----------|

| id | SERIAL | Первичный ключ |

| title | VARCHAR(255) | Название занятия |

| start\_time | TIMESTAMP | Время начала |

| end\_time | TIMESTAMP | Время окончания |

| category\_id | INTEGER | Внешний ключ к categories |

| user\_id | INTEGER | Внешний ключ к users |

| location | VARCHAR(255) | Место проведения |

| description | TEXT | Описание занятия |

| created\_at | TIMESTAMP | Дата создания записи |

#### Таблица assignments

| Поле | Тип | Описание |

|------|-----|----------|

| id | SERIAL | Первичный ключ |

| title | VARCHAR(255) | Название задания |

| description | TEXT | Описание задания |

| due\_date | TIMESTAMP | Срок выполнения |

| completed | BOOLEAN | Статус выполнения (выполнено/не выполнено) |

| lesson\_id | INTEGER | Внешний ключ к lessons |

| user\_id | INTEGER | Внешний ключ к users |

| created\_at | TIMESTAMP | Дата создания записи |

#### Таблица todos

| Поле | Тип | Описание |

|------|-----|----------|

| id | SERIAL | Первичный ключ |

| title | VARCHAR(255) | Название задачи |

| completed | BOOLEAN | Статус выполнения |

| priority | VARCHAR(50) | Приоритет (low/medium/high) |

| user\_id | INTEGER | Внешний ключ к users |

| created\_at | TIMESTAMP | Дата создания записи |