

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«САРАТОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
ИМЕНИ ГАГАРИНА Ю.А.»

Кафедра «Естественные и математические науки»

Специальность ПИНЖ — Программная инженерия

Текст программы

Информационной системы «Личный кабинет
образовательной организации. Подсистема
студенческого кабинета»

Выполнил: студент X курса
учебной группы XXXX
очной формы обучения
XXXXXXX

Проверил: преподаватель кафедры
ЕМН XXXXXX

СОДЕРЖАНИЕ

Код MySQL	3
Код – base.html	34
Код – profile.html	34
Код – portfolio.html	44
Код – app.py	87

Код MySQL

-- Создание базы данных

```
CREATE DATABASE IF NOT EXISTS student_SSTU;  
USE student_SSTU;
```

-- форма обучения

```
CREATE TABLE IF NOT EXISTS `Form_study`(  
    id INT AUTO_INCREMENT PRIMARY KEY,  
    title VARCHAR(50) NOT NULL UNIQUE  
)
```

-- Специальности

```
CREATE TABLE IF NOT EXISTS `Specialty`(  
    id INT AUTO_INCREMENT PRIMARY KEY,  
    title VARCHAR(50) NOT NULL UNIQUE COMMENT "Полное название  
специальности",  
    cut VARCHAR(50) NOT NULL UNIQUE COMMENT "Сокращение",  
    cod_specialty VARCHAR(20) NOT NULL COMMENT 'Код специальности'  
)
```

```
CREATE TABLE IF NOT EXISTS `tutor` (
```

```
    id INT AUTO_INCREMENT PRIMARY KEY,
```

-- Личные данные

```
    surname VARCHAR(50) NOT NULL COMMENT 'Фамилия',  
    first_name VARCHAR(50) NOT NULL COMMENT 'Имя',  
    middle_name VARCHAR(50) COMMENT 'Отчество'
```

```
);
```

-- Таблица студенческих групп

```
CREATE TABLE IF NOT EXISTS `Student_group`(
    id INT AUTO_INCREMENT PRIMARY KEY,
    naming VARCHAR(20) NOT NULL COMMENT 'Название группы',
    specialty_id INT NOT NULL COMMENT "Специальность",
    form_study_id INT NOT NULL COMMENT "Форма обучения",
    tutor_id INT NOT NULL COMMENT "Куратор",
    course INT NOT NULL COMMENT "Курс",
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (specialty_id) REFERENCES `Specialty`(id),
    FOREIGN KEY (form_study_id) REFERENCES `Form_study`(id),
    FOREIGN KEY (tutor_id) REFERENCES `tutor`(id)
);
```

-- Таблица студентов (основная информация)

```
CREATE TABLE IF NOT EXISTS `Student`(
    id INT AUTO_INCREMENT PRIMARY KEY,
    -- Личные данные
    surname VARCHAR(50) NOT NULL COMMENT 'Фамилия',
    first_name VARCHAR(50) NOT NULL COMMENT 'Имя',
    middle_name VARCHAR(50) COMMENT 'Отчество',
    birth_date DATE NOT NULL COMMENT 'Дата рождения',
    -- Учебная информация
    student_group_id INT NOT NULL COMMENT 'Группа в которой учится
    студент',
```

```
gradebook_number VARCHAR(20) NOT NULL UNIQUE COMMENT  
'Зачётка',
```

```
-- Контакты
```

```
phone VARCHAR(20) NOT NULL UNIQUE COMMENT 'Номер  
телефона',
```

```
email VARCHAR(100) NOT NULL UNIQUE COMMENT 'Почта',
```

```
address TEXT NOT NULL COMMENT 'Адрес',
```

```
-- Безопасность
```

```
student_password VARCHAR(255) NOT NULL COMMENT 'Хэш пароля',
```

```
profile_photo VARCHAR(255) COMMENT 'Путь к фото профиля',
```

```
-- Технические поля
```

```
created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
```

```
updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON UPDATE  
CURRENT_TIMESTAMP,
```

```
-- создаем связи
```

```
-- ON DELETE CASCADE -- если удалиться форма обучения, удаляться  
студенты с этой формой обучения
```

```
FOREIGN KEY (student_group_id) REFERENCES `Student_group`(id)  
);
```

```
-- Категории деятельности (Научная, Общественная, Культурная, Спортивная)  
CREATE TABLE IF NOT EXISTS `Activity_Category` (
```

```
id INT AUTO_INCREMENT PRIMARY KEY,
```

```
naming VARCHAR(100) NOT NULL COMMENT 'Название категории (напр.  
Научно-исследовательская)',
```

```
cod VARCHAR(50) NOT NULL UNIQUE COMMENT 'Код для системы  
(science, social, cultural, sport)'  
);
```

```
-- справочник уровней мероприятия  
CREATE TABLE IF NOT EXISTS `level_type`(  
    id INT AUTO_INCREMENT PRIMARY KEY,  
    title VARCHAR(50) NOT NULL UNIQUE  
);
```

-- Справочник критериев (Сюда заносятся данные из PDF таблиц)

```
CREATE TABLE IF NOT EXISTS `Rating_Criteria` (  
    id INT AUTO_INCREMENT PRIMARY KEY,  
    category_id INT NOT NULL,
```

-- Описание критерия
section_naming VARCHAR(255) COMMENT 'Название раздела (напр.
Предметные олимпиады)',
description_text VARCHAR(255) NOT NULL COMMENT 'Конкретное
достижение (напр. Гран-при, 1 место, Участие)',

-- Уровень мероприятия (из PDF часто влияет на балл)
level_type_id INT NOT NULL COMMENT "('university', 'city', 'regional',
'federal', 'international', 'other')",

-- Баллы
points INT NOT NULL DEFAULT 0 COMMENT 'Количество баллов за
единицу (из PDF)',

```
FOREIGN KEY (category_id) REFERENCES `Activity_Category`(id) ON  
DELETE CASCADE,
```

```
FOREIGN KEY (level_type_id) REFERENCES `level_type`(id) ON DELETE  
CASCADE
```

```
) COMMENT 'Таблица правил начисления баллов';
```

-- Семестры (чтобы разделять рейтинги по периодам)

```
CREATE TABLE IF NOT EXISTS `Academic_Period` (
```

```
    id INT AUTO_INCREMENT PRIMARY KEY,
```

```
    naming VARCHAR(50) NOT NULL COMMENT 'Например: Осенний  
семестр 2023',
```

```
    start_date DATE NOT NULL,
```

```
    end_date DATE NOT NULL
```

```
);
```

-- Достижения студента (Связь Студента и Критерия)

```
CREATE TABLE IF NOT EXISTS `Student_Achievement` (
```

```
    id INT AUTO_INCREMENT PRIMARY KEY,
```

```
    student_id INT NOT NULL,
```

```
    criteria_id INT NOT NULL,
```

```
    period_id INT NOT NULL,
```

-- Количественные данные

```
    quantity INT DEFAULT 1 COMMENT 'Количество (если студент ввел 2  
статьи)',
```

-- Подтверждение

```
    document_title VARCHAR(255) COMMENT 'Название документа (грамота  
№...)',
```

```
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,  
  
    FOREIGN KEY (student_id) REFERENCES `Student`(id) ON DELETE  
CASCADE,  
    FOREIGN KEY (criteria_id) REFERENCES `Rating_Criteria`(id),  
    FOREIGN KEY (period_id) REFERENCES `Academic_Period`(id)  
);
```

```
CREATE OR REPLACE VIEW Student_Rating_Summary AS  
SELECT
```

```
    s.id AS student_id,  
    s.surname,  
    s.first_name,  
    s.student_group_id,  
    ap.id AS period_id,  
    ap.naming AS period_naming,  
    cat.naming AS category_naming,  
    SUM(rc.points * sa.quantity) AS total_points  
FROM Student s  
JOIN Student_Achievement sa ON s.id = sa.student_id  
JOIN Rating_Criteria rc ON sa.criteria_id = rc.id  
JOIN Activity_Category cat ON rc.category_id = cat.id  
JOIN Academic_Period ap ON sa.period_id = ap.id  
GROUP BY  
    s.id, s.surname, s.first_name, s.student_group_id,  
    ap.id, ap.naming, cat.id, cat.naming;
```

```
INSERT INTO Form_study (title) VALUES  
    ('Очная'),  
    ('Очно-заочная'),  
    ('Заочная'),  
    ('Сокращенная форма');
```

```
INSERT INTO Specialty (title, cut, cod_specialty) VALUES  
    ('Информатика и вычислительная техника', 'ИВЧТ', '09.03.01'),  
    ('Программная инженерия', 'ПИНЖ', '09.03.04'),  
    ('Информационные системы и технологии', 'ИСТ', '09.03.02');
```

```
INSERT INTO tutor (surname, first_name, middle_name) VALUES  
    ('Иванов', 'Пётр', 'Сергеевич'),  
    ('Сидорова', 'Мария', 'Александровна'),  
    ('Кузнецов', 'Илья', 'Геннадьевич');
```

```
INSERT INTO Student_group (naming, specialty_id, form_study_id, tutor_id, course) VALUES  
    ('ИВЧТ-11', 1, 1, 1, 1),  
    ('ПИНЖ-21', 2, 1, 2, 2),  
    ('ИСТ-31', 3, 3, 3, 3);
```

```
INSERT INTO Student (surname, first_name, middle_name, birth_date, student_group_id, gradebook_number, phone, email, address, student_password) VALUES  
    ('Петров', 'Алексей', 'Дмитриевич', '2005-01-12', 1, '12345', '+79170000001', 'petrov@example.com', 'ул. Ленина, 10', '123'),  
    ('Соколова', 'Елена', 'Игоревна', '2004-05-22', 1, '12346', '+79170000002', 'sokolova@example.com', 'ул. Пушкина, 15', '123'),
```

('Орлов', 'Никита', 'Павлович', '2003-11-02', 2, '22347', '+79170000003', 'orlov@example.com', 'ул. Гагарина, 20', '123');

-- Курс 1 (Группа ИВТ-11, ID 1)

```
INSERT INTO Student (surname, first_name, middle_name, birth_date, student_group_id, gradebook_number, phone, email, address, student_password) VALUES
```

('Громов', 'Антон', 'Викторович', '2005-03-01', 1, 'Z12347', '+79170000004', 'gromov@uni.ru', 'ул. Мира, 5', '123');

-- Курс 2 (Группа ПИ-21, ID 2)

```
INSERT INTO Student (surname, first_name, middle_name, birth_date, student_group_id, gradebook_number, phone, email, address, student_password) VALUES
```

('Зайцева', 'Виктория', 'Олеговна', '2004-09-10', 2, 'Z22348', '+79170000005', 'zaytseva@uni.ru', 'пр. Победы, 12', '123');

-- Курс 3 (Группа ИСТ-31, ID 3)

```
INSERT INTO Student (surname, first_name, middle_name, birth_date, student_group_id, gradebook_number, phone, email, address, student_password) VALUES
```

('Федоров', 'Павел', 'Андреевич', '2003-06-25', 3, 'Z32349', '+79170000006', 'fedorov@uni.ru', 'ул. Заводская, 3', '123');

```
INSERT INTO Activity_Category (naming, cod) VALUES
```

('Научная деятельность', 'science'), -- ID 1

('Культурная деятельность', 'cultural'), -- ID 2

('Общественная деятельность', 'social'), -- ID 3

('Спортивная деятельность', 'sport'); -- ID 4

```
INSERT INTO level_type (title) VALUES
('Университетский'), -- ID 1
('Городской'), -- ID 2
('Региональный'), -- ID 3
('Федеральный'), -- ID 4
('Международный'); -- ID 5
```

```
--
```

```
=====
```

-- 1. Заполняем критерии для НАУЧНОЙ деятельности (category_id = 1)

-- Очищен description_text от указаний на уровень

```
--
```

```
=====
```

```
INSERT INTO Rating_Criteria (category_id, section_naming, description_text,
level_type_id, points) VALUES
```

-- Участие в олимпиадах

(1, 'Участие в олимпиадах', 'Участие в предметных олимпиадах, конкурсах курсовых и иных работ', 1, 1),

(1, 'Участие в олимпиадах', 'Участие в предметных олимпиадах, конкурсах курсовых и иных работ', 4, 2),

(1, 'Участие в олимпиадах', 'Участие в международных экзаменах PET, FCE, TOEFL с получением сертификата', 5, 3),

-- Призовые места олимпиад

-- Уровни: Университетский (1), Федеральный (4), Международный (5)

(1, 'Призовые места олимпиад', 'Гран-при', 1, 4),

(1, 'Призовые места олимпиад', '1 место', 1, 3),

(1, 'Призовые места олимпиад', '2 место', 1, 2),

(1, 'Призовые места олимпиад', '3 место', 1, 1),

(1, 'Призовые места олимпиад', 'Гран-при', 4, 5),

(1, 'Призовые места олимпиад', '1 место', 4, 4),

(1, 'Призовые места олимпиад', '2 место', 4, 3),

(1, 'Призовые места олимпиад', '3 место', 4, 2),

(1, 'Призовые места олимпиад', 'Гран-при', 5, 6),

(1, 'Призовые места олимпиад', '1 место', 5, 5),

(1, 'Призовые места олимпиад', '2 место', 5, 4),

(1, 'Призовые места олимпиад', '3 место', 5, 3),

-- Участие в конференциях с докладом

-- Уровни: Университетский (1), Городской (2), Региональный (3),
Федеральный (4), Международный (5)

(1, 'Конференции', 'Участие с докладом в конференции', 1, 3),

(1, 'Конференции', 'Участие с докладом в конференции', 1, 4), -- Межвуз

(1, 'Конференции', 'Участие с докладом в конференции', 2, 5),

(1, 'Конференции', 'Участие с докладом в конференции', 3, 6), -- Областная

(1, 'Конференции', 'Участие с докладом в конференции', 3, 7), -- Региональная

(1, 'Конференции', 'Участие с докладом в конференции', 4, 8),

(1, 'Конференции', 'Участие с докладом в конференции', 5, 9),

-- Призовые места на конференциях

(1, 'Призовые места конференций', 'Гран-при', 1, 4),

(1, 'Призовые места конференций', '1 место', 1, 3),

(1, 'Призовые места конференций', '2 место', 1, 2),

(1, 'Призовые места конференций', '3 место', 1, 1),

(1, 'Призовые места конференций', 'Специальная грамота', 1, 2),

(1, 'Призовые места конференций', 'Гран-при', 2, 6),
(1, 'Призовые места конференций', '1 место', 2, 5),
(1, 'Призовые места конференций', '2 место', 2, 4),
(1, 'Призовые места конференций', '3 место', 2, 3),
(1, 'Призовые места конференций', 'Специальная грамота', 2, 4),

(1, 'Призовые места конференций', 'Гран-при', 3, 7),
(1, 'Призовые места конференций', '1 место', 3, 6),
(1, 'Призовые места конференций', '2 место', 3, 5),
(1, 'Призовые места конференций', '3 место', 3, 4),
(1, 'Призовые места конференций', 'Специальная грамота', 3, 5),

(1, 'Призовые места конференций', 'Гран-при', 4, 9),
(1, 'Призовые места конференций', '1 место', 4, 8),
(1, 'Призовые места конференций', '2 место', 4, 7),
(1, 'Призовые места конференций', '3 место', 4, 6),
(1, 'Призовые места конференций', 'Специальная грамота', 4, 7),

(1, 'Призовые места конференций', 'Гран-при', 5, 10),
(1, 'Призовые места конференций', '1 место', 5, 9),
(1, 'Призовые места конференций', '2 место', 5, 8),
(1, 'Призовые места конференций', '3 место', 5, 7),
(1, 'Призовые места конференций', 'Специальная грамота', 5, 8),

-- Участие в научных семинарах

(1, 'Научные семинары', 'Участие в научном семинаре', 1, 3),
(1, 'Научные семинары', 'Участие в научном семинаре', 1, 4), -- Межвуз
(1, 'Научные семинары', 'Участие в научном семинаре', 2, 5),
(1, 'Научные семинары', 'Участие в научном семинаре', 3, 6), -- Областной
(1, 'Научные семинары', 'Участие в научном семинаре', 3, 7), -- Региональный

(1, 'Научные семинары', 'Участие в научном семинаре', 4, 8),

(1, 'Научные семинары', 'Участие в научном семинаре', 5, 9),

-- Публикации статей

(1, 'Публикации', 'Публикация в изданиях (ВАК)', 4, 10),

(1, 'Публикации', 'Публикация в зарубежных изданиях', 5, 10),

(1, 'Публикации', 'Публикация в изданиях', 4, 6), -- Всероссийские

(1, 'Публикации', 'Публикация в изданиях', 3, 5), -- Региональные

(1, 'Публикации', 'Публикация в изданиях', 2, 4), -- Городские

(1, 'Публикации', 'Публикация в изданиях', 1, 3), -- Вузовские

-- Изобретательская деятельность (Оставляем как есть, т.к. названия специфичны)

(1, 'Изобретательская деятельность', 'Участие в изобретательской деятельности', 1, 3),

(1, 'Изобретательская деятельность', 'Получение промышленного образца РФ', 4, 10),

(1, 'Изобретательская деятельность', 'Получение патента на изобретение', 4, 8),

(1, 'Изобретательская деятельность', 'Получение полезной модели', 4, 6),

-- Конкурсы научных работ (места)

(1, 'Конкурсы научных работ', 'Участие в конкурсе научных работ (проектов)', 1, 3),

(1, 'Конкурсы научных работ', 'Гран-при', 1, 6),

(1, 'Конкурсы научных работ', '1 место', 1, 5),

(1, 'Конкурсы научных работ', '2 место', 1, 4),

(1, 'Конкурсы научных работ', '3 место', 1, 3),

(1, 'Конкурсы научных работ', 'Специальная грамота', 1, 4),

-- Участие в выставках

(1, 'Выставки', 'Участие в выставке', 1, 3),
(1, 'Выставки', 'Участие в выставке', 1, 4), -- Межвуз
(1, 'Выставки', 'Участие в выставке', 2, 5),
(1, 'Выставки', 'Участие в выставке', 3, 6), -- Областная
(1, 'Выставки', 'Участие в выставке', 3, 7), -- Региональная
(1, 'Выставки', 'Участие в выставке', 4, 8),
(1, 'Выставки', 'Участие в выставке', 5, 9),

-- Призовые места на выставках

(1, 'Призовые места выставок', 'Гран-при', 1, 6),
(1, 'Призовые места выставок', '1 место', 1, 5),
(1, 'Призовые места выставок', '2 место', 1, 4),
(1, 'Призовые места выставок', '3 место', 1, 3),
(1, 'Призовые места выставок', 'Специальная грамота', 1, 4),

(1, 'Призовые места выставок', 'Гран-при', 2, 8),
(1, 'Призовые места выставок', '1 место', 2, 7),
(1, 'Призовые места выставок', '2 место', 2, 6),
(1, 'Призовые места выставок', '3 место', 2, 5),
(1, 'Призовые места выставок', 'Специальная грамота', 2, 6),

-- Прочая научная деятельность (Оставляем как есть)

(1, 'Прочая научная деятельность', 'Участие в написании заявки на научный грант', 1, 4),
(1, 'Прочая научная деятельность', 'Получение международной стипендии/премии', 5, 10),
(1, 'Прочая научная деятельность', 'Получение федеральной стипендии/премии', 4, 8),
(1, 'Прочая научная деятельность', 'Получение региональной стипендии/премии', 3, 6),

(1, 'Прочая научная деятельность', 'Член студенческого научно-технического общества (СНТО)', 1, 3);

--

=====

-- 2. Заполняем критерии для КУЛЬТУРНОЙ деятельности (category_id = 2)

-- Очищен description_text от указаний на уровень

--

=====

```
INSERT INTO Rating_Criteria (category_id, section_naming, description_text, level_type_id, points) VALUES
```

-- Культурно-массовая деятельность

(2, 'Участие в творчестве', 'Участник творческого коллектива СГТУ', 1, 3),

-- Участие в смотрах художественной самодеятельности

(2, 'Смотры художественной самодеятельности', 'Участие в смотрах художественной самодеятельности', 1, 3),

(2, 'Смотры художественной самодеятельности', 'Участие в смотрах художественной самодеятельности', 2, 4),

(2, 'Смотры художественной самодеятельности', 'Участие в смотрах художественной самодеятельности', 3, 5),

(2, 'Смотры художественной самодеятельности', 'Участие в смотрах художественной самодеятельности', 4, 6),

(2, 'Смотры художественной самодеятельности', 'Участие в смотрах художественной самодеятельности', 5, 7),

-- Призовые места на смотрах

(2, 'Призовые места смотров', 'Гран-при', 1, 5),

(2, 'Призовые места смотров', '1 место', 1, 4),

- (2, 'Призовые места смотров', '2 место', 1, 3),
(2, 'Призовые места смотров', '3 место', 1, 2),
(2, 'Призовые места смотров', 'Специальная грамота', 1, 3),
- (2, 'Призовые места смотров', 'Гран-при', 2, 6),
(2, 'Призовые места смотров', '1 место', 2, 5),
(2, 'Призовые места смотров', '2 место', 2, 4),
(2, 'Призовые места смотров', '3 место', 2, 3),
(2, 'Призовые места смотров', 'Специальная грамота', 2, 4),
- (2, 'Призовые места смотров', 'Гран-при', 3, 7),
(2, 'Призовые места смотров', '1 место', 3, 6),
(2, 'Призовые места смотров', '2 место', 3, 5),
(2, 'Призовые места смотров', '3 место', 3, 4),
(2, 'Призовые места смотров', 'Специальная грамота', 3, 5),
- (2, 'Призовые места смотров', 'Гран-при', 4, 9),
(2, 'Призовые места смотров', '1 место', 4, 8),
(2, 'Призовые места смотров', '2 место', 4, 7),
(2, 'Призовые места смотров', '3 место', 4, 6),
(2, 'Призовые места смотров', 'Специальная грамота', 4, 7),
- (2, 'Призовые места смотров', 'Гран-при', 5, 10),
(2, 'Призовые места смотров', '1 место', 5, 9),
(2, 'Призовые места смотров', '2 место', 5, 8),
(2, 'Призовые места смотров', '3 место', 5, 7),
(2, 'Призовые места смотров', 'Специальная грамота', 5, 8),

-- Организация мероприятий (Оставляем как есть)

(2, 'Организация мероприятий', 'Участие в организации художественных выставок', 1, 3),

(2, 'Организация мероприятий', 'Участие в организации факультетских праздников', 1, 3);

--

-- 3. Заполняем критерии для ОБЩЕСТВЕННОЙ деятельности (category_id = 3)

-- ИСПРАВЛЕН ID (было 4, стало 3) и ОЧИЩЕН description_text

--

INSERT INTO Rating_Criteria (category_id, section_naming, description_text, level_type_id, points) VALUES

-- Волонтерство

(3, 'Волонтерство', 'Участник волонтерского движения', 1, 3), -- Факультет

(3, 'Волонтерство', 'Участник волонтерского движения', 1, 4), -- ВУЗ

(3, 'Волонтерство', 'Участник волонтерского движения', 2, 5),

(3, 'Волонтерство', 'Участник волонтерского движения', 3, 6),

(3, 'Волонтерство', 'Участник волонтерского движения', 4, 7),

(3, 'Волонтерство', 'Участник волонтерского движения', 5, 8),

-- Член профсоюзных объединений

(3, 'Профсоюзная деятельность', 'Член профсоюзных объединений', 1, 3),

(3, 'Профсоюзная деятельность', 'Член профсоюзных объединений', 1, 4),

(3, 'Профсоюзная деятельность', 'Член профсоюзных объединений', 2, 5),

(3, 'Профсоюзная деятельность', 'Член профсоюзных объединений', 3, 6),

(3, 'Профсоюзная деятельность', 'Член профсоюзных объединений', 4, 7),

(3, 'Профсоюзная деятельность', 'Член профсоюзных объединений', 5, 8),

-- Участие в волонтерских группах

(3, 'Волонтерские работы', 'Участие в волонтерских работах', 1, 3),

(3, 'Волонтерские работы', 'Участие в волонтерских работах', 1, 4), -- Межвуз

(3, 'Волонтерские работы', 'Участие в волонтерских работах', 2, 5),

(3, 'Волонтерские работы', 'Участие в волонтерских работах', 3, 6),

(3, 'Волонтерские работы', 'Участие в волонтерских работах', 4, 7),

(3, 'Волонтерские работы', 'Участие в волонтерских работах', 5, 8),

-- Участие в профориентационных мероприятиях

(3, 'Профориентация', 'Участие в профориентационных мероприятиях', 1, 3),

(3, 'Профориентация', 'Участие в профориентационных мероприятиях', 1, 4), -- ВУЗ

(3, 'Профориентация', 'Участие в профориентационных мероприятиях', 2, 5),

(3, 'Профориентация', 'Участие в профориентационных мероприятиях', 3, 6),

(3, 'Профориентация', 'Участие в профориентационных мероприятиях', 4, 7),

(3, 'Профориентация', 'Участие в профориентационных мероприятиях', 5, 8),

-- Редакторская деятельность (Оставляем как есть, это конкретные роли)

(3, 'Редакторская деятельность', 'Главный редактор факультетской стенгазеты', 1, 3),

(3, 'Редакторская деятельность', 'Член факультетской стенгазеты', 1, 3),

(3, 'Редакторская деятельность', 'Подготовка материалов для вузовских СМИ', 1, 3),

-- Общественные поручения (Оставляем как есть, это конкретные роли)

(3, 'Общественные поручения', 'Член студсовета общежития', 1, 5),

(3, 'Общественные поручения', 'Староста этажа общежития', 1, 6),

(3, 'Общественные поручения', 'Председатель студсовета общежития', 1, 10),

(3, 'Общественные поручения', 'Член Совета студентов и аспирантов (CCA)', 1, 5),

(3, 'Общественные поручения', 'Председатель Совета студентов и аспирантов (CCA)', 1, 10),

(3, 'Общественные поручения', 'Староста студенческой группы', 1, 6),

(3, 'Общественные поручения', 'Профорг студенческой группы', 1, 3),

(3, 'Общественные поручения', 'Культурорганизатор факультета', 1, 10),

(3, 'Общественные поручения', 'Председатель профбюро факультета', 1, 9),

(3, 'Общественные поручения', 'Член профбюро факультета', 1, 6),

-- Посещение мероприятий

(3, 'Посещение мероприятий', 'Посещение мероприятий в составе группы', 1, 1), -- В рамках вуза

(3, 'Посещение мероприятий', 'Посещение мероприятий в составе группы', 1, 3); -- Вне вуза

--

=====

-- 4. Заполняем критерии для СПОРТИВНОЙ деятельности (category_id = 4)

-- ИСПРАВЛЕН ID (было 3, стало 4) и ОЧИЩЕН description_text

--

=====

```
INSERT INTO Rating_Criteria (category_id, section_naming, description_text, level_type_id, points) VALUES
```

-- Спортивно-оздоровительная деятельность (Оставляем как есть)

(4, 'Участие в спорте', 'Участник спортивной секции СГТУ', 1, 3),

-- Участие в спортивных соревнованиях

- (4, 'Спортивные соревнования', 'Участие в соревнованиях', 1, 1),
- (4, 'Спортивные соревнования', 'Участие в соревнованиях', 1, 2), -- Межвуз
- (4, 'Спортивные соревнования', 'Участие в соревнованиях', 2, 3),
- (4, 'Спортивные соревнования', 'Участие в соревнованиях', 3, 4),
- (4, 'Спортивные соревнования', 'Участие в соревнованиях', 4, 25),
- (4, 'Спортивные соревнования', 'Участие в соревнованиях', 5, 10),

-- Призовые места в спорте

- (4, 'Призовые места в спорте', '1 место', 1, 4),
- (4, 'Призовые места в спорте', '2 место', 1, 3),
- (4, 'Призовые места в спорте', '3 место', 1, 2),

- (4, 'Призовые места в спорте', '1 место', 2, 5),
- (4, 'Призовые места в спорте', '2 место', 2, 4),
- (4, 'Призовые места в спорте', '3 место', 2, 3),

- (4, 'Призовые места в спорте', '1 место', 3, 6),
- (4, 'Призовые места в спорте', '2 место', 3, 5),
- (4, 'Призовые места в спорте', '3 место', 3, 4),

- (4, 'Призовые места в спорте', '1 место', 4, 35),
- (4, 'Призовые места в спорте', '2 место', 4, 20),
- (4, 'Призовые места в спорте', '3 место', 4, 15),

- (4, 'Призовые места в спорте', '1 место', 5, 40),
- (4, 'Призовые места в спорте', '2 место', 5, 35),
- (4, 'Призовые места в спорте', '3 место', 5, 25);

```
--  
=====--  
-- Дополнительные INSERT-ы (Оставляем как есть, но с новыми ID)  
--  
=====
```

INSERT INTO Academic_Period (naming, start_date, end_date) VALUES
('семестр (Весна 2025)', '2025-02-01', '2025-06-30'),
('семестр (Осень 2025)', '2025-09-01', '2026-01-30');

-- 2. ДОСТИЖЕНИЯ СТУДЕНТОВ

-- Внимание: Здесь мы используем вложенные SELECT, чтобы точно попасть в нужные ID критериев,
-- так как после пересоздания таблиц ID могут сдвинуться.

INSERT INTO Student_Achievement (student_id, criteria_id, period_id, quantity, document_title, created_at) VALUES

-- === Студент 1 (Петров) ===

-- Наука: Участие в олимпиаде (ищем ID по тексту и категории)
(1,
(SELECT id FROM Rating_Criteria WHERE description_text LIKE 'Участие в предметных олимпиадах%' AND category_id=1 LIMIT 1),
2, 1, 'Сертификат участника', NOW()),

-- Культура: Участие в творческом коллективе

(1,
(SELECT id FROM Rating_Criteria WHERE description_text LIKE 'Участник творческого коллектива%' AND category_id=2 LIMIT 1),
2, 1, 'Справка из студклуба', NOW()),

-- Спорт: 1 место (ищем конкретно приз)

(1,

(SELECT id FROM Rating_Criteria WHERE description_text = '1 место' AND category_id=4 AND level_type_id=1 LIMIT 1),

2, 1, 'Грамота за победу в баскетболе', NOW()),

-- === Студент 2 (Соколова) ===

-- Общественная: Волонтер (ВУЗ)

(2,

(SELECT id FROM Rating_Criteria WHERE description_text = 'Участник волонтерского движения' AND category_id=3 AND level_type_id=1 LIMIT 1),

2, 1, 'Благодарственное письмо', NOW()),

-- Наука: Гран-при на конференции

(2,

(SELECT id FROM Rating_Criteria WHERE description_text = 'Гран-при' AND category_id=1 AND level_type_id=2 LIMIT 1),

2, 1, 'Диплом победителя', NOW()),

-- === Студент 3 (Орлов) ===

-- Спорт: Участие в соревнованиях

(3,

(SELECT id FROM Rating_Criteria WHERE description_text LIKE 'Участие в соревнованиях' AND category_id=4 AND level_type_id=2 LIMIT 1),

2, 1, 'Протокол соревнований', NOW()),

```
-- Культура: Организация праздников  
(3,  
(SELECT id FROM Rating_Criteria WHERE description_text LIKE 'Участие в  
организации факультетских%' AND category_id=2 LIMIT 1),  
2, 1, 'Благодарность деканата', NOW());
```

-- 2. Получение ID новых студентов

```
SET @gromov_id = (SELECT id FROM Student WHERE email =  
'gromov@uni.ru');  
SET @zaytseva_id = (SELECT id FROM Student WHERE email =  
'zaytseva@uni.ru');  
SET @fedorov_id = (SELECT id FROM Student WHERE email =  
'fedorov@uni.ru');  
SET @current_period_id = 2; -- ID, который используется в существующих  
записях
```

-- 3. Добавление достижений

```
-- === Студент 4 (Громов, Курс 1) ===  
-- Цель: Обойти Петрова в Спорте и Культуре  
INSERT INTO Student_Achievement (student_id, criteria_id, period_id, quantity,  
document_title, created_at) VALUES  
-- Спорт: 1 место (Университетский, 4 балла)  
(@gromov_id,  
(SELECT id FROM Rating_Criteria WHERE description_text = '1 место' AND  
category_id=4 AND level_type_id=1 LIMIT 1),  
@current_period_id, 1, 'Золотая медаль по баскетболу', NOW()),
```

-- Культура: 1 место (Университетский, 4 балла)
(@gromov_id,
(SELECT id FROM Rating_Criteria WHERE description_text = '1 место' AND category_id=2 AND level_type_id=1 LIMIT 1),
@current_period_id, 1, 'Диплом за танцевальный конкурс', NOW());

-- === Студент 5 (Зайцева, Курс 2) ===

-- Цель: Создать конкуренцию для Орлова в новых для него категориях (Наука, Общественная)

INSERT INTO Student_Achievement (student_id, criteria_id, period_id, quantity, document_title, created_at) VALUES

-- Наука: 1 место (Университетский, 3 балла)

(@zaytseva_id,
(SELECT id FROM Rating_Criteria WHERE description_text = '1 место' AND category_id=1 AND level_type_id=1 LIMIT 1),
@current_period_id, 1, 'Диплом олимпиады по IT', NOW()),

-- Общественная: Член профсоюза (Университетский, 3 балла)

(@zaytseva_id,
(SELECT id FROM Rating_Criteria WHERE description_text = 'Член профсоюзных объединений' AND category_id=3 AND level_type_id=1 LIMIT 1),
@current_period_id, 1, 'Профсоюзный билет', NOW());

-- === Студент 6 (Федоров, Курс 3) ===

-- Цель: Быть единственным в Спорт-направлении на своем курсе

INSERT INTO Student_Achievement (student_id, criteria_id, period_id, quantity, document_title, created_at) VALUES

-- Спорт: Участие в соревнованиях (Городской, 3 балла, 3 раза)
(@fedorov_id,
(SELECT id FROM Rating_Criteria WHERE description_text LIKE 'Участие в
соревнованиях' AND category_id=4 AND level_type_id=2 LIMIT 1),
@current_period_id, 3, 'Протоколы всех соревнований', NOW());

-- Добавляем колонку типа достижения (если еще не добавлена выше)
-- Это нужно для корректной работы фильтров в JS
ALTER TABLE Rating_Criteria
ADD COLUMN achievement_type VARCHAR(50)
COMMENT 'Тип: participation (участие), prize (призовое место), other (другое)';

-- Обновляем типы для JS логики
UPDATE Rating_Criteria
SET achievement_type = CASE
 WHEN description_text LIKE '%участие%' OR description_text LIKE
 '%участник%' OR description_text LIKE '%член%' THEN 'participation'
 WHEN description_text LIKE '%место%' OR description_text LIKE '%гран-
 при%' OR description_text LIKE '%грамота%' THEN 'prize'
 ELSE 'other'
END;

DELIMITER //

-- Триггер предотвращает дублирование документов по названию
CREATE TRIGGER check_duplicate_document_before_insert
BEFORE INSERT ON Student_Achievement
FOR EACH ROW
BEGIN
 -- Проверяем, есть ли у этого студента документ с таким же названием

```
-- Исключаем пустые названия, если они разрешены
IF NEW.document_title IS NOT NULL AND NEW.document_title != " THEN
    IF EXISTS (
        SELECT 1
        FROM Student_Achievement
        WHERE student_id = NEW.student_id
        AND document_title = NEW.document_title
        -- Можно добавить AND period_id = NEW.period_id если названия
        могут повторяться в разных семестрах
    ) THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'Ошибка: Документ с таким названием уже
загружен!';
    END IF;
END IF;
DELIMITER //
```

```
DELIMITER ;
```

```
DELIMITER //
```

```
-- Триггер для предотвращения вставки отрицательного числа
CREATE TRIGGER check_quantity_before_insert
BEFORE INSERT ON Student_Achievement
FOR EACH ROW
BEGIN
    IF NEW.quantity < 0 THEN
        SIGNAL SQLSTATE '45000'
```

```
SET MESSAGE_TEXT = 'Ошибка: Количество достижений (quantity) не
может быть отрицательным.';

END IF;

END//
```

-- Триггер для предотвращения изменения на отрицательное число

```
CREATE TRIGGER check_quantity_before_update
```

```
BEFORE UPDATE ON Student_Achievement
```

```
FOR EACH ROW
```

```
BEGIN
```

```
IF NEW.quantity < 0 THEN
```

```
    SIGNAL SQLSTATE '45000'
```

```
    SET MESSAGE_TEXT = 'Ошибка: Количество достижений (quantity) не
может быть отрицательным.';

END IF;
```

```
END//
```

```
DELIMITER ;
```

-- Процедура для вывода рейтинга по направлениям достижений

```
DELIMITER //
```

```
CREATE PROCEDURE GetCourseRatingAnalysis(
```

```
    IN studentId INT,
```

```
    IN periodId INT
```

```
)
```

```
BEGIN
```

```
    DECLARE my_course INT;
```

-- 1. Узнаем курс студента

```
SELECT sg.course INTO my_course
FROM Student s
JOIN Student_group sg ON s.student_group_id = sg.id
WHERE s.id = studentId;
```

-- 2. Выводим статистику по каждой категории

```
SELECT
    ac.naming AS category_name,
    ac.cod AS category_cod,
```

-- Мои баллы в этой категории

```
(SELECT COALESCE(SUM(rc.points * sa.quantity), 0)
```

```
FROM Student_Achievement sa
JOIN Rating_Criteria rc ON sa.criteria_id = rc.id
WHERE sa.student_id = studentId
AND sa.period_id = periodId
AND rc.category_id = ac.id
```

```
) as my_points,
```

-- Мое место на курсе (считаем, у скольких людей БОЛЬШЕ, чем у меня)

```
(SELECT COUNT(*) + 1
```

```
FROM (
```

```
    SELECT s_sub.id, SUM(rc_sub.points * sa_sub.quantity) as total_p
```

```
    FROM Student s_sub
```

```
    JOIN Student_group sg_sub ON s_sub.student_group_id = sg_sub.id
```

```
    JOIN Student_Achievement sa_sub ON s_sub.id = sa_sub.student_id
```

```
    JOIN Rating_Criteria rc_sub ON sa_sub.criteria_id = rc_sub.id
```

```
    WHERE sg_sub.course = my_course -- Тот же курс
```

```
        AND sa_sub.period_id = periodId
        AND rc_sub.category_id = ac.id -- Та же категория
        GROUP BY s_sub.id
    ) as competitors
    WHERE competitors.total_p >
        (SELECT COALESCE(SUM(rc2.points * sa2.quantity), 0)
        FROM Student_Achievement sa2
        JOIN Rating_Criteria rc2 ON sa2.criteria_id = rc2.id
        WHERE sa2.student_id = studentId
        AND sa2.period_id = periodId
        AND rc2.category_id = ac.id)
)
```

) as my_rank,

```
-- Всего участников в этой категории на курсе (конкуренция)
(SELECT COUNT(DISTINCT sa_sub.student_id)
FROM Student_Achievement sa_sub
JOIN Student s_sub ON sa_sub.student_id = s_sub.id
JOIN Student_group sg_sub ON s_sub.student_group_id = sg_sub.id
JOIN Rating_Criteria rc_sub ON sa_sub.criteria_id = rc_sub.id
WHERE sg_sub.course = my_course
AND sa_sub.period_id = periodId
AND rc_sub.category_id = ac.id
) as total_participants
```

FROM Activity_Category ac;

END//

DELIMITER ;

Style.css

```
:where([class^="ri-"])::before {
```

```
content: "\f3c2";
```

```
}
```

```
/* Основные стили */
```

```
body {
```

```
    font-family: 'Roboto', sans-serif;
```

```
}
```

```
#profile {
```

```
    display: flex;
```

```
    margin: 20px;
```

```
}
```

```
aside {
```

```
    width: 223px;
```

```
    height: 896px;
```

```
}
```

```
main {
```

```
    flex: 1;
```

```
    margin-left: 20px;
```

```
}
```

```
/* Шапка */
```

```
header {
```

```
    height: 96px;
```

```
    background-color: #6D28D9;
```

```
    color: white;
```

```
}
```

```
/* Отступы и размеры */  
main {  
    margin-top: 20px;  
    padding: 20px;  
}  
  
.grid {  
    display: grid;  
    gap: 20px;  
}  
  
@media (max-width: 1024px) {  
    .grid {  
        grid-template-columns: 1fr;  
    }  
}  
  
.bg-primary {  
    background-color: #3B82F6;  
}  
  
.bg-purple-500 {  
    background-color: #6D28D9;  
}  
  
.bg-white {  
    background-color: #ffffff;  
}  
  
.rounded-lg {
```

```
border-radius: 8px;  
}  
  
.shadow-sm {  
  box-shadow: 0 1px 2px rgba(0, 0, 0, 0.05);  
}  
  
.border {  
  border: 1px solid #e5e7eb;  
}  
  
.text-gray-600 {  
  color: #4B5563;  
}  
  
.text-gray-900 {  
  color: #111827;  
}  
  
.text-primary {  
  color: #3B82F6;  
}  
  
.hover\:bg-gray-100:hover {  
  background-color: #f3f4f6;  
}  
  
.transition-colors {  
  transition: background-color 0.2s;  
}
```

Код – base.html

```
<!DOCTYPE html>
<html lang="ru">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>{ % block title % }Личный кабинет{ % endblock % }</title>

    <!-- Tailwind CSS -->
    <script src="https://cdn.tailwindcss.com"></script>
    <script>
        tailwind.config = {
            theme: {
                extend: {
                    colors: {
                        primary: '#3B82F6',
                        secondary: '#10B981',
                    },
                    fontSize: {
                        'heading': '34px',
                        'subheading': '21px',
                        'base': '14px',
                        'description': '10px'
                    }
                }
            }
        }
    </script>
```

```
<!-- Google Fonts -->
<link href="https://fonts.googleapis.com/css2?family=Pacifico&display=swap"
rel="stylesheet">

<!-- Remix Icons -->
<link
href="https://cdnjs.cloudflare.com/ajax/libs/remixicon/4.6.0/remixicon.min.css"
rel="stylesheet">

<!-- Custom Styles -->
<link rel="stylesheet" href="{{ url_for('static', filename='css/styles.css') }}">

{ % block head_extra % }{ % endblock % }

</head>

<body class="bg-gray-50 text-base">
<div class="min-h-screen flex flex-col">
<!-- Header - 96px height -->
<header class="h-24 bg-white shadow-sm border-b border-gray-200 w-full">
<div class="flex items-center h-full">
<!-- Logo - 223px width, 96px height -->
<div class="w-[223px] h-24 flex items-center justify-center border-r border-gray-200">

</div>

<!-- Search and user menu -->
```

```
<div class="flex-1 flex items-center justify-between px-6">
  <div class="flex-1 max-w-md">
    <div class="relative">
      <div class="absolute inset-y-0 left-0 pl-3 flex items-center">
        <div class="w-5 h-5 flex items-center justify-center text-gray-400">
          <i class="ri-search-line"></i>
        </div>
      </div>
      <input
        class="w-full pl-10 pr-4 py-2 border border-gray-300 rounded-lg focus:ring-2 focus:ring-primary focus:border-transparent text-base"
        placeholder="Поиск..." type="text" />
    </div>
  </div>

  <div class="flex items-center gap-4">
    <a href="{{ url_for('notifications') }}" class="relative p-2 rounded-lg hover:bg-gray-100">
      <div class="w-5 h-5 flex items-center justify-center">
        <i class="ri-notification-line"></i>
      </div>
      <span
        class="absolute -top-1 -right-1 bg-red-500 text-white text-decoration w-5 h-5 rounded-full flex items-center justify-center">3</span>
    </a>
    <a href="{{ url_for('messages') }}" class="relative p-2 rounded-lg hover:bg-gray-100">
      <div class="w-5 h-5 flex items-center justify-center">
        <i class="ri-message-line"></i>
      </div>
    </a>
  </div>

```

```
</div>

<span
    class="absolute -top-1 -right-1 bg-primary text-white text-
description w-5 h-5 rounded-full flex items-center justify-center">7</span>

</a>

<!-- Profile Dropdown -->
{ % if session.get('student_name') % }

<div class="relative" id="profile-dropdown">
    <button id="profile-btn" class="flex items-center gap-3 p-2
rounded-lg hover:bg-gray-100">
        <!-- В шапке сайта -->
        
        <span class="text-base font-medium text-gray-700">{{ session.get('student_name') }}</span>
    <div class="w-4 h-4 flex items-center justify-center">
        <i class="ri-arrow-down-s-line"></i>
    </div>
</button>

<!-- Выпадающее меню -->
```

```
<div id="dropdown-menu" class="absolute right-0 mt-2 w-48 bg-white rounded-lg shadow-lg border border-gray-200 hidden z-50">
    <a href="{{ url_for('profile') }}" class="block px-4 py-2 hover:bg-gray-100 text-base">Профиль</a>
    <a href="{{ url_for('logout') }}" class="block px-4 py-2 hover:bg-gray-100 text-red-600 text-base">Выйти</a>
</div>
</div>
{ % endif % }
</div>
</div>
</div>
</div>
</header>
```

```
<div class="flex flex-1">
    <!-- Sidebar - 223px width, 896px height (between header and footer) -->
    <aside class="w-[223px] bg-white shadow-lg flex flex-col border-r border-gray-200">
        <nav class="flex-1 p-4 space-y-2">
            <a href="{{ url_for('profile') }}"
                class="nav-item flex items-center gap-3 px-4 py-3 rounded-lg
                hover:bg-gray-100 transition-colors { % if request.endpoint == 'profile' % }bg-gray-100{ % endif % }">
                <i class="ri-user-line w-5 h-5"></i>
                <span class="text-base">Профиль</span>
            </a>
            <a href="{{ url_for('notifications') }}"
                class="nav-item flex items-center gap-3 px-4 py-3 rounded-lg
                hover:bg-gray-100 transition-colors { % if request.endpoint == 'notifications' % }bg-gray-100{ % endif % }">
```

```
<i class="ri-notification-line w-5 h-5"></i>
<span class="text-base">Уведомления</span>
</a>
<a href="{{ url_for('messages') }}"
    class="nav-item flex items-center gap-3 px-4 py-3 rounded-lg
    hover:bg-gray-100 transition-colors { % if request.endpoint == 'messages' % }bg-
    gray-100{ % endif % }">
    <i class="ri-message-line w-5 h-5"></i>
    <span class="text-base">Сообщения</span>
</a>
<a href="{{ url_for('grades') }}"
    class="nav-item flex items-center gap-3 px-4 py-3 rounded-lg
    hover:bg-gray-100 transition-colors { % if request.endpoint == 'grades' % }bg-gray-
    100{ % endif % }">
    <i class="ri-bar-chart-line w-5 h-5"></i>
    <span class="text-base">Успеваемость</span>
</a>
<a href="{{ url_for('materials') }}"
    class="nav-item flex items-center gap-3 px-4 py-3 rounded-lg
    hover:bg-gray-100 transition-colors { % if request.endpoint == 'materials' % }bg-
    gray-100{ % endif % }">
    <i class="ri-book-line w-5 h-5"></i>
    <span class="text-base">Материалы</span>
</a>
<a href="{{ url_for('portfolio') }}"
    class="nav-item flex items-center gap-3 px-4 py-3 rounded-lg
    hover:bg-gray-100 transition-colors { % if request.endpoint == 'portfolio' % }bg-
    gray-100{ % endif % }">
    <i class="ri-folder-line w-5 h-5"></i>
    <span class="text-base">Портфолио</span>
```

```
</a>

<a href="{ { url_for('decanat') } }"
    class="nav-item flex items-center gap-3 px-4 py-3 rounded-lg
    hover:bg-gray-100 transition-colors { % if request.endpoint == 'decanat' % }bg-gray-
    100{ % endif % }">
    <i class="ri-government-line w-5 h-5"></i>
    <span class="text-base">Онлайн-деканат</span>
</a>

<a href="{ { url_for('stipendii') } }"
    class="nav-item flex items-center gap-3 px-4 py-3 rounded-lg
    hover:bg-gray-100 transition-colors { % if request.endpoint == 'stipendii' % }bg-
    gray-100{ % endif % }">
    <i class="ri-award-line w-5 h-5"></i>
    <span class="text-base">Стипендии</span>
</a>

</nav>

</aside>
```

```
<!-- Main Content Area -->
<main class="flex-1 flex flex-col min-h-0">
    <!-- Page Content - This will be the content block -->
    <div class="flex-1 p-[20px] overflow-auto">
        { % block content % }{ % endblock % }
    </div>
</main>
</div>
```

```
<!-- Footer - 32px height -->
<footer class="h-8 bg-white border-t border-gray-200 w-full flex items-center
justify-center">
```

```
<div class="text-description text-gray-500">
    © 2025 Саратовский государственный технический университет
    имени Гагарина Ю.А.

</div>
</footer>
</div>

<div class="fixed bottom-5 right-5 z-50 flex flex-col gap-3 w-full max-w-sm
pointer-events-none" id="toast-container">
    {% with messages = get_flashed_messages(with_categories=true) %}
        {% if messages %}
            {% for category, message in messages %}
                <div class="toast-message pointer-events-auto transform transition-all
duration-300 translate-x-0 opacity-100 flex items-center p-4 bg-white rounded-lg
shadow-lg border-1-4 {% if category == 'success' %}border-green-500{% else
%}border-red-500{% endif %}" role="alert">
                    <div class="flex-shrink-0 mr-3">
                        {% if category == 'success' %}
                            <div class="w-8 h-8 bg-green-100 rounded-full flex items-
center justify-center">
                                <i class="ri-check-line text-green-600 text-lg"></i>
                            </div>
                        {% else %}
                            <div class="w-8 h-8 bg-red-100 rounded-full flex items-center
justify-center">
                                <i class="ri-error-warning-line text-red-600 text-lg"></i>
                            </div>
                        {% endif %}
                    </div>
                </div>
            {% endfor %}
        {% endif %}
    {% endwith %}
</div>
```

```
<div class="flex-1 mr-2">
    <p class="text-sm font-medium text-gray-900">
        { % if category == 'success' % } Успешно{ % else % }Ошибка{ %
    endif % }

    </p>
    <p class="text-sm text-gray-600">{ { message } }</p>
</div>

<button onclick="this.parentElement.remove()" class="text-gray-400 hover:text-gray-900 transition-colors">
    <i class="ri-close-line text-xl"></i>
</button>
</div>

{ % endfor %

{ % endif %

{ % endwith %

</div>

<script>
    // Автоматическое скрытие уведомлений через 5 секунд
    document.addEventListener('DOMContentLoaded', () => {
        const toasts = document.querySelectorAll('.toast-message');
        toasts.forEach((toast, index) => {
            // Добавляем небольшую задержку появления для эффекта каскада
            setTimeout(() => {
                toast.classList.remove('translate-y-10', 'opacity-0');
            }, index * 100);

            // Таймер исчезновения
            setTimeout(() => {

```

```
toast.classList.add('translate-x-full', 'opacity-0');

setTimeout(() => {
    toast.remove();
}, 300); // Ждем окончания анимации перед удалением из DOM
}, 5000 + (index * 500));

});

});

</script>

<script>
// Управление выпадающим меню профиля

document.addEventListener('DOMContentLoaded', function() {
    const profileBtn = document.getElementById('profile-btn');
    const dropdownMenu = document.getElementById('dropdown-menu');

    if (profileBtn && dropdownMenu) {
        profileBtn.addEventListener('click', function(e) {
            e.stopPropagation();
            dropdownMenu.classList.toggle('hidden');
        });
    }

    // Закрытие меню при клике вне его
    document.addEventListener('click', function(e) {
        if (!dropdownMenu.contains(e.target) &&
!profileBtn.contains(e.target)) {
            dropdownMenu.classList.add('hidden');
        }
    });
}

// Закрытие меню при нажатии ESC

```

```

document.addEventListener('keydown', function(e) {
    if (e.key === 'Escape') {
        dropdownMenu.classList.add('hidden');
    }
});

});

</script>

{ % block scripts % }{ % endblock % }

</body>

</html>

```

Код – profile.html

```

{ % extends "base.html" % }

{ % block title % }Профиль{ % endblock % }

{ % block content % }

<!-- Используем p-5 для 20px отступов как в notifications.html -->
<div class="p-5">
    <div class="flex flex-col lg:flex-row gap-3">
        <!-- Основной контент - левая часть -->
        <div class="bg-white rounded-lg shadow-sm border border-gray-200 p-4 flex-1">
            <div class="mb-3">
                <h1 class="text-heading font-bold text-gray-900 leading-tight">
                    Профиль пользователя
                </h1>

```

```
<p class="text-base text-gray-600">  
    Управление личными данными и настройками аккаунта  
</p>  
</div>  
  
<!-- Форма обновления профиля -->  
<form action="{{ url_for('update_profile') }}" method="POST">  
    <!-- Личные данные -->  
    <div class="mb-3">  
        <h2 class="text-subheading font-semibold text-gray-900 mb-2">  
            Личные данные  
        </h2>  
        <div class="grid grid-cols-1 md:grid-cols-2 gap-2">  
            <div>  
                <label class="block text-base font-medium text-gray-700 mb-1">  
                    Фамилия  
                </label>  
                <input class="w-full px-3 py-1 border border-gray-300 rounded-lg bg-gray-50 text-base"  
                      type="text" value="{{ student.surname }}" readonly />  
            </div>  
            <div>  
                <label class="block text-base font-medium text-gray-700 mb-1">  
                    Имя  
                </label>  
                <input class="w-full px-3 py-1 border border-gray-300 rounded-lg bg-gray-50 text-base"  
                      type="text" value="{{ student.first_name }}" readonly />  
            </div>  
        </div>  
</div>
```

```
<label class="block text-base font-medium text-gray-700 mb-1">
    Отчество
</label>
<input class="w-full px-3 py-1 border border-gray-300 rounded-lg bg-gray-50 text-base"
    type="text" value="{{ student.middle_name or '' }}" readonly
/>
</div>
<div>
    <label class="block text-base font-medium text-gray-700 mb-1">
        Дата рождения
    </label>
    <input class="w-full px-3 py-1 border border-gray-300 rounded-lg bg-gray-50 text-base"
        type="text" value="{{ student.birth_date_str }}" readonly />
</div>
<div>
    <label class="block text-base font-medium text-gray-700 mb-1">
        Группа
    </label>
    <input class="w-full px-3 py-1 border border-gray-300 rounded-lg bg-gray-50 text-base"
        readonly type="text" value="{{ student.group_name }}" />
</div>
<div>
    <label class="block text-base font-medium text-gray-700 mb-1">
        Курс
    </label>
    <input class="w-full px-3 py-1 border border-gray-300 rounded-lg bg-gray-50 text-base"
```

```
        readonly type="text" value="{{ student.course }}" />
    </div>
</div>
</div>

<!-- Контактная информация -->
<div class="mb-3">
    <h2 class="text-subheading font-semibold text-gray-900 mb-2">
        Контактная информация
    </h2>
    <div class="grid grid-cols-1 md:grid-cols-2 gap-2">
        <div>
            <label class="block text-base font-medium text-gray-700 mb-1">
                Email
            </label>
            <input name="email"
                class="w-full px-3 py-1 border border-gray-300 rounded-lg
focus:ring-2 focus:ring-primary focus:border-transparent text-base"
                type="email" value="{{ student.email }}""
                pattern="[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}"
                title="Введите корректный email"
                placeholder="example@domain.com" required />
        </div>
        <div>
            <label class="block text-base font-medium text-gray-700 mb-1">
                Телефон
            </label>
            <input name="phone"
                class="w-full px-3 py-1 border border-gray-300 rounded-lg
focus:ring-2 focus:ring-primary focus:border-transparent text-base"
                type="text" value="{{ student.phone }}" />
        </div>
    </div>
</div>
```

```
        type="tel" value="{{ student.phone }}" pattern="\+7\d{10}"  
        title="Введите телефон в формате +7XXXXXXXXXXX"  
placeholder="+7XXXXXXXXXXX" required />  
  
    </div>  
  
    <div class="md:col-span-2">  
        <label class="block text-base font-medium text-gray-700 mb-1">  
            Адрес  
        </label>  
        <input name="address"  
               class="w-full px-3 py-1 border border-gray-300 rounded-lg  
               focus:ring-2 focus:ring-primary focus:border-transparent text-base"  
               type="text" value="{{ student.address }}" title="Введите  
полный адрес"  
               placeholder="ул. Название улицы, д. Номер" minlength="5"  
required />  
  
        </div>  
  
    </div>  
  
    </div>  
  
    <!-- Кнопки сохранения -->  
    <div class="mt-4 flex flex-col sm:flex-row justify-end gap-2">  
        <button type="button" onclick="location.reload()"  
               class="px-3 py-1 border border-gray-300 rounded-lg hover:bg-gray-  
50 transition-colors text-base order-2 sm:order-1">  
            Отменить  
        </button>  
        <button type="submit"  
               class="bg-primary text-white px-3 py-1 rounded-lg hover:bg-blue-  
600 transition-colors text-base order-1 sm:order-2">  
            Сохранить изменения  
        </button>  
    </div>
```

```
</button>

</div>

</form>

<!-- Смена пароля (отдельная форма) --&gt;
&lt;div class="mt-4 pt-4 border-t border-gray-200"&gt;
    &lt;form action="{{ url_for('update_password') }}" method="POST"&gt;
        &lt;h2 class="text-subheading font-semibold text-gray-900 mb-2"&gt;
            Смена пароля
        &lt;/h2&gt;
        &lt;div class="grid grid-cols-1 md:grid-cols-2 gap-2"&gt;
            &lt;div&gt;
                &lt;label class="block text-base font-medium text-gray-700 mb-1"&gt;
                    Текущий пароль
                &lt;/label&gt;
                &lt;input name="current_password"
                    class="w-full px-3 py-1 border border-gray-300 rounded-lg
focus:ring-2 focus:ring-primary focus:border-transparent text-base"
                    type="password" required /&gt;
            &lt;/div&gt;
            &lt;div&gt;
                &lt;label class="block text-base font-medium text-gray-700 mb-1"&gt;
                    Новый пароль
                &lt;/label&gt;
                &lt;input name="new_password"
                    class="w-full px-3 py-1 border border-gray-300 rounded-lg
focus:ring-2 focus:ring-primary focus:border-transparent text-base"
                    type="password" required /&gt;
            &lt;/div&gt;
            &lt;div class="md:col-span-2"&gt;</pre>
```

```
<label class="block text-base font-medium text-gray-700 mb-1">
    Подтвердите пароль
</label>
<input name="confirm_password"
       class="w-full px-3 py-1 border border-gray-300 rounded-lg
       focus:ring-2 focus:ring-primary focus:border-transparent text-base"
       type="password" required />
</div>
</div>
<div class="mt-2 flex flex-col sm:flex-row justify-end gap-2">
    <button type="button" onclick="location.reload()"
           class="px-3 py-1 border border-gray-300 rounded-lg hover:bg-
           gray-50 transition-colors text-base">
        Отменить
    </button>
    <button type="submit"
           class="bg-primary text-white px-3 py-1 rounded-lg hover:bg-blue-
           600 transition-colors text-base">
        Сменить пароль
    </button>
</div>
</div>
</div>

<!-- Боковая панель - правая часть -->
<div class="flex flex-col gap-3 lg:w-64">
    <!-- Карточка с фото профиля -->
    <div class="bg-white rounded-lg shadow-sm border border-gray-200 p-4">
        <h2 class="text-subheading font-semibold text-gray-900 mb-2">
```

Фото профиля

</h2>

<div class="text-center">

<!-- В карточке с фото профиля -->

<button type="button" onclick="showPhotoUpload()" class="bg-primary text-white px-3 py-1 rounded-lg hover:bg-blue-600 transition-colors text-base">

Изменить фото

</button>

</div>

</div>

<!-- Карточка с информацией об обучении -->

<div class="h-[595px] bg-white rounded-lg shadow-sm border border-gray-200 p-4">

<h2 class="text-subheading font-semibold text-gray-900 mb-2">

Информация об обучении

</h2>

<div class="space-y-1.5">

<div>

Номер зачетной книжки:

```
</span>
<span class="font-medium text-gray-900 text-base">
  {{ student.gradebook_number }}
</span>
</div>
<div>
  <span class="block text-base text-gray-600">
    Специальность:
  </span>
  <span class="font-medium text-gray-900 text-base">
    {{ student.specialty_name }} ({{ student.cod_specialty }})
  </span>
</div>
<div>
  <span class="block text-base text-gray-600">
    Форма обучения:
  </span>
  <span class="font-medium text-gray-900 text-base">
    {{ student.form_study_name }}
  </span>
</div>
<div>
  <span class="block text-base text-gray-600">
    Куратор:
  </span>
  <span class="font-medium text-gray-900 text-base">
    {{ student.tutor_surname }} {{ student.tutor_first_name }} {{ student.tutor_middle_name }}
  </span>
</div>
```

```
<div>
    <span class="block text-base text-gray-600">
        Год поступления:
    </span>
    <span class="font-medium text-gray-900 text-base">
        {{ student.created_year if student.created_year else '2021' }}
    </span>
</div>
<div>
    <span class="block text-base text-gray-600">
        Год окончания:
    </span>
    <span class="font-medium text-gray-900 text-base">
        {{ student.graduation_year }}
    </span>
</div>
</div>
</div>
</div>
```

```
<!-- Модальное окно для загрузки фото -->
<div id="photoUploadModal" class="hidden fixed inset-0 bg-gray-900 bg-opacity-50 flex items-center justify-center z-50">
    <div class="bg-white rounded-lg shadow-xl w-full max-w-md mx-4">
        <div class="p-4 border-b border-gray-200">
            <div class="flex items-center justify-between">
                <h3 class="text-subheading font-semibold text-gray-900">
                    Изменить фото профиля
                </h3>
            </div>
        </div>
    </div>
</div>
```

```
</h3>

<button onclick="hidePhotoUpload()" class="text-gray-400 hover:text-gray-500">
    <div class="w-5 h-5 flex items-center justify-center">
        ×
    </div>
</button>
</div>
</div>

<form action="{{ url_for('upload_photo') }}" method="POST" class="p-4">
    <div class="mb-3">
        <label class="block text-base font-medium text-gray-700 mb-1">
            URL foto
        </label>
        <input type="url" name="photo_url"
            class="w-full px-3 py-1 border border-gray-300 rounded-lg focus:ring-2 focus:ring-primary focus:border-transparent text-base"
            placeholder="https://example.com/photo.jpg" required />
    </div>
    <div class="flex justify-end gap-2">
        <button type="button" onclick="hidePhotoUpload()"
            class="px-3 py-1 border border-gray-300 rounded-lg hover:bg-gray-50 transition-colors text-base">
            Отмена
        </button>
        <button type="submit"
            class="bg-primary text-white px-3 py-1 rounded-lg hover:bg-blue-600 transition-colors text-base">
            Сохранить
        </button>
    </div>
</form>
```

```
</div>
</form>
</div>
</div>

<script>
    function showPhotoUpload() {
        document.getElementById('photoUploadModal').classList.remove('hidden');
        document.body.style.overflow = 'hidden';
    }

    function hidePhotoUpload() {
        document.getElementById('photoUploadModal').classList.add('hidden');
        document.body.style.overflow = "";
    }

    // Закрытие по клику вне модального окна
    document.getElementById('photoUploadModal').addEventListener('click',
function (e) {
    if (e.target === this) {
        hidePhotoUpload();
    }
});
</script>

<script>
// Динамическая валидация формы
document.addEventListener('DOMContentLoaded', function() {
    const form = document.querySelector('form[action="{!! url_for('update_profile') !!}""]');
})
```

```
if (form) {  
    form.addEventListener('submit', function(event) {  
        let isValid = true;  
  
        const email = form.querySelector('input[name="email"]');  
        const phone = form.querySelector('input[name="phone"]');  
        const address = form.querySelector('input[name="address"]');  
  
        // Проверка email  
        const emailPattern = /^[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}$/;  
        if (!emailPattern.test(email.value)) {  
            showError(email, 'Email должен быть в формате  
example@domain.com');  
            isValid = false;  
        } else {  
            clearError(email);  
        }  
  
        // Проверка телефона  
        const phonePattern = /\^+7\d{10}\$/;  
        if (!phonePattern.test(phone.value)) {  
            showError(phone, 'Телефон должен быть в формате  
+7XXXXXXXXXX (10 цифр после +7)');  
            isValid = false;  
        } else {  
            clearError(phone);  
        }  
  
        // Проверка адреса
```

```
if (address.value.length < 5) {
    showError(address, 'Адрес должен содержать не менее 5 символов');
    isValid = false;
} else if (!/(ул\.|улица|проспект|пр\.).*(дом|д\.|\\d)/i.test(address.value)) {
    showError(address, 'Адрес должен содержать улицу и дом (например,
"ул. Ленина, д. 10")');
    isValid = false;
} else {
    clearError(address);
}

if (!isValid) {
    event.preventDefault();
}
});

// Валидация на лету
['email', 'phone', 'address'].forEach(fieldName => {
    const field = form.querySelector(`input[name="${fieldName}"]`);
    if (field) {
        field.addEventListener('blur', function() {
            validateField(this);
        });
        field.addEventListener('input', function() {
            clearError(this);
        });
    }
});

function validateField(field) {
```

```
if (field.name === 'email') {
    const pattern = /^[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}$/;
    if (!pattern.test(field.value)) {
        showError(field,      'Email      должен      быть      в      формате
example@domain.com');
    } else {
        clearError(field);
    }
} else if (field.name === 'phone') {
    const pattern = /^+7\d{10}$/;
    if (!pattern.test(field.value)) {
        showError(field,      'Телефон      должен      быть      в      формате
+7XXXXXXXXXXX (10 цифр после +7)');
    } else {
        clearError(field);
    }
} else if (field.name === 'address') {
    if (field.value.length < 5) {
        showError(field, 'Адрес должен содержать не менее 5 символов');
    } else if (!/(ул\.|улица|проспект|пр\.).*(дом|д\.|d)/i.test(field.value)) {
        showError(field, 'Адрес должен содержать улицу и дом');
    } else {
        clearError(field);
    }
}
```

```
function showError(field, message) {
    clearError(field);
    field.classList.add('border-red-500');
```

```
let errorDiv = document.createElement('div');
errorDiv.className = 'text-red-500 text-description mt-1';
errorDiv.textContent = message;
errorDiv.id = field.name + '-error';

field.parentNode.appendChild(errorDiv);

}

function clearError(field) {
    field.classList.remove('border-red-500');
    const errorDiv = document.getElementById(field.name + '-error');
    if (errorDiv) {
        errorDiv.remove();
    }
}
});

</script>φ
{ % endblock % }
```

Код – portfolio.html

```
{ % extends "base.html" % }

{ % block title % }Портфолио и рейтинг{ % endblock % }

{ % block styles % }
<style>
.achievement-card {
    transition: transform 0.2s;
```

```
        }
```



```
.achievement-card:hover {  
    transform: translateY(-2px);  
}  
  
/* Стили для пошагового индикатора */  


```
.step-indicator {
 width: 32px;
 height: 32px;
 border-radius: 50%;
 display: flex;
 align-items: center;
 justify-content: center;
 font-weight: 600;
 font-size: 14px;
}

.step-active {
 background-color: #3B82F6;
 color: white;
}

.step-inactive {
 background-color: #F3F4F6;
 color: #9CA3AF;
}
</style>
{% endblock %}
```


```

```
{% block content %}

<div class="p-5">
  <div class="mb-[20px]">
    <div class="flex items-center justify-between">
      <div>
        <h1 class="text-heading font-bold text-gray-900">
          Портфолио
        </h1>
        <p class="text-base text-gray-500 mt-1">
          Управление достижениями и рейтингом
        </p>
      </div>
      <div class="text-center bg-white border border-gray-200 rounded-lg px-6 py-3 shadow-sm">
        <div class="text-subheading font-bold text-primary">{{ data.total_points }}</div>
        <div class="text-base text-gray-500">Общий балл</div>
      </div>
    </div>
  </div>

<div class="grid grid-cols-1 md:grid-cols-2 lg:grid-cols-4 gap-4 mb-[10px]">
  {% for category in data.categories %}
    <div
      class="bg-white rounded-xl shadow-sm border border-gray-100 p-5
      hover:border-gray-300 transition-colors group">
      <div class="flex justify-between items-start mb-1">
        <div class="w-10 h-10 rounded-lg flex items-center justify-center
        {% if category.cod == 'science' %}bg-blue-50 text-blue-600{% endif
        %}>
```

```
{% if category.cod == 'cultural' %}bg-purple-50 text-purple-600{%
endif %}

{%
if category.cod == 'sport' %}bg-green-50 text-green-600{%
endif %}

{%
if category.cod == 'social' %}bg-yellow-50 text-yellow-600{%
endif %}">

<i class="text-subheading

{%
if category.cod == 'science' %}ri-microscope-line{%
endif %

{%
if category.cod == 'cultural' %}ri-palette-line{%
endif %

{%
if category.cod == 'sport' %}ri-run-line{%
endif %

{%
if category.cod == 'social' %}ri-team-line{%
endif %}">

</i>
</div>
<span class="bg-gray-100 text-gray-600 text-description font-bold px-2
py-1 rounded-full">
    {{ category.total_count }} шт.
</span>
</div>

<h3 class="font-semibold text-gray-900 mb-1">{{ category.name }}</h3>
<div class="text-text-subheading font-bold

{%
if category.cod == 'science' %}text-blue-600{%
endif %

{%
if category.cod == 'cultural' %}text-purple-600{%
endif %

{%
if category.cod == 'sport' %}text-green-600{%
endif %

{%
if category.cod == 'social' %}text-yellow-600{%
endif %}">
    {{ category.points }} <span class="text-base font-medium text-gray-
400">баллов</span>
</div>
```

```
<button onclick="showAddForm('{{ category.cod }}', '{{ category.id }}')"
    class="w-full mt-4 py-1 text-base font-medium text-gray-600 border
    border-gray-200 rounded-lg hover:bg-gray-50 hover:text-gray-900 transition-
    colors">
    <i class="ri-add-line mr-1"></i> Добавить
</button>
</div>
{ % endfor %}
</div>

<div class="mb-[20px]">
    <h2 class="text-subheading font-bold text-gray-900 mb-2">Структура
    рейтинга</h2>
    <div class="grid grid-cols-1 md:grid-cols-2 gap-4">
        { % for category in data.categories %}
            { % set percent = (category.points / data.total_points * 100)|round|int if
            data.total_points > 0 else 0 % }

            <div class="bg-white border border-gray-100 rounded-xl p-4 shadow-sm
            flex flex-col justify-center h-full">
                <div class="flex justify-between items-end mb-1">
                    <div class="flex items-center gap-2">
                        <span class="w-2 h-2 rounded-full
                            { % if category.cod == 'science' % }bg-blue-500{ % endif %
                            { % if category.cod == 'cultural' % }bg-purple-500{ % endif %
                            { % if category.cod == 'sport' % }bg-green-500{ % endif %
                            { % if category.cod == 'social' % }bg-yellow-500{ % endif %
                            % }"></span>
                        <span class="font-medium text-gray-700">{{ category.name
                            } }</span>
                    </div>
                </div>
            </div>
        { % endfor %}
    </div>
</div>
```

```
</div>

<span class="text-description text-gray-400 font-medium">{ { percent
} }% от общего</span>

</div>

<div class="flex items-center gap-4">
    <div class="flex-1 h-2 bg-gray-100 rounded-full overflow-hidden">
        <div class="h-full rounded-full transition-all duration-700
            { % if category.cod == 'science' % }bg-blue-500{ % endif % }
            { % if category.cod == 'cultural' % }bg-purple-500{ % endif % }
            { % if category.cod == 'sport' % }bg-green-500{ % endif % }
            { % if category.cod == 'social' % }bg-yellow-500{ % endif % }"
            style="width: { { percent } }%">
            </div>
        </div>
    <div class="text-right whitespace nowrap min-w-[60px]">
        <span class="font-bold text-gray-900">{ { category.points
} }</span>
        <span class="text-description text-gray-400 font-
normal">балл.</span>
        </div>
    </div>
</div>
{ % endfor % }

</div>
</div>

{ % if data.analysis % }

<div class="bg-white rounded-lg shadow-sm border border-gray-200 p-6 mb-
[20px]">
```

```
<h2 class="text-xl font-semibold text-gray-900 mb-4">
  <i class="ri-line-chart-line mr-2"></i>Анализ конкуренции на курсе
</h2>
<p class="text-base text-gray-500 mb-2">
  Таблица показывает ваше место среди всех студентов {{ student.course
}} курса.

  Используйте это, чтобы выбрать направление с наименьшей
конкуренцией для подачи на стипендию.

</p>
```

```
<div class="overflow-x-auto">
  <table class="w-full min-w-full">
    <thead>
      <tr class="bg-gray-50 border-b border-gray-200">
        <th class="text-left py-3 px-4 font-medium text-gray-600">Направление</th>
        <th class="text-center py-3 px-4 font-medium text-gray-600">Ваши
          баллы</th>
        <th class="text-center py-3 px-4 font-medium text-gray-600">Ваше
          место</th>
        <th class="text-center py-3 px-4 font-medium text-gray-600">Конкуренция</th>
        <th class="text-left py-3 px-4 font-medium text-gray-600">Статус</th>
      </tr>
    </thead>
    <tbody>
      { % for item in data.analysis % }
      <tr class="border-b border-gray-100 hover:bg-gray-50">
        <td class="py-1.5 px-4 font-medium text-gray-900">
```

```
    {{ item.category_name }}
```

```
</td>
```

```
<td class="py-1.5 px-4 text-center">
```

```
    <span class="font-bold text-primary">{{ item.my_points }}
```

```
</span>
```

```
</td>
```

```
<td class="py-1.5 px-4 text-center">
```

```
    <span
```

```
        class="inline-flex items-center justify-center w-8 h-8 rounded-full
```

```
            {% if item.my_rank == 1 %}bg-yellow-100 text-yellow-700
```

```
font-bold{% else %}bg-gray-100 text-gray-600{% endif %}>
```

```
        {{ item.my_rank }}
```

```
</span>
```

```
</td>
```

```
<td class="py-1.5 px-4 text-center text-gray-500">
```

```
    из {{ item.total_participants }} студентов
```

```
</td>
```

```
<td class="py-1.5 px-4">
```

```
    {% if item.my_rank <= 3 and item.my_points > 0 %}
```

```
        <span class="text-green-600 text-base font-medium flex items-
```

```
center gap-1">
```

```
        <i class="ri-check-double-line"></i> Высокий шанс
```

```
</span>
```

```
    {% elif item.total_participants < 5 %} <span
```

```
        class="text-blue-600 text-base font-medium flex items-center
```

```
gap-1">
```

```
        <i class="ri-lightbulb-line"></i> Мало участников
```

```
</span>
```

```
    {% else %}
```

```
<span class="text-gray-400 text-base">Обычный</span>
{ % endif %

</td>
</tr>
{ % endfor %

</tbody>
</table>
</div>
</div>
{ % endif %

<div class="bg-white rounded-xl shadow-sm border border-gray-200 p-6 mb-[20px]">
<div class="flex flex-col sm:flex-row sm:items-center justify-between mb-6 gap-4">
<h2 class="text-subheading font-bold text-gray-900 mb-[5px]">
История достижений
</h2>
<div class="flex gap-2">
<select id="periodFilter"
class="bg-gray-50 border-none text-base font-medium text-gray-600 rounded-lg mb-[5px] px-3 py-2 focus:ring-2 focus:ring-primary cursor-pointer hover:bg-gray-100 transition-colors">
<option value="">Все семестры</option>
{ % for period in data.periods %

<option value="{{ period.id }}">{{ period.naming }}</option>
{ % endfor %

</select>
<select id="categoryFilter"
```

```
        class="bg-gray-50 border-none text-base font-medium text-gray-600 rounded-lg mb-[5px] px-3 py-2 focus:ring-2 focus:ring-primary cursor-pointer hover:bg-gray-100 transition-colors">  
            <option value="">Все категории</option>  
            {% for category in data.categories %}  
                <option value="{{ category.cod }}">{{ category.name }}</option>  
            {% endfor %}  
        </select>  
    </div>  
</div>  
  
{% if data.recent_achievements %}  
<div class="overflow-x-auto">  
    <table class="w-full min-w-full">  
        <thead>  
            <tr class="border-b border-gray-100">  
                <th class="text-left py-3 px-4 text-description font-semibold text-gray-400 uppercase tracking-wider">  
                    Категория</th>  
                <th class="text-left py-3 px-4 text-description font-semibold text-gray-400 uppercase tracking-wider">  
                    Описание</th>  
                <th class="text-left py-3 px-4 text-description font-semibold text-gray-400 uppercase tracking-wider">  
                    Баллы</th>  
                <th class="text-left py-3 px-4 text-description font-semibold text-gray-400 uppercase tracking-wider">  
                    Дата</th>  
                <th class="w-10"></th>  
            </tr>
```

```
</thead>

<tbody id="achievementsTableBody" class="divide-y divide-gray-50">
    { % for achievement in data.recent_achievements % }

    <tr class="hover:bg-gray-50 transition-colors group">
        <td class="py-3 px-4">
            <div class="flex items-center gap-2">
                <div
                    class="w-1.5 h-1.5 rounded-full
                    { % if achievement.category_name == 'Научная
                    деятельность' % }bg-blue-500{ % endif % }
                    { % if achievement.category_name == 'Культурная
                    деятельность' % }bg-purple-500{ % endif % }
                    { % if achievement.category_name == 'Спортивная
                    деятельность' % }bg-green-500{ % endif % }
                    { % if achievement.category_name == 'Общественная
                    деятельность' % }bg-yellow-500{ % endif % }">
                </div>
                <span class="text-base font-medium text-gray-700">{ {
                    achievement.category_name } }</span>
            </div>
        </td>
        <td class="py-4 px-4">
            <div class="text-base font-medium text-gray-900">{ {
                achievement.description_text } }</div>
            <div class="text-description text-gray-500 mt-0.5">
                { { achievement.level_title } }
                { % if achievement.document_title % }
                    <span class="mx-1 text-gray-300">|</span>{ {
                        achievement.document_title } }
                { % endif % }
            
```

```

        </div>
    </td>
<td class="py-4 px-4">
    <div class="flex items-center gap-1">
        <span class="text-base font-bold text-gray-900">{ {
achievement.points *
            achievement.quantity } }</span>
        { % if achievement.quantity > 1 % }
        <span class="text-description text-gray-400 bg-gray-100 px-1.5 rounded ml-1">x{ {
            achievement.quantity } }</span>
        { % endif % }
    </div>
</td>
<td class="py-4 px-4 text-base text-gray-500">
    { { achievement.created_at.strftime('%d.%m.%Y') } }
</td>
<td class="py-4 px-4 text-right">
    <button onclick="deleteAchievement('{ { achievement.id } }')"
        class="text-gray-400 hover:text-red-500 transition-colors opacity-0 group-hover:opacity-100 p-1">
        <i class="ri-delete-bin-line"></i>
    </button>
</td>
</tr>
{ % endfor % }
</tbody>
</table>
</div>
{ % else % }

```

```
<div class="text-center py-16">
    <div class="w-12 h-12 bg-gray-50 rounded-full flex items-center justify-center mx-auto mb-3">
        <i class="ri-add-circle-line text-text-subheading text-gray-300"></i>
    </div>
    <p class="text-gray-500 text-base">Нет достижений. Добавьте первое!</p>
</div>
{ % endif % }
</div>
</div>

<div id="addModal"
    class="fixed inset-0 bg-gray-900/50 backdrop-blur-sm hidden z-50 flex items-center justify-center p-4">
    <div class="bg-white rounded-xl shadow-2xl w-full max-w-2xl mx-auto transform transition-all relative">
        <div class="p-6">
            <div class="flex items-center justify-between mb-8">
                <div>
                    <h3 class="text-subheading font-bold text-gray-900">Новое достижение</h3>
                </div>
                <button onclick="closeAddForm()" class="text-gray-400 hover:text-gray-600 transition-colors">
                    <i class="ri-close-line text-xl"></i>
                </button>
            </div>
            <div class="flex items-center justify-between mb-8 px-4">
```

```
<div class="flex-1 flex items-center">
    <div class="step-indicator step-active" id="step1ind">1</div>
    <div class="flex-1 h-0.5 bg-gray-100 mx-2"></div>
    <div class="step-indicator step-inactive" id="step2ind">2</div>
    <div class="flex-1 h-0.5 bg-gray-100 mx-2"></div>
    <div class="step-indicator step-inactive" id="step3ind">3</div>
    <div class="flex-1 h-0.5 bg-gray-100 mx-2"></div>
    <div class="step-indicator step-inactive" id="step4ind">4</div>
</div>
</div>

<form id="achievementForm" method="POST" action="{{
url_for('add_achievement') }}">
    <input type="hidden" name="category_id" id="categoryId">

    <div id="step1" class="step-content">
        <h4 class="text-base font-semibold text-gray-500 uppercase tracking-
wide mb-4">Тип участия</h4>
        <div class="grid grid-cols-1 md:grid-cols-3 gap-4">
            <button
                type="button"
                onclick="selectAchievementType('participation')" id="btnParticipation"
                class="p-4 border border-gray-200 rounded-xl hover:border-
primary/50 hover:bg-blue-50/50 transition-all text-left group">
                <div class="mb-3 text-gray-400 group-hover:text-primary
transition-colors">
                    <i class="ri-user-smile-line text-text-subheading"></i>
                </div>
                <div class="font-semibold text-gray-900 mb-1">Участие</div>
                <div class="text-description text-gray-500">Без призового
места</div>
            
```

```
</button>

<button type="button" onclick="selectAchievementType('prize')"
id="btnPrize"
    class="p-4 border border-gray-200 rounded-xl hover:border-green-500/50 hover:bg-green-50/50 transition-all text-left group">
    <div class="mb-3 text-gray-400 group-hover:text-green-600
transition-colors">
        <i class="ri-trophy-line text-text-subheading"></i>
    </div>
    <div class="font-semibold text-gray-900 mb-1">Призовое
место</div>
    <div class="text-description text-gray-500">1-3 место, гран-
при</div>
</button>

<button type="button" onclick="selectAchievementType('other')"
id="btnOther"
    class="p-4 border border-gray-200 rounded-xl hover:border-
purple-500/50 hover:bg-purple-50/50 transition-all text-left group">
    <div class="mb-3 text-gray-400 group-hover:text-purple-600
transition-colors">
        <i class="ri-star-smile-line text-text-subheading"></i>
    </div>
    <div class="font-semibold text-gray-900 mb-1">Другое</div>
    <div class="text-description text-gray-500">Иные
активности</div>
</button>
</div>
</div>
```

```
<div id="step2" class="step-content" style="display: none;">
    <h4 class="text-base font-semibold text-gray-500 uppercase tracking-wide mb-4">Раздел</h4>
        <div class="grid grid-cols-1 md:grid-cols-2 gap-3" id="sectionButtons"></div>
            <div id="noSectionsMessage" class="hidden text-center py-8 text-gray-500">
                Нет доступных разделов
            </div>
        </div>

    <div id="step3" class="step-content" style="display: none;">
        <h4 class="text-base font-semibold text-gray-500 uppercase tracking-wide mb-4">Достижение</h4>
            <div class="space-y-2" id="criteriaList"></div>
            <div id="noCriteriaMessage" class="hidden text-center py-8 text-gray-500">
                Нет доступных критериев
            </div>
        </div>

    <div id="step4" class="step-content" style="display: none;">
        <div class="bg-gray-50 rounded-lg p-4 mb-6 flex justify-between items-start">
            <div>
                <div class="text-base text-gray-500" id="selectedCriteriaName"></div>
            </div>
        </div>
    </div>
</div>
```

```
<div class="font-semibold text-gray-900 mt-1" id="selectedCriteriaDesc"></div>

</div>
<div class="text-right">
    <span class="text-text-subheading font-bold text-primary" id="selectedCriteriaPoints">0</span>
        <div class="text-description text-gray-500">баллов</div>
    </div>
</div>

<div class="space-y-4">
    <div class="grid grid-cols-2 gap-4">
        <div>
            <label class="block text-base font-medium text-gray-700 mb-1">Уровень</label>
            <select id="levelSelect" name="level_type_id" class="w-full border-gray-300 rounded-lg text-base focus:ring-primary focus:border-primary"></select>
        </div>
        <div>
            <label class="block text-base font-medium text-gray-700 mb-1">Количество</label>
            <input type="number" name="quantity" value="1" min="1" class="w-full border-gray-300 rounded-lg text-base focus:ring-primary focus:border-primary">
        </div>
    </div>
</div>
```

```
<label class="block text-base font-medium text-gray-700 mb-1">Подтверждающий документ</label>
<input type="text" name="document_title"
       class="w-full border-gray-300 rounded-lg text-base focus:ring-primary focus:border-primary"
       placeholder="Например: Грамота №123">
</div>

<div>
    <label class="block text-base font-medium text-gray-700 mb-1">Семестр</label>
    <div class="w-full border border-gray-200 bg-gray-50 rounded-lg px-3 py-2 text-base text-gray-500 cursor-not-allowed">
        {{ data.current_period.naming }}
    </div>
    <input type="hidden" name="period_id" value="{{ data.current_period.id }}">
    </div>
    </div>
    </div>

    <input type="hidden" name="criteria_id" id="criteriaId">
    <input type="hidden" name="achievement_type" id="achievementType">

    <div class="mt-8 pt-4 border-t border-gray-100 flex justify-between items-center">
        <button type="button" onclick="prevStep()" id="prevBtn"
               class="text-gray-500 hover:text-gray-800 text-base font-medium px-4 py-2">
```

Назад

</button>

```
<div class="flex gap-2">
    <button type="button" onclick="closeAddForm()"
        class="text-gray-500 hover:text-gray-800 text-base font-medium
px-4 py-2">
        Отмена
    </button>
    <button type="button" onclick="nextStep()" id="nextBtn"
        class="bg-gray-900 text-white hover:bg-gray-800 px-6 py-2
rounded-lg text-base font-medium transition-colors">
        Далее
    </button>
    <button type="submit" id="submitBtn"
        class="bg-primary text-white hover:bg-blue-600 px-6 py-2
rounded-lg text-base font-medium transition-colors hidden">
        Сохранить
    </button>
</div>
</div>
</form>
</div>
</div>
</div>
{ % endblock % }
```

{ % block scripts % }

<script>

// --- Логика фильтрации ---

```
document.addEventListener('DOMContentLoaded', function () {
    const periodFilter = document.getElementById('periodFilter');
    const categoryFilter = document.getElementById('categoryFilter');
    if (periodFilter) periodFilter.addEventListener('change', applyFilters);
    if (categoryFilter) categoryFilter.addEventListener('change', applyFilters);
});

function applyFilters() {
    const periodId = document.getElementById('periodFilter').value;
    const categoryCod = document.getElementById('categoryFilter').value;
    const tableBody = document.getElementById('achievementsTableBody');

    tableBody.innerHTML = '<tr><td colspan="5" class="text-center py-8 text-gray-400">Загрузка...</td></tr>';
}

fetch('/filter_achievements?period_id=${periodId}&category_cod=${categoryCod}')
.then(response => response.json())
.then(data => {
    tableBody.innerHTML = "";
    if (data.length === 0) {
        tableBody.innerHTML = '<tr><td colspan="5" class="text-center py-12 text-gray-400">Нет записей</td></tr>';
        return;
    }

    data.forEach(achievement => {
        let colorClass = 'bg-gray-400';

```

```
        if (achievement.category_name === 'Научная деятельность')
colorClass = 'bg-blue-500';

        else if (achievement.category_name === 'Культурная деятельность')
colorClass = 'bg-purple-500';

        else if (achievement.category_name === 'Спортивная деятельность')
colorClass = 'bg-green-500';

        else if (achievement.category_name === 'Общественная
деятельность') colorClass = 'bg-yellow-500';

const row = `

<tr class="hover:bg-gray-50 transition-colors group">
  <td class="py-4 px-4">
    <div class="flex items-center gap-2">
      <div class="w-1.5 h-1.5 rounded-full ${colorClass}"></div>
      <span class="text-base font-medium text-gray-700">${achievement.category_name}</span>
    </div>
  </td>
  <td class="py-4 px-4">
    <div class="text-base font-medium text-gray-900">${achievement.description_text}</div>
    <div class="text-description text-gray-500 mt-0.5">
      ${achievement.level_title}
      ${achievement.document_title ? `<span class="mx-1 text-gray-300">|</span> ${achievement.document_title}</span>` : ""}
    </div>
  </td>
  <td class="py-4 px-4">
    <div class="flex items-center gap-1">
```

```
<span class="text-base font-bold text-gray-900">${achievement.points * achievement.quantity}</span>
    ${achievement.quantity > 1 ? `<span class="text-description text-gray-400 bg-gray-100 px-1.5 rounded ml-1">x${achievement.quantity}</span>` : ""}
</div>
</td>
<td class="py-4 px-4 text-base text-gray-500">
    ${achievement.created_at}
</td>
<td class="py-4 px-4 text-right">
    <button onclick="deleteAchievement('${achievement.id}')"
        class="text-gray-400 hover:text-red-500 transition-colors opacity-0 group-hover:opacity-100 p-1">
        <i class="ri-delete-bin-line"></i>
    </button>
</td>
</tr>
`;
tableBody.innerHTML += row;
});
})
.catch(error => {
    console.error('Ошибка:', error);
    tableBody.innerHTML = '<tr><td colspan="5" class="text-center py-4 text-red-500">Ошибка</td></tr>';
});
}

// --- Логика модального окна (Compact Style) ---
```

```
let currentStep = 1;
let currentCategoryCod = "";
let currentAchievementType = "";
let currentSection = "";
let sectionsData = {};
let availableCriteriaNames = [];
let currentSelectedName = "";

function showAddForm(categoryCod, categoryId) {
    currentCategoryCod = categoryCod;
    currentStep = 1;
    document.getElementById('categoryId').value = categoryId;

    // Сброс стилей кнопок первого шага
    ['btnParticipation', 'btnPrize', 'btnOther'].forEach(id => {
        const btn = document.getElementById(id);
        // Сбрасываем цвета к дефолтным серым
        btn.className = 'p-4 border border-gray-200 rounded-xl hover:border-gray-300 transition-all text-left group';

        // Иконки
        const iconDiv = btn.querySelector('div:first-child');
        iconDiv.className = 'mb-3 text-gray-400 group-hover:text-gray-600 transition-colors';
    });

    showStep(1);
    document.getElementById('addModal').classList.remove('hidden');
    loadCriteriaData();
}

}
```

```
function loadCriteriaData() {
    fetch(`/get_criteria_data/${currentCategoryCod}`)
        .then(r => r.json())
        .then(data => { sectionsData = data.sections || {} });
}

function selectAchievementType(type) {
    currentAchievementType = type;
    document.getElementById('achievementType').value = type;

    // Визуальное выделение
    const btnMap = { 'participation': 'btnParticipation', 'prize': 'btnPrize', 'other': 'btnOther' };
    const activeClassMap = {
        'participation': 'border-primary bg-blue-50',
        'prize': 'border-green-500 bg-green-50',
        'other': 'border-purple-500 bg-purple-50'
    };

    // Сброс всех
    ['btnParticipation', 'btnPrize', 'btnOther'].forEach(id => {
        const btn = document.getElementById(id);
        btn.className = `p-4 border border-gray-200 rounded-xl hover:border-gray-300 transition-all text-left group`;
    });

    // Активный
    const activeBtn = document.getElementById(btnMap[type]);
    activeBtn.className = `p-4 rounded-xl text-left group transition-all border-2 ${activeClassMap[type]}`;
}
```

```
    setTimeout(() => nextStep(), 200);

}

function showStep(step) {
    document.querySelectorAll('.step-content').forEach(el => el.style.display = 'none');
    document.getElementById(`step${step}`).style.display = 'block';

    updateStepIndicators(step);

    const nextBtn = document.getElementById('nextBtn');
    const submitBtn = document.getElementById('submitBtn');
    const prevBtn = document.getElementById('prevBtn');

    prevBtn.style.visibility = step > 1 ? 'visible' : 'hidden';

    if (step === 4) {
        nextBtn.classList.add('hidden');
        submitBtn.classList.remove('hidden');
    } else {
        nextBtn.classList.remove('hidden');
        submitBtn.classList.add('hidden');
    }

    if (step === 2) populateSections();
    if (step === 3) populateCriteria();
    if (step === 4) populateDetails();
}
```

```
function populateSections() {  
  const container = document.getElementById('sectionButtons');  
  container.innerHTML = "";  
  const uniqueSections = [...new Set(Object.values(sectionsData).filter(c => c.achievement_type === currentAchievementType).map(c => c.section_naming))];  
  
  if (uniqueSections.length === 0) {  
  
    document.getElementById('noSectionsMessage').classList.remove('hidden');  
    return;  
  }  
  document.getElementById('noSectionsMessage').classList.add('hidden');  
  
  uniqueSections.forEach(sec => {  
    const btn = document.createElement('button');  
    btn.type = 'button';  
    btn.className = 'p-3 border border-gray-200 rounded-lg hover:border-primary hover:bg-blue-50 text-left w-full text-base font-medium transition-colors';  
    btn.textContent = sec;  
    btn.onclick = () => { currentSection = sec; nextStep(); };  
    container.appendChild(btn);  
  });  
}  
  
function populateCriteria() {  
  const list = document.getElementById('criteriaList');  
  list.innerHTML = "";  
  const relevant = Object.values(sectionsData).filter(c => c.section_naming === currentSection && c.achievement_type === currentAchievementType);  
  availableCriteriaNames = [...new Set(relevant.map(c => c.description_text))];
```

```

if (availableCriteriaNames.length === 0) {
    document.getElementById('noCriteriaMessage').classList.remove('hidden');
    return;
}
document.getElementById('noCriteriaMessage').classList.add('hidden');

availableCriteriaNames.forEach(name => {
    const div = document.createElement('div');
    div.className = 'p-3 border border-gray-200 rounded-lg hover:border-primary hover:bg-blue-50 cursor-pointer flex justify-between items-center text-base transition-colors';
    div.innerHTML = `<span class="font-medium text-gray-700">${name}</span><i class="ri-arrow-right-s-line text-gray-400"></i>`;
    div.onclick = () => { currentSelectedName = name; nextStep(); };
    list.appendChild(div);
});
}

function populateDetails() {
    document.getElementById('selectedCriteriaName').textContent = currentSection;
    document.getElementById('selectedCriteriaDesc').textContent = currentSelectedName;
}

const levelSelect = document.getElementById('levelSelect');
levelSelect.innerHTML = "";

const variants = Object.values(sectionsData).filter(c => c.section_naming === currentSection && c.description_text === currentSelectedName);

```

```

variants.forEach(v => {
    const opt = document.createElement('option');
    opt.value = v.level_type_id;
    opt.textContent = v.level_title;
    opt.dataset.points = v.points;
    opt.dataset.realId = v.id;
    levelSelect.appendChild(opt);
});

const updatePoints = () => {
    const opt = levelSelect.options[levelSelect.selectedIndex];
    if (opt) {
        document.getElementById('selectedCriteriaPoints').textContent =
        opt.dataset.points;
        document.getElementById('criteriaId').value = opt.dataset.realId;
    }
};

levelSelect.onchange = updatePoints;
// Инициализация первого значения
if (variants.length > 0) updatePoints();

}

function updateStepIndicators(step) {
    for (let i = 1; i <= 4; i++) {
        const el = document.getElementById(`step${i}ind`);
        if (i <= step) {
            el.classList.remove('step-inactive');
            el.classList.add('step-active');
        }
    }
}

```

```

    } else {
        el.classList.remove('step-active');
        el.classList.add('step-inactive');
    }
}

function nextStep() { if (currentStep < 4) showStep(++currentStep); }
function prevStep() { if (currentStep > 1) showStep(--currentStep); }
function closeAddForm() {
document.getElementById('addModal').classList.add('hidden'); }

function deleteAchievement(id) { if (confirm('Удалить запись?')) window.location.href = `/delete_achievement/${id}`; }

</script>
{ % endblock %

```

Код – app.py

```

from flask import Flask, render_template, session, request, redirect, url_for, flash, jsonify
import pymysql as db
from config import host, user, password, port, db_name

app = Flask(__name__)
app.secret_key = 'мой_секретный_ключ'

# Подключение к базе данных
def get_db_connection():
    """Получить соединение с БД"""
try:
    connection = db.connect(

```

```
host=host,
port=port,
user=user,
password=password,
database=db_name,
cursorclass=db.cursors.DictCursor,
autocommit=True

)

return connection

except Exception as ex:
    print("Ошибка подключения к БД:", ex)
    return None

@app.route('/login', methods=['GET', 'POST'])
def login():

    """Страница входа"""

    if request.method == 'POST':
        email = request.form['email']
        password_input = request.form['password']

        conn = get_db_connection()
        if conn:
            try:
                with conn.cursor() as cursor:
                    # Проверяем логин и пароль
                    sql = """
SELECT s.*, sg.naming as group_name, sg.course,
       sp.title as specialty_name, sp.cod_specialty,
       fs.title as form_study_name,
       t.surname as tutor_surname, t.first_name as tutor_first_name,
       t.last_name as tutor_last_name
FROM student s
JOIN student_group sg ON s.id = sg.student_id
JOIN specialty sp ON sg.specialty_id = sp.id
JOIN form_study fs ON sg.form_study_id = fs.id
JOIN teacher t ON s.tutor_id = t.id
WHERE s.email = %s AND s.password = %s
"""
                    cursor.execute(sql, (email, password_input))
                    result = cursor.fetchone()
                    if result:
                        session['user_id'] = result[0]
                        session['group_name'] = result[1]
                        session['course'] = result[2]
                        session['specialty_name'] = result[3]
                        session['form_study_name'] = result[4]
                        session['tutor_surname'] = result[5]
                        session['tutor_first_name'] = result[6]
                        session['tutor_last_name'] = result[7]
                        return redirect(url_for('index'))
                    else:
                        flash('Неверный логин или пароль')
            except Exception as e:
                print(f'Ошибка при выполнении запроса: {e}')
        else:
            flash('Не удалось подключиться к базе данных')
    else:
        return render_template('login.html')
```

```
t.middle_name as tutor_middle_name
FROM Student s
JOIN Student_group sg ON s.student_group_id = sg.id
JOIN Specialty sp ON sg.specialty_id = sp.id
JOIN Form_study fs ON sg.form_study_id = fs.id
JOIN tutor t ON sg.tutor_id = t.id
WHERE s.email = %s AND s.student_password = %s
"""

cursor.execute(sql, (email, password_input))
student = cursor.fetchone()

if student:
    # Сохраняем данные в сессии
    session['student_id'] = student['id']
    session['student_name'] = f'{student["surname"]}'
    session['student_first_name'] = student['first_name']
    session['student_email'] = student['email']
    session['student_photo'] = student.get('profile_photo', "")
    session['student_data'] = student

    flash('Вход выполнен успешно!', 'success')
    return redirect(url_for('profile'))

else:
    flash('Неверный email или пароль', 'error')

except Exception as e:
    print("Ошибка при авторизации:", e)
    flash('Ошибка при входе в систему', 'error')

finally:
    conn.close()
```

```
        return render_template('login.html')

@app.route('/logout')
def logout():
    """Выход из системы"""
    session.clear()
    flash('Вы вышли из системы', 'success')
    return redirect(url_for('login'))

@app.route('/')
def index():
    """Главная страница - перенаправление"""
    if 'student_id' in session:
        return redirect(url_for('profile'))
    return redirect(url_for('login'))

@app.route('/profile')
def profile():
    """Страница профиля"""
    if 'student_id' not in session:
        return redirect(url_for('login'))

    conn = get_db_connection()
    student = None

    if conn:
        try:
            with conn.cursor() as cursor:
                # Получаем данные студента
                sql = """
```

```
SELECT s.*,
       DATE_FORMAT(s.birth_date,      '%d.%m.%Y')      as
birth_date_str,
       sg.naming as group_name, sg.course,
       sp.title as specialty_name, sp.cod_specialty,
       fs.title as form_study_name,
       t.surname as tutor_surname, t.first_name as tutor_first_name,
       t.middle_name as tutor_middle_name,
       YEAR(s.created_at) as created_year,
       YEAR(DATE_ADD(s.created_at,  INTERVAL 4 YEAR)) as
graduation_year
```

```
FROM Student s
JOIN Student_group sg ON s.student_group_id = sg.id
JOIN Specialty sp ON sg.specialty_id = sp.id
JOIN Form_study fs ON sg.form_study_id = fs.id
JOIN tutor t ON sg.tutor_id = t.id
WHERE s.id = %s
""""
```

```
cursor.execute(sql, (session['student_id'],))
student = cursor.fetchone()
```

```
if not student:
```

```
    flash('Данные студента не найдены', 'error')
    return redirect(url_for('login'))
```

```
except Exception as e:
```

```
    print("Ошибка при получении данных профиля:", e)
    flash('Ошибка при загрузке данных профиля', 'error')
```

```
finally:
```

```
    conn.close()
```

```
return render_template('profile.html', student=student)

@app.route('/update_profile', methods=['POST'])
def update_profile():
    """Обновление профиля с валидацией"""
    if 'student_id' not in session:
        return redirect(url_for('login'))

    email = request.form.get('email', "").strip()
    phone = request.form.get('phone', "").strip()
    address = request.form.get('address', "").strip()

    # Валидация email
    import re

    email_pattern = r'^[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}$'
    if not re.match(email_pattern, email):
        flash('Email должен быть в формате example@domain.com', 'error')
        return redirect(url_for('profile'))

    # Валидация телефона (российский формат: +7XXXXXXXXXX)
    phone_pattern = r'^\+7\d{10}$'
    if not re.match(phone_pattern, phone):
        flash('Телефон должен быть в формате +7XXXXXXXXXX (10 цифр после +7)', 'error')
        return redirect(url_for('profile'))

    # Валидация адреса (минимум 5 символов, должен содержать улицу и дом)
    if len(address) < 5:
```

```
flash('Адрес должен содержать не менее 5 символов', 'error')
return redirect(url_for('profile'))

if not any(word in address.lower() for word in ['ул.', 'улица', 'проспект', 'пр.',
'dом', 'д.']):
    flash('Адрес должен содержать указание на улицу и дом (например, "ул.
Ленина, д. 10")', 'error')
return redirect(url_for('profile'))

conn = get_db_connection()
if conn:
    try:
        with conn.cursor() as cursor:
            # Проверяем, не занят ли email другим пользователем
            check_sql = "SELECT id FROM Student WHERE email = %s AND id != %s"
            cursor.execute(check_sql, (email, session['student_id']))
            if cursor.fetchone():
                flash('Этот email уже используется другим пользователем', 'error')
                return redirect(url_for('profile'))

            # Проверяем, не занят ли телефон другим пользователем
            check_sql = "SELECT id FROM Student WHERE phone = %s AND id != %s"
            cursor.execute(check_sql, (phone, session['student_id']))
            if cursor.fetchone():
                flash('Этот телефон уже используется другим пользователем',
'error')
                return redirect(url_for('profile'))
```

```
# Обновляем данные
update_sql = """
UPDATE Student
SET email = %s, phone = %s, address = %s
WHERE id = %s
"""

cursor.execute(update_sql, (email, phone, address, session['student_id']))
conn.commit()

# Обновляем email в сессии
session['student_email'] = email

flash('Профиль успешно обновлен!', 'success')

except Exception as e:
    print("Ошибка при обновлении профиля:", e)
    flash('Ошибка при обновлении профиля', 'error')
finally:
    conn.close()

return redirect(url_for('profile'))

@app.route('/update_password', methods=['POST'])
def update_password():
    """Обновление пароля"""
    if 'student_id' not in session:
        return redirect(url_for('login'))

    current_password = request.form.get('current_password', '')
    new_password = request.form.get('new_password', '')
```

```
confirm_password = request.form.get('confirm_password', "")\n\nif new_password != confirm_password:\n    flash('Новый пароль и подтверждение не совпадают', 'error')\n    return redirect(url_for('profile'))\n\nconn = get_db_connection()\n\nif conn:\n    try:\n        with conn.cursor() as cursor:\n            # Проверяем текущий пароль\n            check_sql = "SELECT id FROM Student WHERE id = %s AND student_password = %s"\n            cursor.execute(check_sql, (session['student_id'], current_password))\n            if not cursor.fetchone():\n                flash('Текущий пароль неверен', 'error')\n                return redirect(url_for('profile'))\n\n            # Обновляем пароль\n            update_sql = "UPDATE Student SET student_password = %s WHERE id = %s"\n            cursor.execute(update_sql, (new_password, session['student_id']))\n            conn.commit()\n\n            flash('Пароль успешно изменен!', 'success')\n\n    except Exception as e:\n        print("Ошибка при изменении пароля:", e)\n        flash('Ошибка при изменении пароля', 'error')\n\n    finally:
```

```
conn.close()

return redirect(url_for('profile'))

@app.route('/upload_photo', methods=['POST'])
def upload_photo():

    """Загрузка фото профиля с валидацией"""

    if 'student_id' not in session:
        return redirect(url_for('login'))

    photo_url = request.form.get('photo_url', "").strip()

    # Базовая валидация URL
    if not photo_url:
        flash('Пожалуйста, укажите URL фотографии', 'error')
        return redirect(url_for('profile'))

    # Проверяем, что URL заканчивается на расширение изображения
    valid_extensions = ('.jpg', '.jpeg', '.png', '.gif', '.bmp', '.webp')
    if not photo_url.lower().endswith(valid_extensions):
        flash('URL должен вести на изображение (jpg, png, gif, bmp, webp)', 'error')
        return redirect(url_for('profile'))

    # Проверяем формат URL
    if not (photo_url.startswith('http://') or photo_url.startswith('https://')):
        flash('URL должен начинаться с http:// или https://', 'error')
        return redirect(url_for('profile'))

conn = get_db_connection()
if conn:
```

```
try:
    with conn.cursor() as cursor:
        update_sql = "UPDATE Student SET profile_photo = %s WHERE id = %s"
        cursor.execute(update_sql, (photo_url, session['student_id']))
        conn.commit()

    # Обновляем фото в сессии
    session['student_photo'] = photo_url

    flash('Фото профиля успешно обновлено!', 'success')

except Exception as e:
    print("Ошибка при обновлении фото:", e)
    flash('Ошибка при обновлении фото профиля', 'error')
finally:
    conn.close()

return redirect(url_for('profile'))

@app.route('/notifications')
def notifications():
    """Страница уведомлений"""
    if 'student_id' not in session:
        return redirect(url_for('login'))
    return render_template('notifications.html')

@app.route('/messages')
def messages():
    """Страница сообщений"""
```

```
if 'student_id' not in session:  
    return redirect(url_for('login'))  
return render_template('messages.html')  
  
@app.route('/grades')  
def grades():  
    """Страница успеваемости"""  
    if 'student_id' not in session:  
        return redirect(url_for('login'))  
    return render_template('grades.html')  
  
@app.route('/materials')  
def materials():  
    """Страница учебных материалов"""  
    if 'student_id' not in session:  
        return redirect(url_for('login'))  
    return render_template('materials.html')  
  
@app.route('/portfolio')  
def portfolio():  
    """Страница портфолио/рейтинга"""  
    if 'student_id' not in session:  
        return redirect(url_for('login'))  
  
    student_id = session['student_id']  
    student = session['student_data']  
  
    conn = get_db_connection()  
    data = {  
        'categories': [],
```

```
'total_points': 0,
'achievements': [],
'periods': [],
'analysis': []
}

if conn:
    try:
        with conn.cursor() as cursor:
            # 1. Получаем все категории деятельности
            cursor.execute("SELECT id, naming, cod FROM Activity_Category")
            categories = cursor.fetchall()

            # 2. Получаем все доступные периоды (для истории)
            cursor.execute("SELECT id, naming FROM Academic_Period ORDER
BY start_date DESC")
            data['periods'] = cursor.fetchall()

            # --- НОВОЕ: ОПРЕДЕЛЯЕМ ТЕКУЩИЙ СЕМЕСТР ---
            # Ищем семестр, который идет сейчас (по дате)
            cursor.execute("SELECT id, naming FROM Academic_Period WHERE
end_date >= CURDATE() ORDER BY start_date LIMIT 1")
            current_period = cursor.fetchone()

            # Если текущего нет (каникулы), берем самый последний из
            # добавленных
            if not current_period and data['periods']:
                current_period = data['periods'][0]

            data['current_period'] = current_period
```

```

# -----



# 3. Для каждой категории получаем достижения студента
for category in categories:

    # Используем ID текущего семестра, если он найден, иначе None
    period_filter_id = current_period['id'] if current_period else 0

    # Баллы по категории за ТЕКУЩИЙ семестр
    sql = """
        SELECT rc.section_naming, rc.description_text, lt.title as level,
               rc.points, sa.quantity, (rc.points * sa.quantity) as total,
               sa.created_at, sa.document_title
        FROM Student_Achievement sa
        JOIN Rating_Criteria rc ON sa.criteria_id = rc.id
        JOIN level_type lt ON rc.level_type_id = lt.id
        WHERE sa.student_id = %s AND rc.category_id = %s
        AND sa.period_id = %s -- Фильтр по текущему семестру
        """
    cursor.execute(sql, (student_id, category['id'], period_filter_id))
    achievements = cursor.fetchall()

    # Сумма баллов по категории (за текущий семестр)
    points_sql = """
        SELECT SUM(rc.points * sa.quantity) as category_total
        FROM Student_Achievement sa
        JOIN Rating_Criteria rc ON sa.criteria_id = rc.id
        WHERE sa.student_id = %s AND rc.category_id = %s
        AND sa.period_id = %s
        """
    cursor.execute(points_sql, (student_id, category['id'], period_filter_id))

```

```

points_result = cursor.fetchone()
category_total = points_result['category_total'] or 0

count_sql = """
SELECT SUM(sa.quantity) as total_count
FROM Student_Achievement sa
JOIN Rating_Criteria rc ON sa.criteria_id = rc.id
WHERE sa.student_id = %s AND rc.category_id = %s
AND sa.period_id = %s
"""

cursor.execute(count_sql, (student_id, category['id'], period_filter_id))
count_result = cursor.fetchone()
total_count = count_result['total_count'] or 0

data['categories'].append({
    'id': category['id'],
    'name': category['naming'],
    'cod': category['cod'],
    'points': category_total,
    'achievements': achievements,
    'total_count': total_count
})
data['total_points'] += category_total

# 4. Получаем последние достижения для отображения (история)
cursor.execute("""
    SELECT sa.*, rc.description_text, rc.points, ac.naming as
category_name,
        ap.naming as period_name, lt.title as level_title
    FROM Student_Achievement sa
""")
```

```
        JOIN Rating_Criteria rc ON sa.criteria_id = rc.id
        JOIN Activity_Category ac ON rc.category_id = ac.id
        JOIN Academic_Period ap ON sa.period_id = ap.id
        JOIN level_type lt ON rc.level_type_id = lt.id
        WHERE sa.student_id = %s
        ORDER BY sa.created_at DESC
    """", (student_id,))
    data['recent_achievements'] = cursor.fetchall()

    # 5. Анализ конкуренции
    if current_period:
        cursor.callproc('GetCourseRatingAnalysis',
                        (student_id,
                         current_period['id']))
        data['analysis'] = cursor.fetchall()

    except Exception as e:
        # ... (остальной код ошибки без изменений)
        print("Ошибка при получении данных портфолио:", e)
        flash('Ошибка при загрузке данных рейтинга', 'error')
    finally:
        conn.close()

    return render_template('portfolio.html', data=data, student=student)

@app.route('/add_achievement', methods=['POST'])
def add_achievement():
    """Добавление достижения"""
    if 'student_id' not in session:
        return redirect(url_for('login'))
```

```
# Получаем данные из формы
criteria_id = request.form.get('criteria_id')
quantity = request.form.get('quantity', 1)
document_title = request.form.get('document_title', "")
period_id = request.form.get('period_id')

conn = get_db_connection()
if conn:
    try:
        with conn.cursor() as cursor:
            sql = """
                INSERT INTO Student_Achievement
                (student_id, criteria_id, period_id, quantity, document_title, created_at)
                VALUES (%s, %s, %s, %s, %s, NOW())
            """
            cursor.execute(sql, (
                session['student_id'],
                criteria_id,
                period_id,
                quantity,
                document_title
            ))
            conn.commit()

        flash('Достижение успешно добавлено!', 'success')

    except Exception as e:
        # Если это наша ошибка из триггера, текст будет внутри e
        error_msg = str(e)
        if "Документ с таким названием уже загружен" in error_msg:
```

```
flash('Ошибка: Такой документ уже существует!', 'error')

else:
    print("Ошибка при добавлении достижения:", e)
    flash('Ошибка при добавлении достижения', 'error')

finally:
    conn.close()

return redirect(url_for('portfolio'))

@app.route('/delete_achievement/<int:achievement_id>')
def delete_achievement(achievement_id):
    """Удаление достижения"""
    if 'student_id' not in session:
        return redirect(url_for('login'))

    conn = get_db_connection()
    if conn:
        try:
            with conn.cursor() as cursor:
                # Проверяем, что достижение принадлежит студенту
                check_sql = "SELECT id FROM Student_Achievement WHERE id = %s
AND student_id = %s"
                cursor.execute(check_sql, (achievement_id, session['student_id']))
                if cursor.fetchone():
                    delete_sql = """
DELETE
FROM Student_Achievement
WHERE id = %s"""
                    cursor.execute(delete_sql, (achievement_id,))


```

```
    conn.commit()

    flash('Достижение удалено', 'success')

else:

    flash('Достижение не найдено или недоступно', 'error')

except Exception as e:

    print("Ошибка при удалении достижения:", e)
    flash('Ошибка при удалении достижения', 'error')

finally:

    conn.close()

return redirect(url_for('portfolio'))

@app.route('/get_criteria/<category_cod>')
def get_criteria(category_cod):

    """Получение критериев для категории (для AJAX)"""

    conn = get_db_connection()
    criteria = []

    if conn:

        try:

            with conn.cursor() as cursor:
                sql = """
SELECT rc.id, rc.section_naming, rc.description_text,
       lt.title as level, rc.points
  FROM Rating_Criteria rc
 JOIN Activity_Category ac ON rc.category_id = ac.id
 JOIN level_type lt ON rc.level_type_id = lt.id
 WHERE ac.cod = %s
 ORDER BY rc.section_naming, rc.points DESC
                """

                cursor.execute(sql, (category_cod,))
                criteria = cursor.fetchall()

        except Exception as e:
            print("Ошибка при получении критериев:", e)
            flash('Ошибка при получении критериев', 'error')

    conn.close()

    return jsonify(criteria)
```

```

"""
cursor.execute(sql, (category_cod,))
criteria = cursor.fetchall()

except Exception as e:
    print("Ошибка при получении критериев:", e)
finally:
    conn.close()

return jsonify(criteria)

@app.route('/get_criteria_data/<category_cod>')
def get_criteria_data(category_cod):
    """Получение структурированных данных по критериям"""
    conn = get_db_connection()
    data = {
        'sections': { },
        'levels': { }
    }

    if conn:
        try:
            with conn.cursor() as cursor:
                # 1. Получаем все критерии для категории
                sql = """
SELECT rc.id, rc.section_naming, rc.description_text,
       rc.level_type_id, rc.points, rc.achievement_type,
       lt.title as level_title
FROM Rating_Criteria rc
JOIN Activity_Category ac ON rc.category_id = ac.id
JOIN level_type lt ON rc.level_type_id = lt.id
"""

                cursor.execute(sql, (category_cod,))
                criteria = cursor.fetchall()

                for row in criteria:
                    section_id = row['section_naming']
                    if section_id not in data['sections']:
                        data['sections'][section_id] = {
                            'name': row['section_naming'],
                            'levels': []
                        }
                    level_id = row['level_type_id']
                    if level_id not in data['sections'][section_id]['levels']:
                        data['sections'][section_id]['levels'].append({
                            'name': row['level_title'],
                            'points': row['points'],
                            'achievement_type': row['achievement_type']
                        })

                return jsonify(data)
        except Exception as e:
            print("Ошибка при получении критериев:", e)

```

```

WHERE ac.cod = %s
ORDER BY rc.section_naming, rc.points DESC
"""

cursor.execute(sql, (category_cod,))
criteria_list = cursor.fetchall()

# Структурируем по ID
for criteria in criteria_list:
    data['sections'][criteria['id']] = {
        'id': criteria['id'],
        'section_naming': criteria['section_naming'],
        'description_text': criteria['description_text'],
        'level_type_id': criteria['level_type_id'],
        'points': criteria['points'],
        'achievement_type': criteria.get('achievement_type', 'other'),
        'level_title': criteria['level_title']
    }

# 2. Получаем все уровни
cursor.execute("SELECT id, title FROM level_type ORDER BY id")
levels = cursor.fetchall()
for level in levels:
    data['levels'][level['id']] = {
        'id': level['id'],
        'title': level['title']
    }

except Exception as e:
    print("Ошибка при получении структурированных данных:", e)
    return jsonify({'error': str(e)}), 500

```

```
finally:  
    conn.close()  
  
return jsonify(data)  
  
@app.route('/filter_achievements')  
def filter_achievements():  
    """API для фильтрации достижений (AJAX)"""  
    if 'student_id' not in session:  
        return jsonify({'error': 'Unauthorized'}), 401  
  
    student_id = session['student_id']  
    period_id = request.args.get('period_id')  
    category_cod = request.args.get('category_cod')  
  
    conn = get_db_connection()  
    achievements = []  
  
    if conn:  
        try:  
            with conn.cursor() as cursor:  
                # Базовый SQL запрос  
                sql = """  
                    SELECT sa.id, sa.quantity, sa.created_at, sa.document_title,  
                           rc.description_text, rc.points,  
                           ac.naming as category_name, ac.cod as category_cod,  
                           lt.title as level_title  
                  FROM Student_Achievement sa  
                  JOIN Rating_Criteria rc ON sa.criteria_id = rc.id  
                  JOIN Activity_Category ac ON rc.category_id = ac.id  
                """  
                cursor.execute(sql)  
                achievements = cursor.fetchall()  
        except Exception as e:  
            print(f"An error occurred: {e}")  
    return jsonify(achievements)
```

```
        JOIN level_type lt ON rc.level_type_id = lt.id
        WHERE sa.student_id = %s
"""

params = [student_id]

# Динамически добавляем условия фильтрации
if period_id:
    sql += " AND sa.period_id = %s"
    params.append(period_id)

if category_cod:
    sql += " AND ac.cod = %s"
    params.append(category_cod)

sql += " ORDER BY sa.created_at DESC"

cursor.execute(sql, tuple(params))
raw_data = cursor.fetchall()

# Преобразуем дату в строку для JSON
for item in raw_data:
    item['created_at'] = item['created_at'].strftime('%d.%m.%Y')
    achievements.append(item)

except Exception as e:
    print("Ошибка фильтрации:", e)
finally:
    conn.close()

return jsonify(achievements)
```

```
@app.route('/decanat')
def decanat():
    """Страница онлайн-деканата"""
    if 'student_id' not in session:
        return redirect(url_for('login'))
    return render_template('decanat.html')

@app.route('/stipendii')
def stipendii():
    """Страница стипендий"""
    if 'student_id' not in session:
        return redirect(url_for('login'))
    return render_template('stipendii.html')

if __name__ == '__main__':
    app.run(debug=True)
```