Федеральное государственное автономное образовательное учреждение высшего образования
«Национальный исследовательский университет ИТМО»

Факультет программной инженерии и компьютерной техники

Информационные системы

Курсовая работа

Выполнил: Онишков Валерий Вячеславович
Группа: P3314
Преподаватель: Королёва Юлия Александровна

Санкт-Петербург
2024г.

Содержание

1 Введение

1.1 Назначение

Назначение этого документа - спецификация требований к ПО описывает функциональные и нефункциональные требования к выпуску платформы дневников внеурочной деятельности. Этот документ предназначен для команд, которые будут реализовывать и проверять корректность работы системы.

1.2 Область применения

Платформа предназначена для создания и ведения электронных дневников успеваемости в образовательных организациях, таких как школы, курсы, учебные центры и другие учреждения, предоставляющие образовательные услуги.
Система позволяет преподавателям, администраторам и студентам (или их родителям/опекунам) легко отслеживать учебный процесс. Платформа предоставляет следующие функции: Преподавателям: управление расписанием, добавление оценок, ведение заметок по каждому ученику, анализ успеваемости. Администрации: настройка структуры курсов, распределение преподавателей, отчетность по успеваемости и посещаемости. Студентам (и их представителям): просмотр оценок, расписания, комментариев от преподавателей, а также прогнозирование результатов на основе текущих данных.

1.3 Обзор документа

Разделы Общего описания содержат описание функционала продукта, описание пользователей, влияющие факторы и ограничения, накладываемые на систему.
Спецификации требований содержат описание функциональных и нефункциональных требований к системе, включающих в себя требования к надежности, удобству использования и производительности платформы. Также в разделе описаны ограничения, накладываемые в процессе разработке и описание основных интерфейсов системы.

2 Общее описание

2.1 Функциональность продукта

В контексте данного документа под системой понимается взаимосвязанная структура, обладающая следующим функционалом:

- Нельзя получить доступ к платформе без регистрации
- Регистрация пользователей с помощью почты, придуманного пароля, фамилии и имени. (Пароль должен состоять из 8 символов, в котором обязательно присутствует хотя бы одна заглавная, прописная буква и хотя бы одна цифра).
- Гость имеет право на регистрацию как студент, преподаватель и администратор.
- Вход через логин и пароль
- Возможность восстановления пароля через email, если пользователь его забыл.
- Гость имеет право на просмотр стартовой страницы и страницы регистрации.
- После успешного входа через логин пользователю предоставляется доступ к его персонализированному кабинету на платформе.
- Зарегистрированный пользователь(студент) имеет доступ к просмотру организаций в которых обучается, а так же имеет доступ к своим журналам от данных организаций
- Зарегистрированный пользователь(студент) имеет право на запрос доступа к классам внутри организаций.

- Зарегистрированный пользователь(преподаватель) имеет доступ к просмотру организаций в которых преподает, а так же просмотру и измененю своих журналов.
- Зарегистрированный пользователь(преподаватель) имеет право на принятие и отвержение запроса студента на добавление в класс преподавателя.
- Зарегистрированный пользователь(преподаватель) имеет право на создание журанала в организации к которой присоединился.
- Зарегистрированный пользователь(администратор) имеет доступ к просмотру,изменению и удалению своих журналов и журналов созданных преподавателями в подчинении.
- Зарегистрированный пользователь(администратор) имеет доступ к созданию организаций
- Зарегистрированный пользователь(администратор) имеет право на принятие или отвержение запроса на присоединение к организации от преподавателя
- Зарегистрированный пользователь(администратор) имеет право на просмотр своих организаций
- Зарегистрированный пользователь(администратор) имеет право на создание, удаление и изменение всех журналов в его организации.
- Зарегистрированный пользователь(администратор) имеет право на исключение преподователя из организации
- Платформа используется только в веб версии.
- Любой пользователь имеет право на удаление личного кабинета.
- При удалении аккаутна пользователя(студента) удаляются все его данные, кроме оценок.
- При удалении аккаунта пользователя(преподавателя) удаляются все его данные, кроме журналов.
- При удалении аккаунта(адмиистратор) удаляются все его данные в том числе журналы преподавателей, находящихся в его подчинении.

2.2 Описание пользователей

Не авторизованный пользователь:
Имеет доступ к странице регистрации и главной странице с информацией о проекте

Авторизованный пользователь(студент):
Имеет доступ к главной странице, к страницам с образовательными учреждениям, своим журналам и личному аккаунту

Авторизованный пользователь(преподаватель):
Имеет доступ к главной странице и журналам за которые отвечат и личному аккаунту

Авторизованный профиль(Администратор):
Имеет доступ к главной странице, к своим журналам, к журналам его подчиненых и личному аккаунту

2.3 Влияющие факторы и зависимости

Доступность контента в связи с географическими откланениями

2.4 Ограничения

Географические ограничения: Данная платформа предусматривает только русский язык, поэтому ограничения для стран с другими языками.

3 Спецификация требований

3.1 Функциональные требования

3.1.1

- Система должна предоставлять возможность регистрации на портале для разных ролей.
- Система должна предоставлять возможность к авторизации на портале
- Система должна предоставлять возможность запроса на доступ к организации
- Система должна предоставлять возможность принять и отвергунть запрос на принятие в организацию
- Система должна предоставлять возможность принять и отвергунть запрос на вступление в класс
- Система должна предоставлять доступ к созданию организации
- Система должна предоставлять доступ к созданию журнала
- Система должна предоставлять доступ к изменению журналов для преподавателей и администраторов
- Система должна предоставлять доступ к просмотру и изменению всех оценок и журналов администратору
- Система должна предоставлять доступ для преподавателей к просмоту и изменению своих журналов
- Система должна предоставлять доступ к просмотру своих оценок студенту

3.2 Требования к удобству использования

- Поддержка разрешений экрана для мобильных и десктоп устройств
- Обеспечить корректную работу на различных браузерах(Chrome 79+, Safari 11+, Mozilla 70+, Яндекс Браузер 21+)

3.3 Ограничения разработки

- Backend часть должна быть реализована при помощи Java + Spring Framework
- Frontend часть должна быть реализована на TypeScript + React
- Данные должны хранится в базе данных PostgreSQL

4 Прецеденты

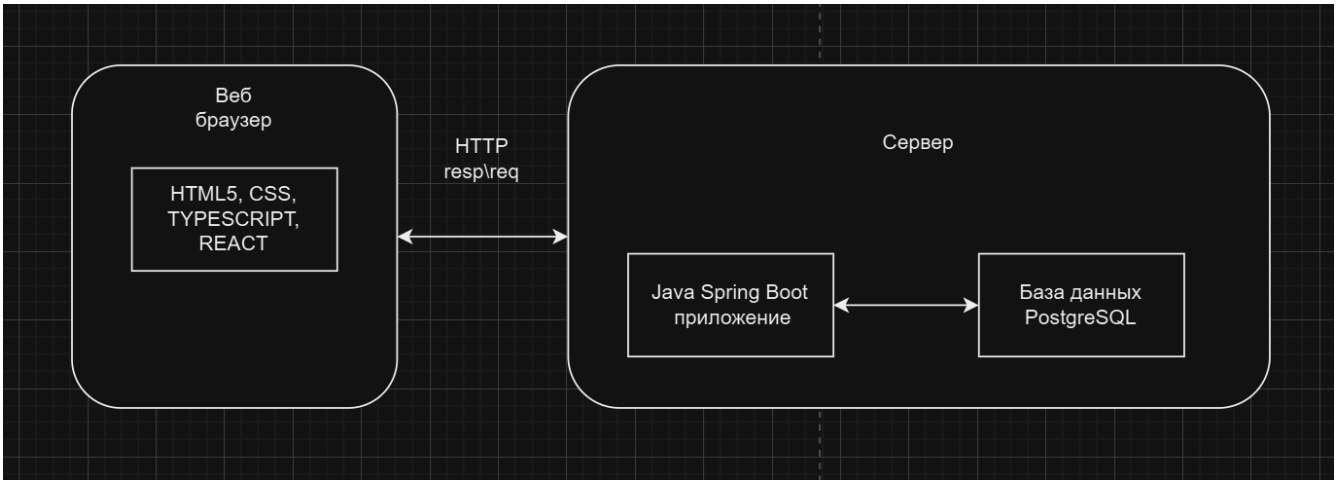
- – Прецендент: Регистрация пользователей
  – ID:1
  – Главный актер:Гость
  – Предусловия: Пользователь находится на главной странице платформы
  – Пользователь переходит на страницу регистрации
  вводит необходимые данные(имя, фамилия, почта, пароль, логин)
  система проверяет данные на корректность и уникальность
  система создает учетную запись и пользователь перходит на страницу
  – Постуслоия: Пользователь может войти в систему под своими учетными данными
- – Прецендент: Отслеживание прогресса обучения в личном кабинете для авторизованного пользователя
  – ID:2
  – Главный актер:Авторизированный пользователь(Студент)

- Предусловия: Пользователь зарегистрирован и авторизован.
- Пользователь переходит в личный кабинет
  пользователь выбирает нужный журнал
- Постуслоия: Пользователь имеет возможность отслеживать прогресс своего обучения.

- Прецендент: Выставление оценок преподавателем
- ID:3
- Главный актер:Авторизированный пользователь(преподаватель)
- Предусловия: Пользователь зарегистрирован и авторизован, а так же принят в огранизацию и создан класс для ведения журнала.
- Пользователь переходит в личный кабинет
  пользователь выбирает нужный журнал
  пользователь выставляет оценку нужному студенту
- Постуслоия: Пользователь имеет возможность изменять и добавлять оценки студентов

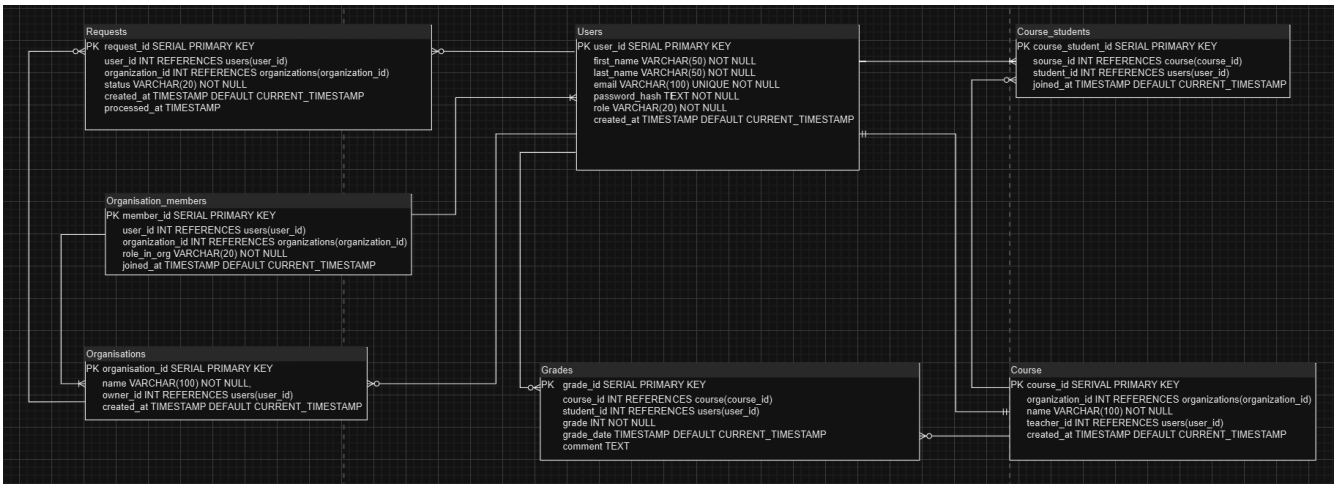
- Прецендент: Создание организации
- ID:4
- Главный актер:Авторизированный пользователь(администратор)
- Предусловия: Пользователь зарегистрирован и авторизован, находится на главной странице
- Пользователь переходит в страницу создания организации
  пользователь вводит все нужные данные(Название)
- Постуслоия: Пользователь имеет доступ к созданию, изменеию и удалению журналов, а так же принимать преподавателей в организацию.

5 Архитектура компонентов



6 База данных

6.1 UML диаграмма



6.2 Создание базы данных

```
CREATE TABLE users (
    user_id SERIAL PRIMARY KEY,
    first_name VARCHAR(50) NOT NULL,
    last_name VARCHAR(50) NOT NULL,
    email VARCHAR(100) UNIQUE NOT NULL,
    password_hash TEXT NOT NULL,
    role VARCHAR(20) NOT NULL,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

CREATE TABLE organizations (
    organization_id SERIAL PRIMARY KEY,
    name VARCHAR(100) NOT NULL,
    owner_id INT REFERENCES users(user_id) ON DELETE CASCADE,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

CREATE TABLE organization_members (
    member_id SERIAL PRIMARY KEY,
    user_id INT REFERENCES users(user_id) ON DELETE CASCADE,
    organization_id INT REFERENCES organizations(organization_id) ON DELETE CASCADE,
    role_in_org VARCHAR(20) NOT NULL,
    joined_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

CREATE TABLE Course (
    course_id SERIAL PRIMARY KEY,
    organization_id INT REFERENCES organizations(organization_id) ON DELETE CASCADE,
    name VARCHAR(100) NOT NULL,
    teacher_id INT REFERENCES users(user_id),
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

```
CREATE TABLE course_students (
    course_student_id SERIAL PRIMARY KEY,
    subject_id INT REFERENCES subjects(subject_id) ON DELETE CASCADE,
    student_id INT REFERENCES users(user_id) ON DELETE CASCADE,
    joined_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

CREATE TABLE grades (
    grade_id SERIAL PRIMARY KEY,
    subject_id INT REFERENCES subjects(subject_id) ON DELETE CASCADE,
    student_id INT REFERENCES users(user_id) ON DELETE CASCADE,
    teacher_id INT REFERENCES users(user_id),
    grade NUMERIC(3, 1) NOT NULL,
    grade_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    comment TEXT
);

CREATE TABLE requests (
    request_id SERIAL PRIMARY KEY,
    user_id INT REFERENCES users(user_id) ON DELETE CASCADE,
    organization_id INT REFERENCES organizations(organization_id),
    subject_id INT REFERENCES subjects(subject_id),
    status VARCHAR(20) NOT NULL CHECK,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    processed_at TIMESTAMP
);
```

6.3 Добавление данных

```
INSERT INTO users (first_name, last_name, email, password_hash, role)
VALUES
    ('Admin', 'Adminov', 'admin@example.com', 'hashed_admin_password', 'owner'),
    ('Ivan', 'Petrov', 'ivan.petrov@example.com', 'hashed_teacher_password', 'teacher'),
    ('Anna', 'Ivanova', 'anna.ivanova@example.com', 'hashed_student_password1', 'student
    ('Oleg', 'Sidorov', 'oleg.sidorov@example.com', 'hashed_student_password2', 'student


INSERT INTO organizations (name, owner_id)
VALUES
    ('School No.1', 1);


INSERT INTO organization_members (user_id, organization_id, role_in_org)
VALUES
    (2, 1, 'teacher'), — Ivan Petrov
    (3, 1, 'student'), — Anna Ivanova
    (4, 1, 'student'); — Oleg Sidorov

INSERT INTO subjects (organization_id, name, teacher_id)
VALUES
    (1, 'Mathematics', 2), — Ivan Petrov teaches Mathematics
    (1, 'Physics', 2); — Ivan Petrov teaches Physics


INSERT INTO subject_students (subject_id, student_id)
```

```
VALUES
    (1, 3), -- Anna Ivanova is enrolled in Mathematics
    (1, 4), -- Oleg Sidorov is enrolled in Mathematics
    (2, 3), -- Anna Ivanova is enrolled in Physics
    (2, 4); -- Oleg Sidorov is enrolled in Physics

-- Grades in Mathematics
INSERT INTO grades (subject_id, student_id, teacher_id, grade, comment)
VALUES
    (1, 3, 2, 85.0, 'Good performance on the test'), -- Anna Ivanova
    (1, 4, 2, 90.0, 'Excellent work'); -- Oleg Sidorov

-- Grades in Physics
INSERT INTO grades (subject_id, student_id, teacher_id, grade, comment)
VALUES
    (2, 3, 2, 78.0, 'Solid work, but some mistakes'), -- Anna Ivanova
    (2, 4, 2, 88.0, 'Very good effort'); -- Oleg Sidorov

-- Request from student to join the organization
INSERT INTO requests (user_id, organization_id, status)
VALUES
    (3, 1, 'approved'), -- Anna Ivanova
    (4, 1, 'approved'); -- Oleg Sidorov

-- Request to join a subject
INSERT INTO requests (user_id, subject_id, status)
VALUES
    (3, 1, 'approved'), -- Anna Ivanova joins Mathematics
    (4, 2, 'approved'); -- Oleg Sidorov joins Physics
```

6.4 Индексы базы данных

```
CREATE INDEX idx_users_email ON users(email);
CREATE INDEX idx_users_role ON users(role);

CREATE INDEX idx_organization_members_user_id ON organization_members(user_id);
CREATE INDEX idx_organization_members_organization_id ON\\ organization_members(organizat

CREATE INDEX idx_course_id ON course(name);
CREATE INDEX idx_course_organisation_id ON course(organisation_id);
CREATE INDEX idx_course_teacher_id ON course(teacher_id);
```

6.5 Триггеры для базы данных

Триггер для проверки на правильность введеной оценки

```
CREATE OR REPLACE FUNCTION check_grade_value()
RETURNS TRIGGER AS $$
BEGIN
    IF NEW.grade < 0 OR NEW.grade > 100 THEN
        RAISE EXCEPTION 'Grade must be between 0 and 100';
    END IF;
    RETURN NEW;
END;
$$ LANGUAGE plpgsql;
```

```
CREATE TRIGGER trg_check_grade
BEFORE INSERT OR UPDATE ON grades
FOR EACH ROW
EXECUTE FUNCTION check_grade_value();
```

Триггер для обновления данных о времени принятия решения по запросу на добавление в организацию

```
CREATE OR REPLACE FUNCTION set_processed_at()
RETURNS TRIGGER AS $$
BEGIN
    IF NEW.status IN ('approved', 'rejected') THEN
        NEW.processed_at = CURRENT_TIMESTAMP;
    END IF;
    RETURN NEW;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER trg_set_processed_at
BEFORE UPDATE ON requests
FOR EACH ROW
EXECUTE FUNCTION set_processed_at();
```