**СОДЕРЖАНИЕ**

[Аннотация 4](#_Toc72446649)

[Введение 5](#_Toc72446650)

[1 ОБЩИЕ СВЕДЕНИЯ О ПРОГРАММНОМ СРЕДСТВЕ 6](#_Toc72446651)

[1.1 Основное функциональное назначение программного средства 6](#_Toc72446652)

[1.2 Полное наименование программного средства 6](#_Toc72446653)

[1.3 Условное обозначение программного средства 6](#_Toc72446654)

[1.4 Разработчики программного средства 6](#_Toc72446655)

[2 ТЕХНИЧЕСКОЕ ЗАДАНИЕ 7](#_Toc72446656)

[2.1 Основание для разработки 7](#_Toc72446657)

[2.2 Назначение разработки 7](#_Toc72446658)

[2.3 Требование к программному средству 7](#_Toc72446659)

[2.4 Требования к программной документации 8](#_Toc72446660)

[2.5 Требования к эргономике и технической эстетике 8](#_Toc72446661)

[2.6 Стадии и этапы разработки 9](#_Toc72446662)

[2.7 Порядок контроля и приемки 10](#_Toc72446663)

[3 ПОЯСНИТЕЛЬНАЯ ЗАПИСКА К ПРОГРАММНОМУ ПРОДУКТУ 11](#_Toc72446664)

[3.1 Декомпозиция поставленной задачи 11](#_Toc72446665)

[3.2 Общая архитектура программного средства 12](#_Toc72446666)

[3.3 Разработка алгоритма решения задачи 15](#_Toc72446667)

[3.4 Реализация функционального назначения программного средства 16](#_Toc72446668)

[3.5 Структурная организация данных 16](#_Toc72446669)

[3.6 Разработка интерфейса ПС 23](#_Toc72446670)

[3.7 Описание структуры выходной информации 29](#_Toc72446671)

[4 РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ 31](#_Toc72446672)

[4.1 Назначение программного средства 31](#_Toc72446673)

[4.2 Условия выполнения программного средства 31](#_Toc72446674)

[4.3 Эксплуатация программного средства 31](#_Toc72446675)

[4.4 Сообщения пользователю 40](#_Toc72446676)

[5 РАЗРАБОТКА ТЕСТОВОГО НАБОРА 41](#_Toc72446677)

[5.1 Определение верхней границы количества тестов 41](#_Toc72446678)

[5.2 Разработка тестовых вариантов 42](#_Toc72446679)

[5.3 Составление отчета о тестировании 44](#_Toc72446680)

[ЗАКЛЮЧЕНИЕ 48](#_Toc72446681)

[Оценка качества программного средства с помощью метрик 48](#_Toc72446682)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 51](#_Toc72446683)

[Приложение А Текст кода программного средства 51](#_Toc72446684)

[Приложение Б Спецификация 84](#_Toc72446685)

[Приложение B Диск с программным средством 86](#_Toc72446686)

**АННОТАЦИЯ**

На дипломную работу студента Белова А.С. на тему: Разработка программного модуля студента колледжа. Дипломная работа представлена пояснительной запиской на 79 листов машинописного текста. Состоит из введения, 5 глав, заключения и списка литературных источников.

В главе 1 содержится общие сведения о программном средстве ПМ студента колледжа. В главе 2 приведено техническое задание на проектирование БД согласно ГОСТ 19.201. Глава 3 содержит пояснительную записку к программному продукту согласно ГОСТ 19.404-79. В главе 4 содержится руководство пользователя согласно ГОСТ 19.504-79. Глава 5 содержит обоснование количества необходимых тестов и тестовые пакеты для оценки работоспособности программного продукта. В главе 6 приведены результаты ручного и автоматического тестирования программного продукта и выполнена оценка качества с помощью метрик.

В работе содержится общее описание программного средства, предназначенного для объекта проектирования – ПМ студента колледжа, изложены требования к программному средству, описаны его архитектура, реализация функций, интерфейс, приведено руководство пользователя, обоснованы тестовые пакеты и выполнена оценка качества программного продукта с помощью метрик.

**ВВЕДЕНИЕ**

В дипломной работе была поставлена задача разработки приложения для ПМ студента колледжа, который позволяет упростить работу с физическими журналами, переведя их в электронный формат, просмотра долгов и хранения своих достижений.

Для достижения поставленной цели был составлен основной алгоритм работы программы, а также разработаны алгоритмы вспомогательных процедур.

Для реализации алгоритмов был выбран язык программирования JavaScript. Выбор обусловлен тем, что возможностей этого языка достаточно для достижения поставленной цели моей дипломной работы.

При разработке интерфейса программы, я учитывал наиболее нужные и важные функциональные возможности, которые должна выполнять данная программа, а также тип пользователей, которые с ней будут работать.

**1 ОБЩИЕ СВЕДЕНИЯ О ПРОГРАММНОМ СРЕДСТВЕ**

**1.1 Основное функциональное назначение программного средства**

Программа предназначена для реализации электронного журнала, просмотра долгов и хранения своих достижения. Программа не требует владения современными информационными технологиями для эффективного использования данного программного средства, т.е. подходит для любого пользователя.

**1.2 Полное наименование программного средства**

Программный модуль студента колледжа.

**1.3 Условное обозначение программного средства**

«StudentCabinet 1.0»

**1.4 Разработчики программного средства**

Программное средство разработал студент Зеленодольского Механического колледжа группы 205 Белов А.С.

**2 ТЕХНИЧЕСКОЕ ЗАДАНИЕ**

**2.1 Основание для разработки**

Работа ведётся на основании задания на дипломную работу по специальности 09.02.03 — «Программирование в компьютерных системах».

**2.2 Назначение разработки**

Программа «StudentCabinet 1.0» работает под управление операционной системы Windows 7, Windows 8, Windows 10. Программа предназначена для автоматизации деятельности ПМ студента колледжа. Цель создания программы – реализация электронного журнала, хранения своих достижений и просмотра долгов. Программа позволяет:

* просматривать оценки по семестрам и предметам;
* просматривать свои долги;
* создавать базу пользователей;
* добавление достижений в базу;
* вступать в организации существующие в колледже.

**2.3 Требование к программному средству**

«StudentCabinet 1.0» устанавливается на персональный компьютер, имеющий представленные минимальные системные требования:

Оперативная память: 4 Гб;

Процессор: Intel Core i / Phenom II X4 / AMD Athlon64 X2 4600 + 2.4 ГГц;

Видео карта: nVidia GeForce 8600 / ATI Radeon HD 2400;

Звуковая карта: Совместимая с DirectX;

Программный продукт «StudentCabinet 1.0» является кроссплатформенным продуктом, что позволяет не ограничиваться в работе с определенной версией ОС.

«StudentCabinet 1.0» написана на мультпарадигменном языке программировании JavaScript (ДжаваСкрипт) в текстовом редактореVisual Studio Code. В качестве СУБД используется PostgreSQL. Программа не может подлежать сторонней оптимизации и редактированию программного кода.

**2.4 Требования к программной документации**

Состав программной документации должен включать в себя:

1. техническое задание;

2. сроки выполнения работ;

3. программу и методику испытаний;

4. эксплуатационные инструкции пользователю;

**2.5 Требования к эргономике и технической эстетике**

Система обеспечивает удобный для пользователей системный интерфейс, отвечающий следующим требованиям:

- единый стиль оформления для пользовательских интерфейсов;

- удобная, интуитивно понятная навигация в интерфейсе пользователя;

- взаимодействие пользователя с системой осуществляется на двух языках по выбору (русский и английский), для наибольшего удобства

Пользовательские интерфейсы системы спроектированы и разработаны с применением единых принципов графического представления информации и организации доступа к функциональным возможностям и сервисам. Разработан графический дизайн пользовательских интерфейсов, цветовые, шрифтовые и композиционные решения для отображения текстов, изображений, таблиц, гиперссылок, управляющих и навигационных элементов (меню, кнопок, форм и т.п.).

Экранные формы спроектированы с учетом требований унификации:

- все экранные формы пользовательского интерфейса выполнены в едином графическом дизайне, с одинаковым расположением основных элементов управления и навигации;

- для обозначения сходных операций использованы сходные графические значки, кнопки и другие управляющие (навигационные) элементы. Термины, используемые для обозначения типовых операций (добавление информационной сущности, редактирование поля данных), а также последовательности действий пользователя при их выполнении унифицированы;

- внешнее поведение сходных элементов интерфейса (реакция на наведение указателя «мыши», переключение фокуса, нажатие кнопки) реализованы одинаково для однотипных элементов.

**2.6 Стадии и этапы разработки**

Разработка должна быть проведена в три стадии:

1. разработка технического задания;

2. рабочее проектирование;

3. внедрение.

На стадии разработки технического задания должен быть выполнен этап разработки, согласования и утверждения настоящего технического задания.

На стадии рабочего проектирования должны быть выполнены перечисленные ниже этапы работ:

1. разработка программы;

2. разработка программной документации;

3. испытания программы.

На этапе разработки технического задания должны быть выполнены перечисленные ниже работы:

1. постановка задачи;

2. определение и уточнение требований к техническим средствам;

3. определение требований к программе;

4. определение стадий, этапов и сроков разработки программы и документации на нее;

5. выбор языков программирования;

6. согласование и утверждение технического задания.

На этапе разработки программы должна быть выполнена работа по программированию (кодированию) и отладке программы.

На этапе разработки программной документации должна быть выполнена разработка программных документов в соответствии с пунктом. Предварительный состав программной документации настоящего технического задания.

На этапе испытаний программы должны быть выполнены перечисленные ниже виды работ:

1. разработка, согласование и утверждение порядка и методики испытаний;

2. проведение испытаний;

3. корректировка программы и программной документации по результатам испытаний.

**2.7 Порядок контроля и приемки**

Сдача-приёмка работ производится поэтапно, в соответствии с рабочей программой и календарным планом. Сдача-приемка осуществляется комиссией, в состав которой входят представители ГАПОУ и работодатель. По результатам приемки подписывается акт приемочной комиссии.

В указанном подразделе, согласно таблице «Сроки выполнения работ» технического задания, будет описаны перечень участвующих организаций, место и сроки проведения работ, согласно п. 2.8 ГОСТ 34.602-89.

Порядок согласования и утверждения приемочной документации должен регламентироваться организационно-распорядительной документацией организации, принимающей участие в создании программного продукта. Согласно разделу «Приемка результатов разработки» ГОСТ 15.001-88 для согласования и утверждения приемочной документации создается приемочная комиссия (приказом).

Статус приемочной комиссии определяется Заказчиком до проведения испытаний.

**3 ПОЯСНИТЕЛЬНАЯ ЗАПИСКА К ПРОГРАММНОМУ ПРОДУКТУ**

**3.1 Декомпозиция поставленной задачи**

Для достижения цели работы была поставлена следующая задача:

Создать модуль “CodeDatabase” отвечающий за сохранение БД. Результат работы модуля - сохранение БД.

Поставленные задачи наглядно отображаются на диаграмме деятельности.

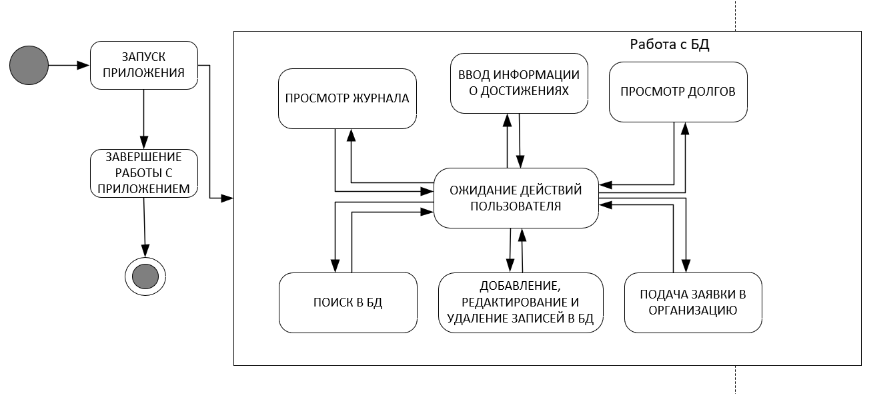


Рис.1 Декомпозиция поставленной задачи

**3.2 Общая архитектура программного средства**

Объектом моделирования является ПМ студента колледжа.

Рассматриваемые процессы: просмотр долгов, журнала вступление в организации.

Объекты моделирования представлены на диаграмме классов, рис.2.

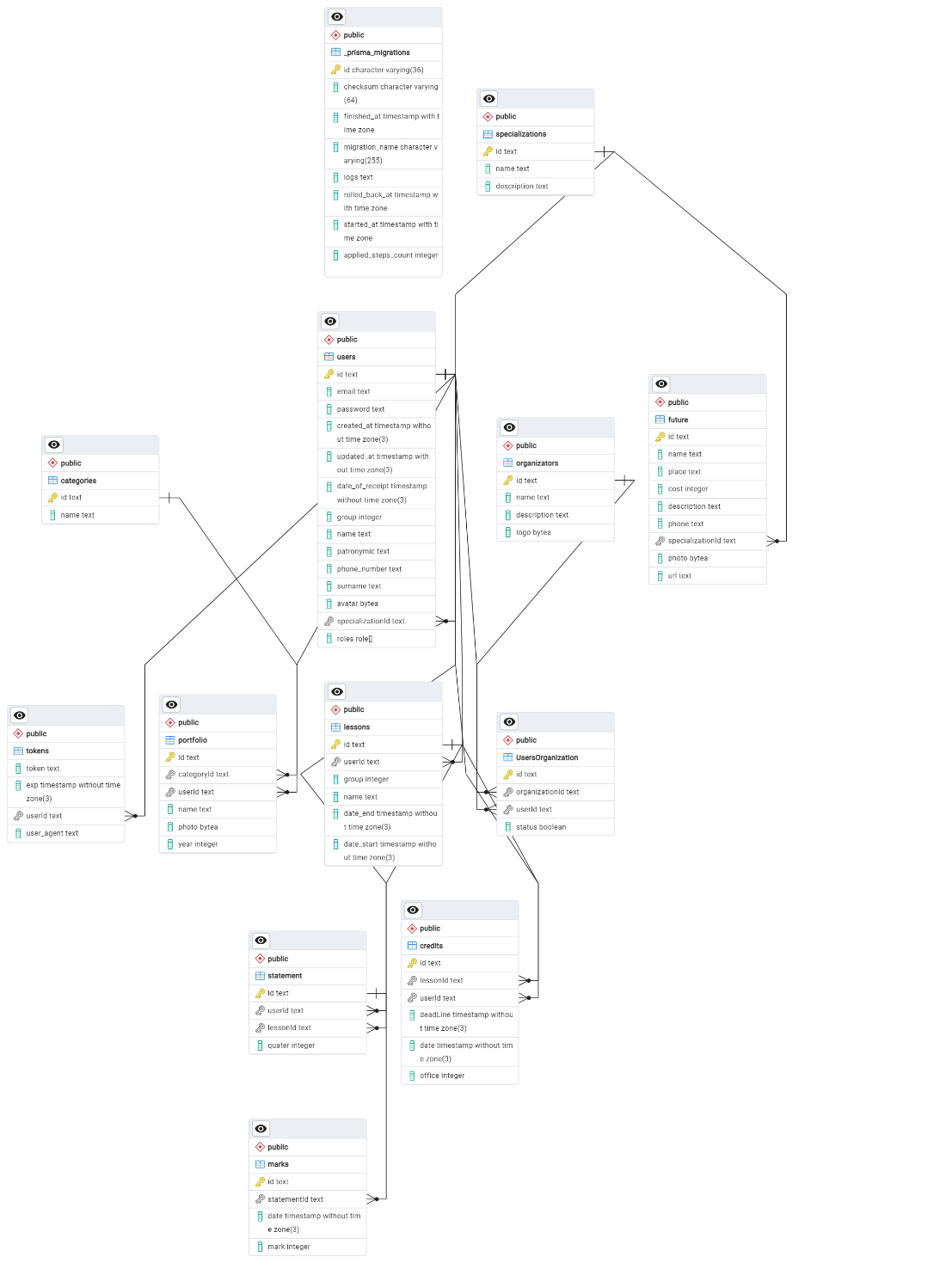


Рис.2 Диаграмма классов

По полученной декомпозиции задач была спроектирована следующая архитектура программного средства.



Рис.3 Схема событийно-ориентированной архитектуры

В «StudentCabinet 1.0» используется событийно-ориентированная архитектура.(Рис.3) Роль агента (источник событий) в программе выполняют: пользователи программы, за роль стоков (потребители событий) отвечают таблицы, входящие в базу данных. Например, когда пользователь выбирает определенное действие: добавить, удалить, редактировать и т.д. система осуществит выбранные действия, и база отреагирует соответствующим образом: запись добавлена, удалена, отредактирована. Системная архитектура пользователя рассматривает это изменение состояния как событие, создаваемое, публикуемое, определяемое и потребляемое различными приложениями в составе архитектуры.



Рис.4 Диаграмма вариантов использования

**3.3 Разработка алгоритма решения задачи**

Рис.5 Алгоритм программы

**3.4 Реализация функционального назначения программного средства**

Программа «StudentCabinet 1.01» имеет следующий набор входных данных, такие как: организаци, портфолио, уроки, специальности, оценки, долги, ведомость, категории.

Данные вводятся отделом кадров или пользователем в соответствующие поля ввода, снабженные всплывающими подсказками.

Выходными данными являются: ведомость, долги, организации и будущее.

Данные вводятся пользователем в соответствующие поля вывода, снабженные всплывающими подсказками.

Выходные данные редактировать вручную невозможно, т.к. они служат в качестве отчетной информации для пользователя.

**3.5 Структурная организация данных**

Для создания БД необходимо определиться с данными, которые необходимы для полноценного функционирования системы. Все эти данные указаны в реляционной модели представленной на рисунке. Любая реляционная база данных и называется реляционной, что характеризуется отношениями (relation) между таблицами. На рисунке изображены таблицы моей базы данных. При этом одна таблица является родительской (главной), а вторая – дочерней (подчиненной). Главной таблицей является «Ведомость». Реляционная модель автоматизированной системы соответствует всем 12 правилам Кодда. (рис.6)



Рис.6 Реляционная модель

Первичный ключ в базе уникален, используется для организации отношений между таблицами, который не может иметь пустых и повторяющихся значений. Первичными ключами в базе являются поля: «Код» (таблица «Оценки»), «Код» (Таблица «Ведомость»), «Код» (Таблица «Уроки»), «Код» (Таблица «Долги»), «Код» (Таблица «Организации\_пользователей»), «Код» (Таблица «Организации»), «Код» (Таблица «Пользователь»), «Код» (Таблица «Токен»), «Код» (Таблица «Специальность»), «Код» (Таблица «Категории»), «Код» (Таблица «Портфолио»), «Код» (Таблица «Будущее»).

Остальные ID-ключи являются внешними ключами.

Для организации более эффективной обработки данных применяется нормализация. Таблицы моей БД находятся в 3НФ:

* БД находится в форме -1НФ потому, что
  + Таблица не имеет повторяющихся записей;
  + Каждый атрибут отношения хранит одно-единственное значение и не является списком, ни множеством значений;
  + Таблица не имеет повторяющихся групп полей.
    - Вторая нормальная форма(2НФ):
      * Устранены атрибуты, зависящие только от части уникального (первичного) идентификатора, т.е. ID.
        + Третья нормальная форма(3НФ):

Отсутствуют атрибуты, зависящие от атрибутов, не входящих в уникальный (первичный) ключ.

На основе реляционной модели базы данных мною в PgAdmin была построена следующая база данных. (Рис.7) В ней первичным ключом является поле таблицы “User” – “id”. Связь таблиц “User” и “Token” по полю “userId” один ко многим (1:М). Связь таблиц “User” и “Portfolio” по полю “userId” один ко многим (1:М). “User” и “Lesson” по полю “userId” один ко многим (1:М). “User” и “Statement” по полю “userId” один ко многим (1:М).”User” и “Credit” по полю “userId” один ко многим (1:М). “User” и “UsersOrganization” по полю “userId” один ко многим (1:М). Связь таблиц “Organizations” и “UsersOrganization” по полю “organizationId” один ко многим (1:М). Связь таблиц “Categories” и “Portfolio” по полю “categoryId” один ко многим (1:М). Связь таблиц “Lessons” и “Statement” по полю “lessonId” один ко многим (1:М). Связь таблиц “Lessons” и “Credit” по полю “lessonId” один ко многим (1:М). Связь таблиц “Statement” и “Marks” по полю “statementId” один ко многим (1:М). Связь таблиц “Specializations” и “Future” по полю “specializationId” один ко многим (1:М). Связь таблиц “Specializations” и “User” по полю “specializationId” один ко многим (1:М). Все связи между таблицами базы данных типа один ко многим.

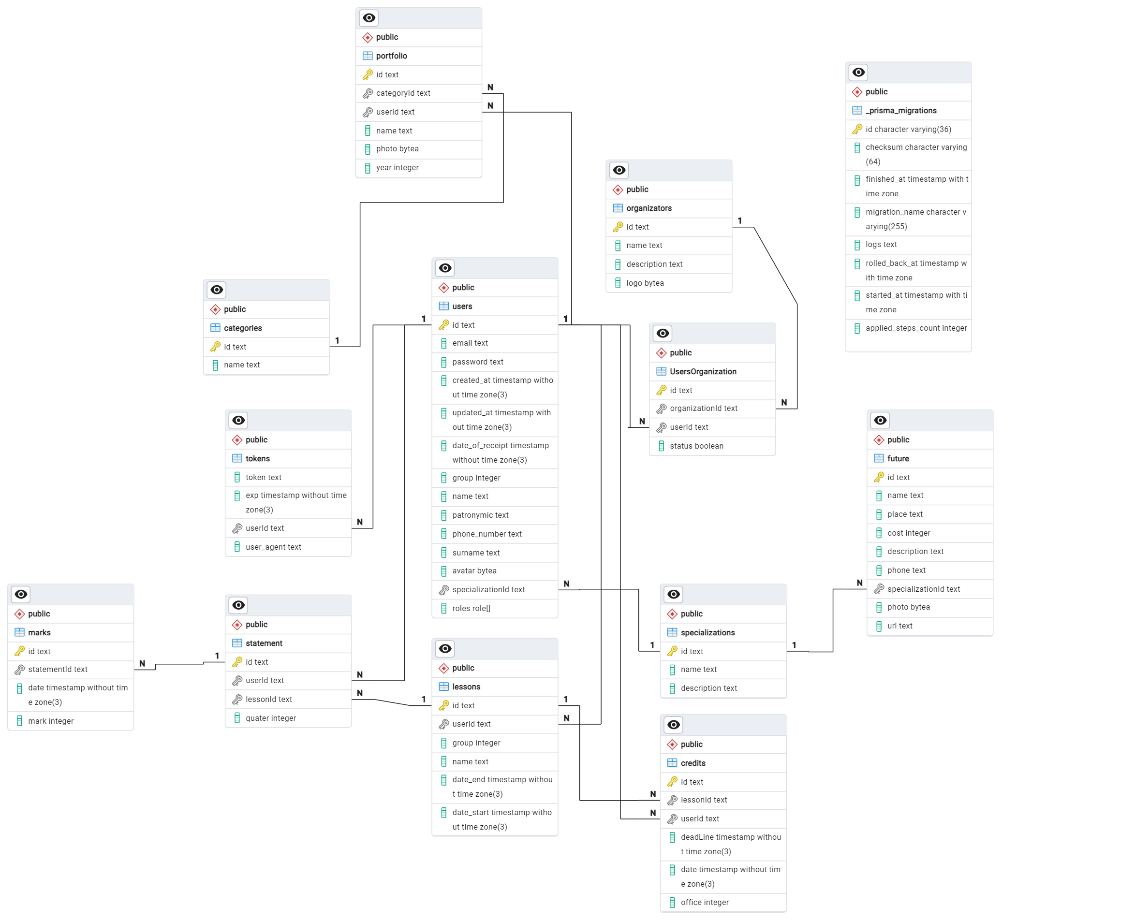


Рис.7 Схема базы данных

Таблица “credits”(Долги)

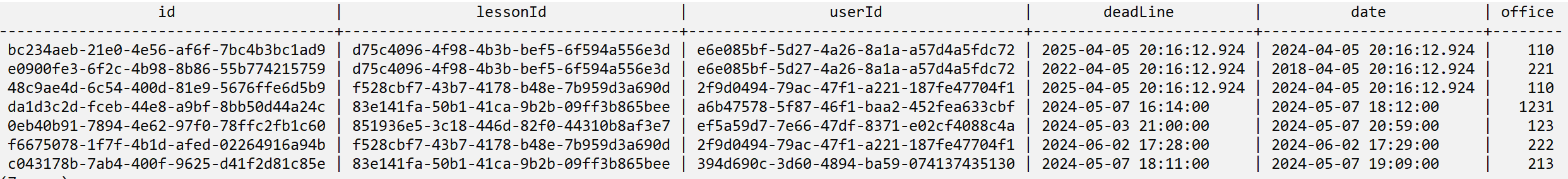


Рис.8 Таблица credits

Таблица “tokens”(Токены)

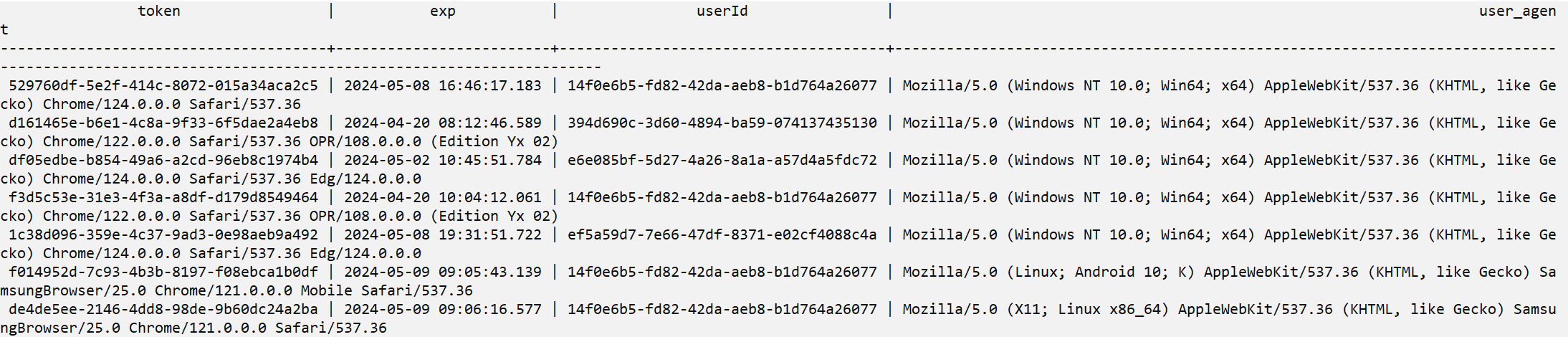


Рис.9 Таблица tokens

Таблица “users”(Пользователи )



Рис.10 Таблица users

Таблица “statement”(Ведомость)

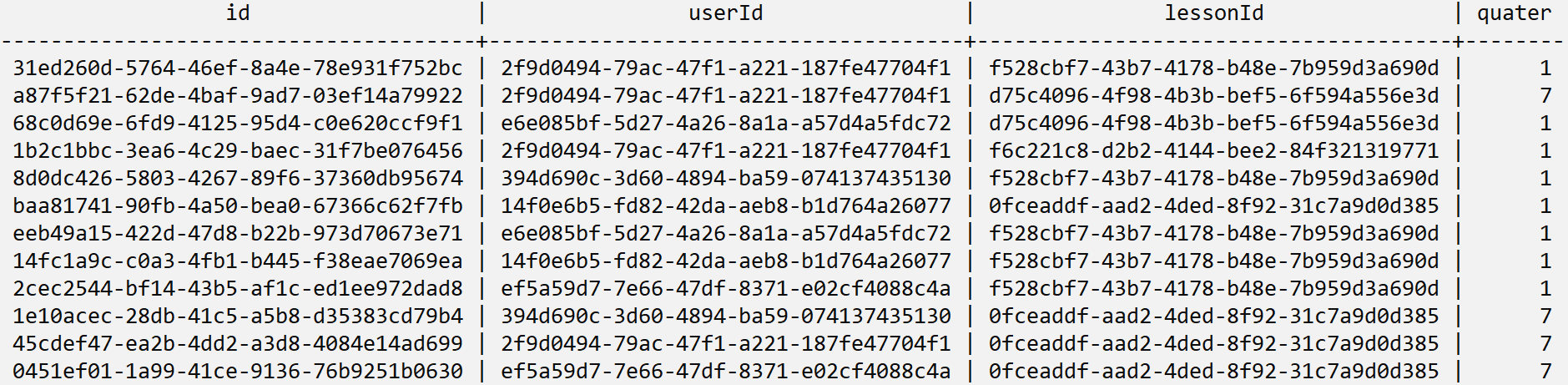


Рис.11 Таблица statement

Таблица “specializations”(Специальности)

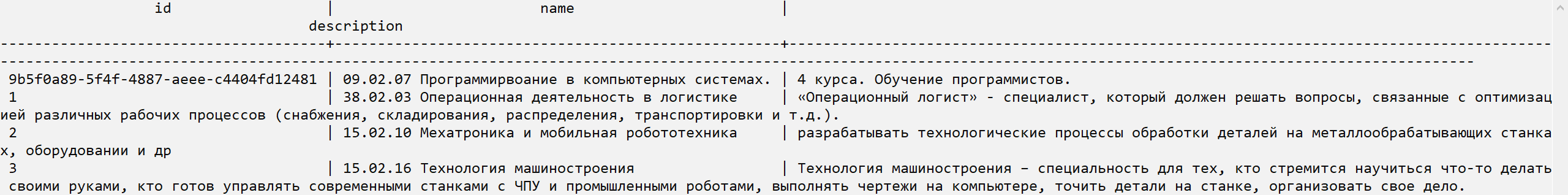


Рис.12 Таблица specializations

Таблица “portfolio”(Портфолио)

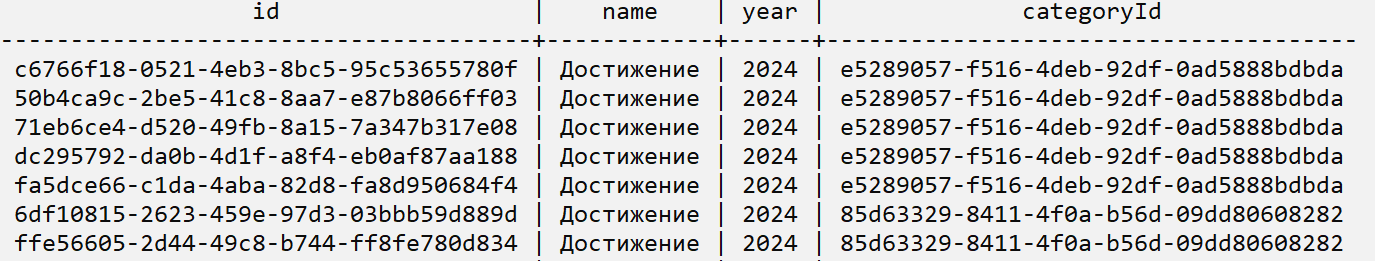


Рис.13 Таблица portfolio

Таблица “organizators”(Организации)

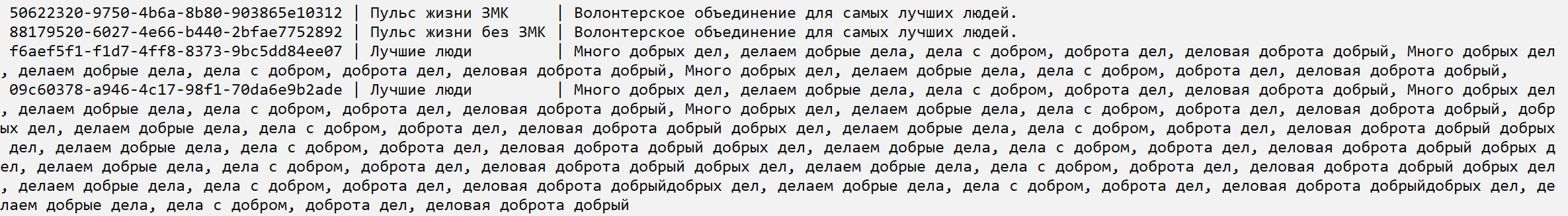


Рис.14 Таблица organizators

Таблица “marks”(Оценки)

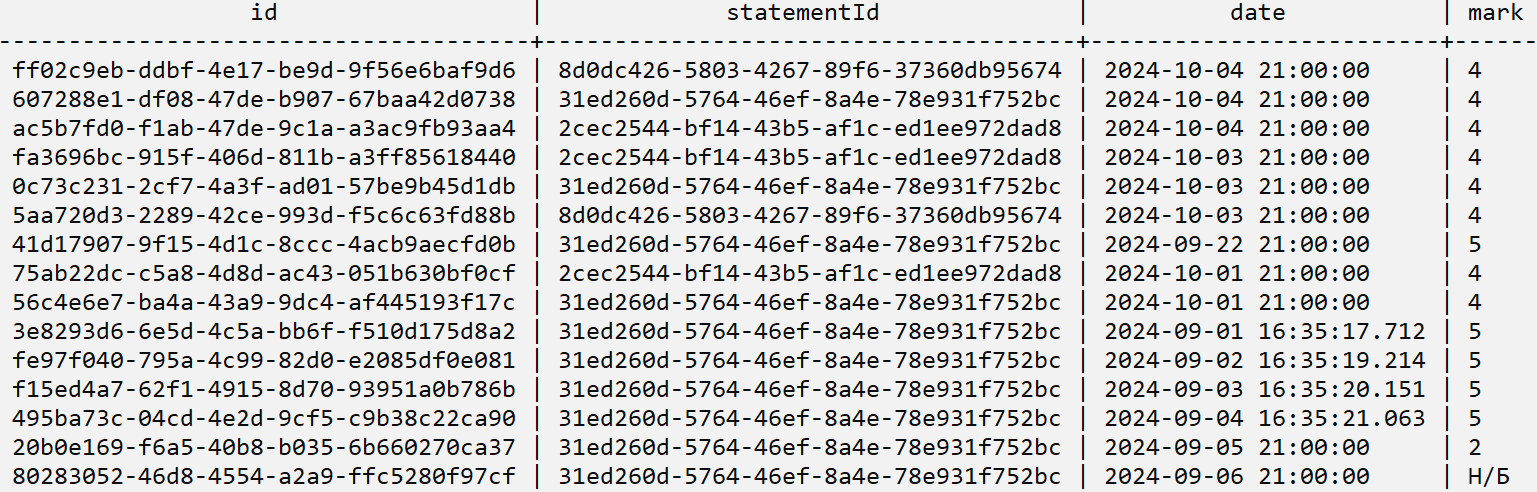


Рис.15 Таблица marks

Таблица “lessons”(Уроки)

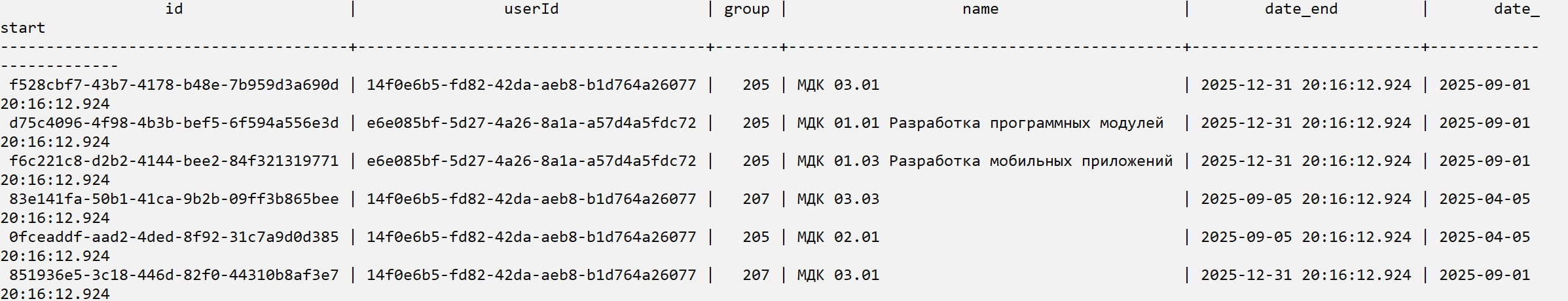


Рис.16 Таблица lessons

Таблица “future”(Будущее)

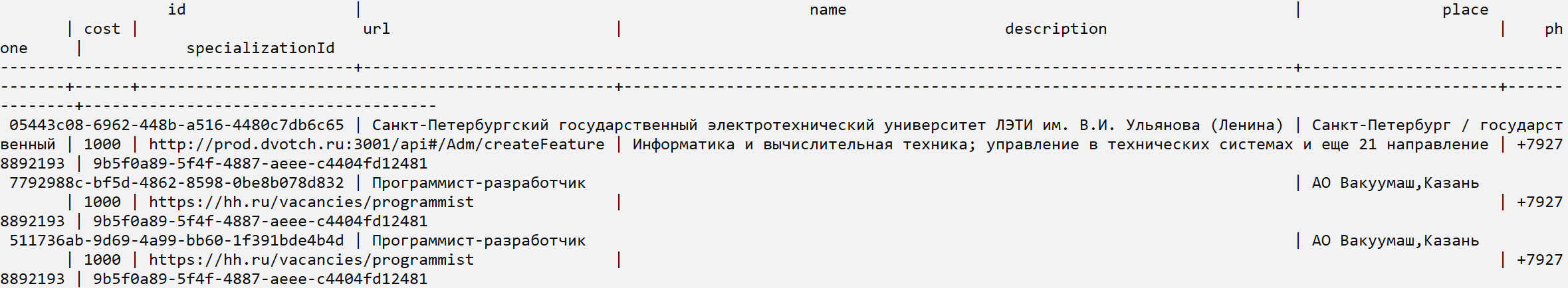


Рис.17 Таблица future

Таблица “categories”(Категории)

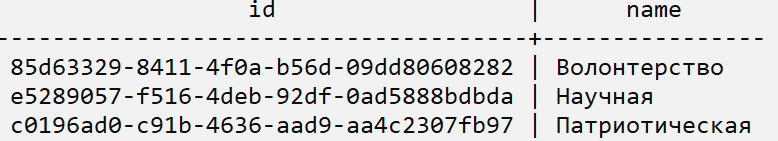


Рис.18 Таблица categories

**3.6 Разработка интерфейса ПС**

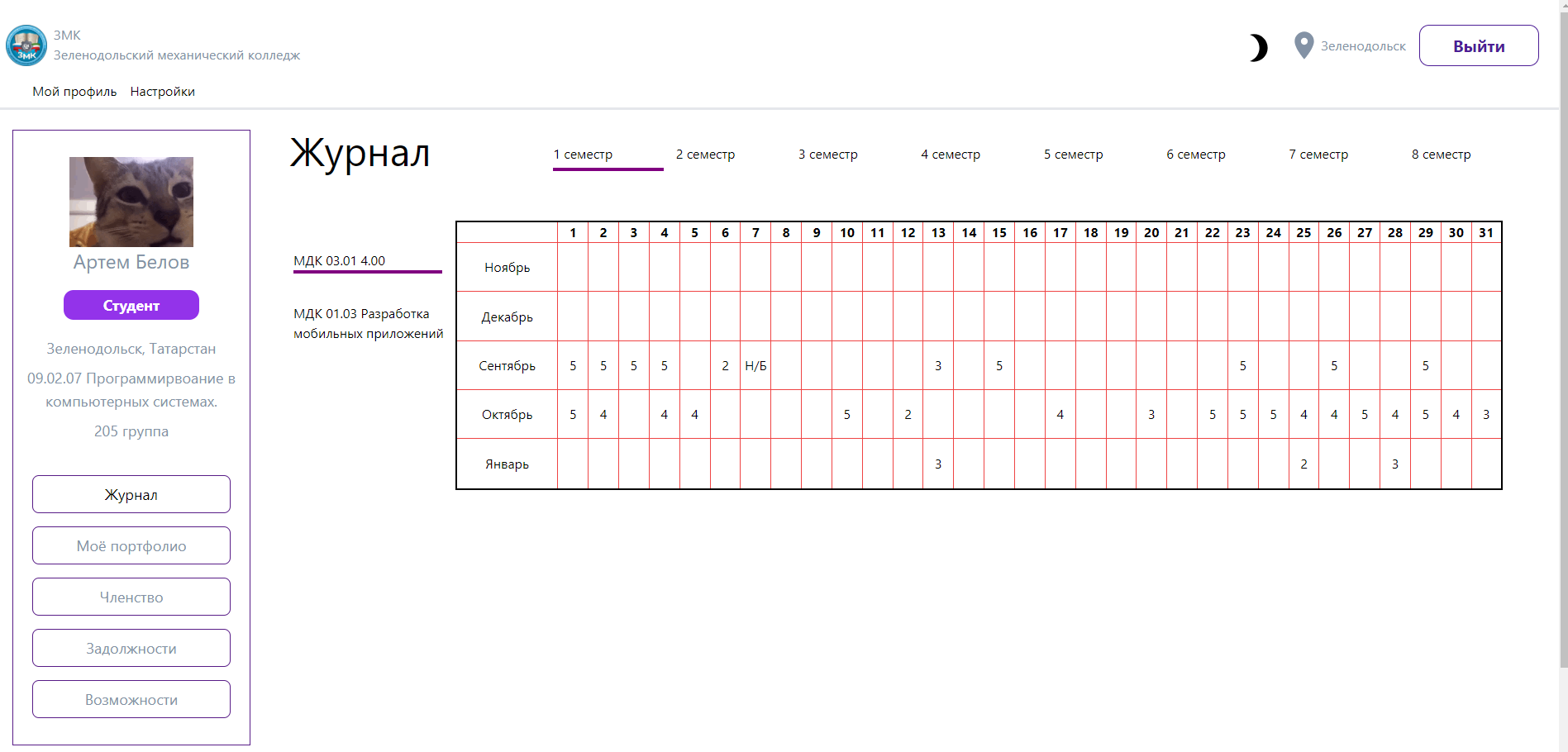
Приложение предназначено для двух типов пользователей: отдел кадров и студент. Для входа в панель отдела кадров необходимо ввести свои учётные данные. Для запуска приложения достаточно щелкнуть дважды левой кнопкой мыши по ярлыку.

После запуска программы пользователь приступит к работе с главной формой программы, на которой отбражается Журнал в виде табличного интерфейса. «StudentCabinet 1.01» обладает удобным интерфейсом, так что даже неопытному в обращение с пользовательскими компьютерными программами пользователю не составит труда быстро привыкнуть к работе с данным программным средством.

На рабочем экране расположены несколько кнопок для комфортной работы с приложением.

**Страница «Журнал»**. Основная форма приложения (Рис.19). При разработке интерфейса этой формы было использовано множество различныx компонентов:

1. Table – таблица для отображения оценок;
2. Buttons – кнопки для перехода на другие формы;
3. Img – фотография профиля.



Button

Img

Table

Button

Рис.19 Страница «Журнал»

**Страница входа в систему**. Интерфейс данной страницы состоит из компонентов: Button – кнопка Войти, Регистрация Input – поле для ввода логина и ввода пароля, h1 – для заголовков на странице(Рис.20).

Button

h1

Input

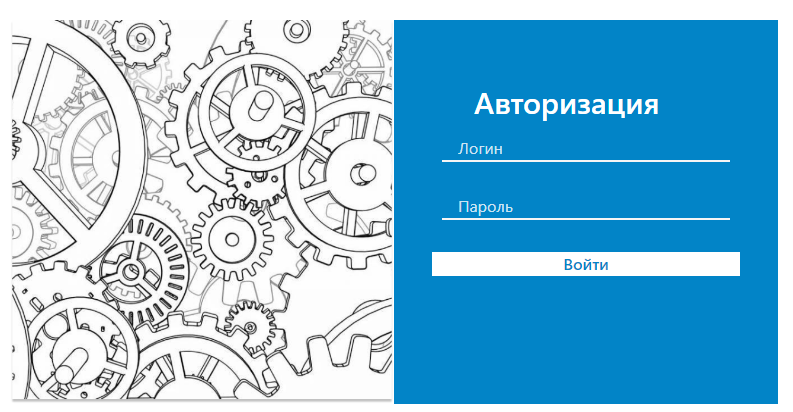
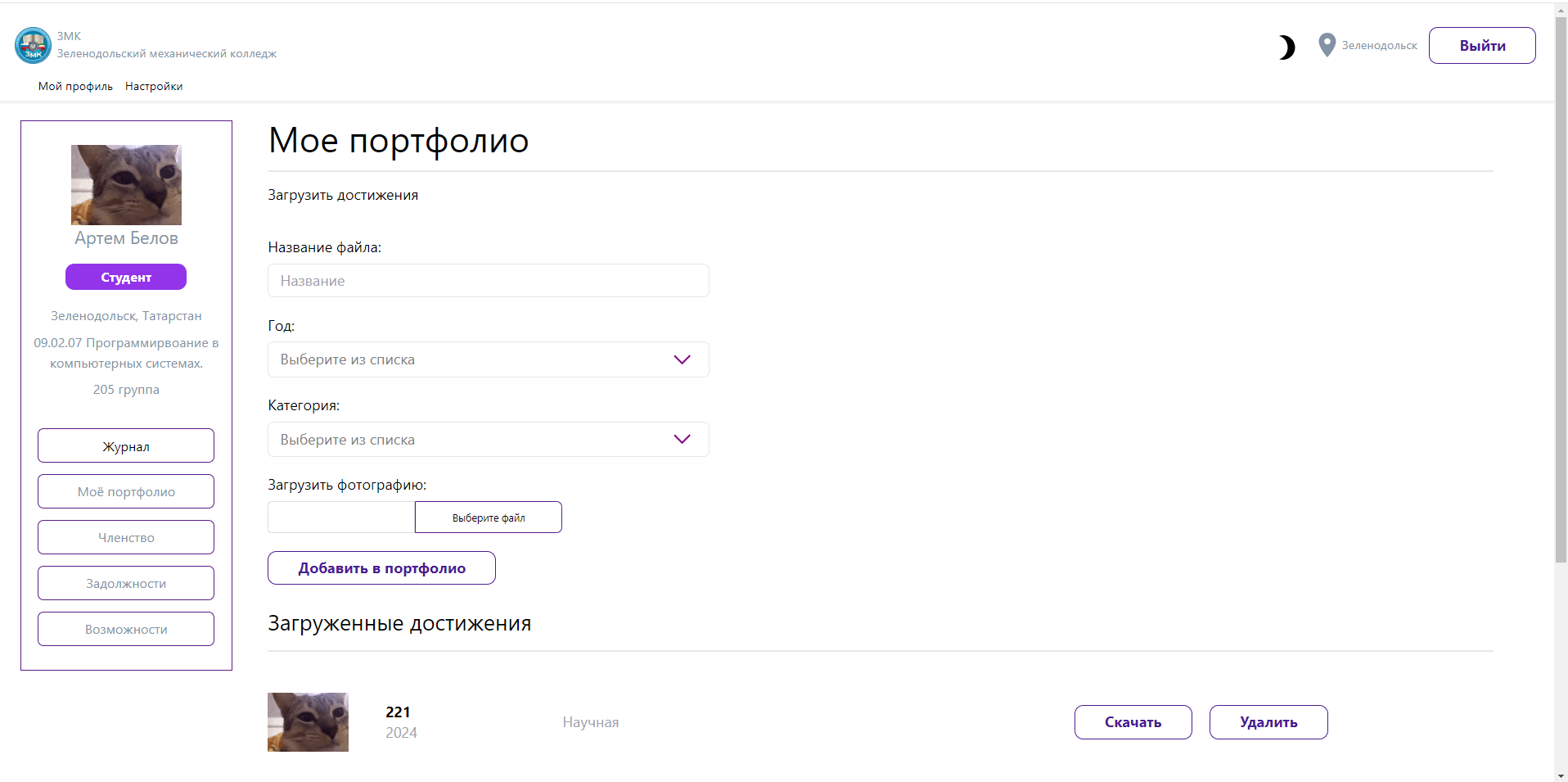


Рис.20 Страница входа в систему

**Страница «Мое портфолио»(**Рис.21**)**. При разработке интерфейса этой страницы были использованы компоненты:

1. Img – для отображения изображения;
2. Buttons – кнопки для отправки, удаления, выбора, скачивания;
3. Input – поля ввода данных;
4. Select – для выбора среди набора данных



Button

Img

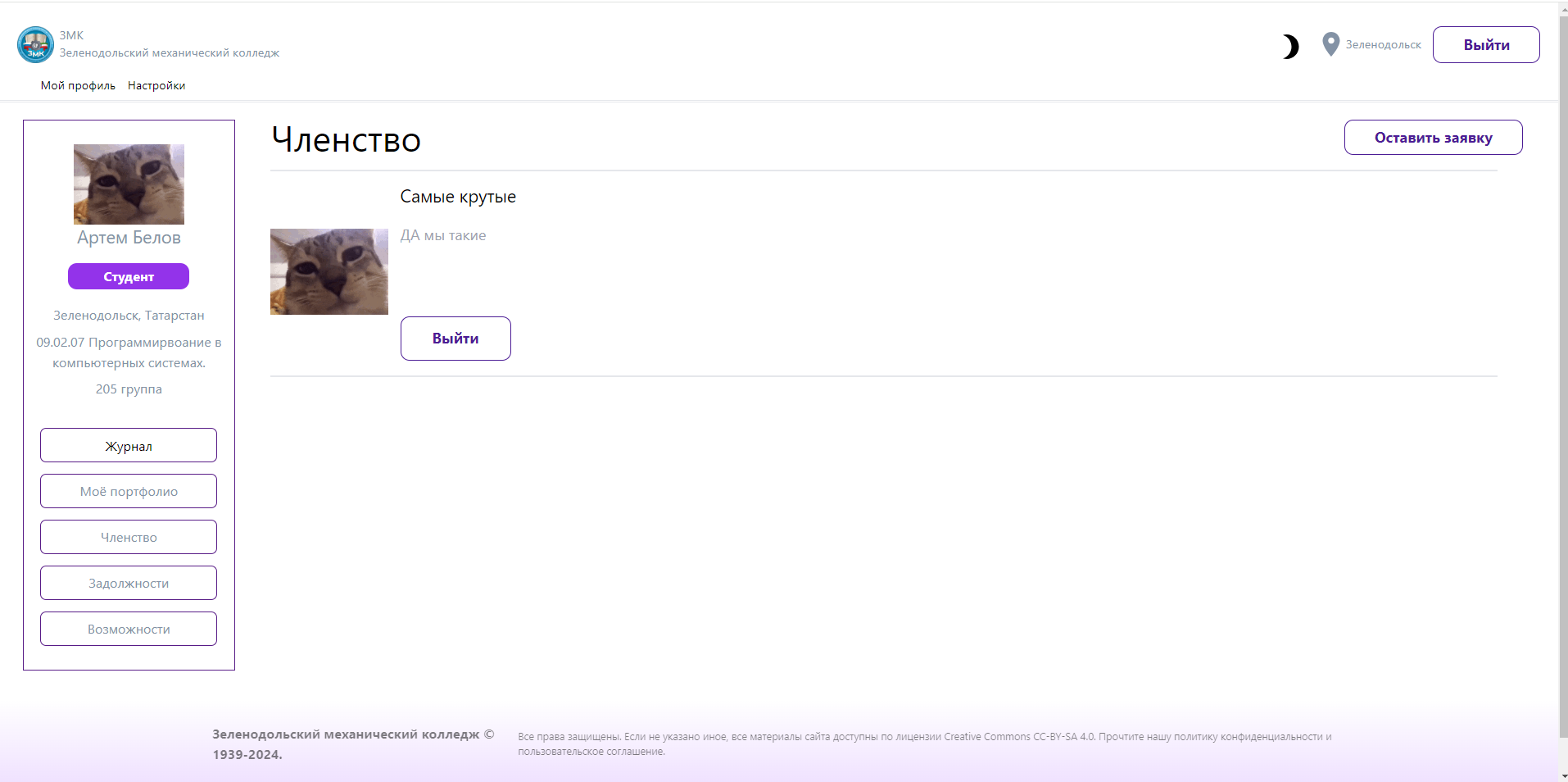
Input

Select

Рис.21 Страница «Мое портфолио»

**Страница «Членство» (**Рис.22**)**. При разработке интерфейса этой страницы были использованы компоненты:

1. Img – для отображения изображений;
2. Buttons – кнопки отправки заявки и выхода из организации;
3. H1 –заголовок первого уровня;
4. H2 – заголовок второго уровня;
5. P – для отображения текста.



P

Button

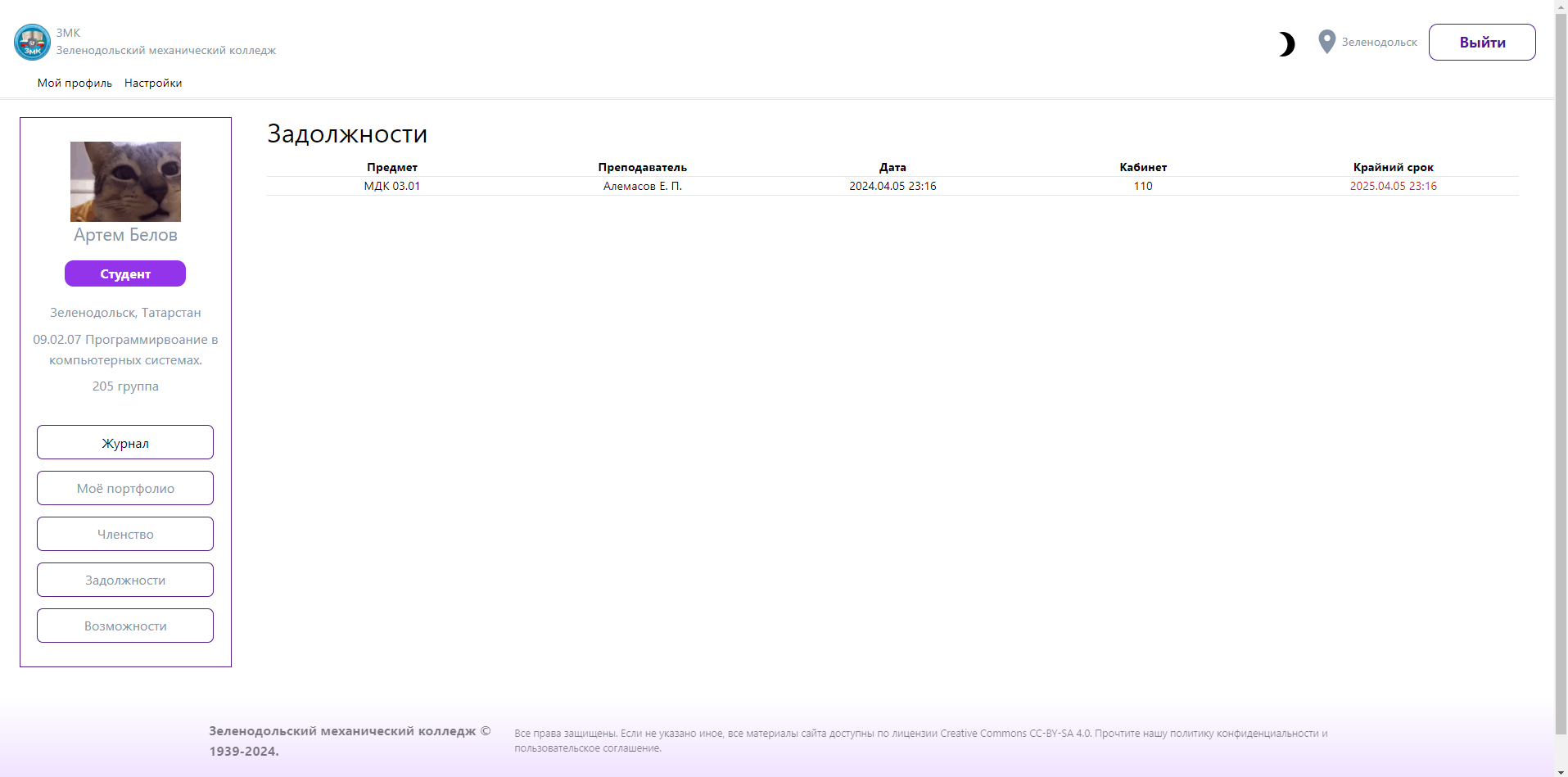
H2

H1

Img

Рис.22 Страница «Членство»

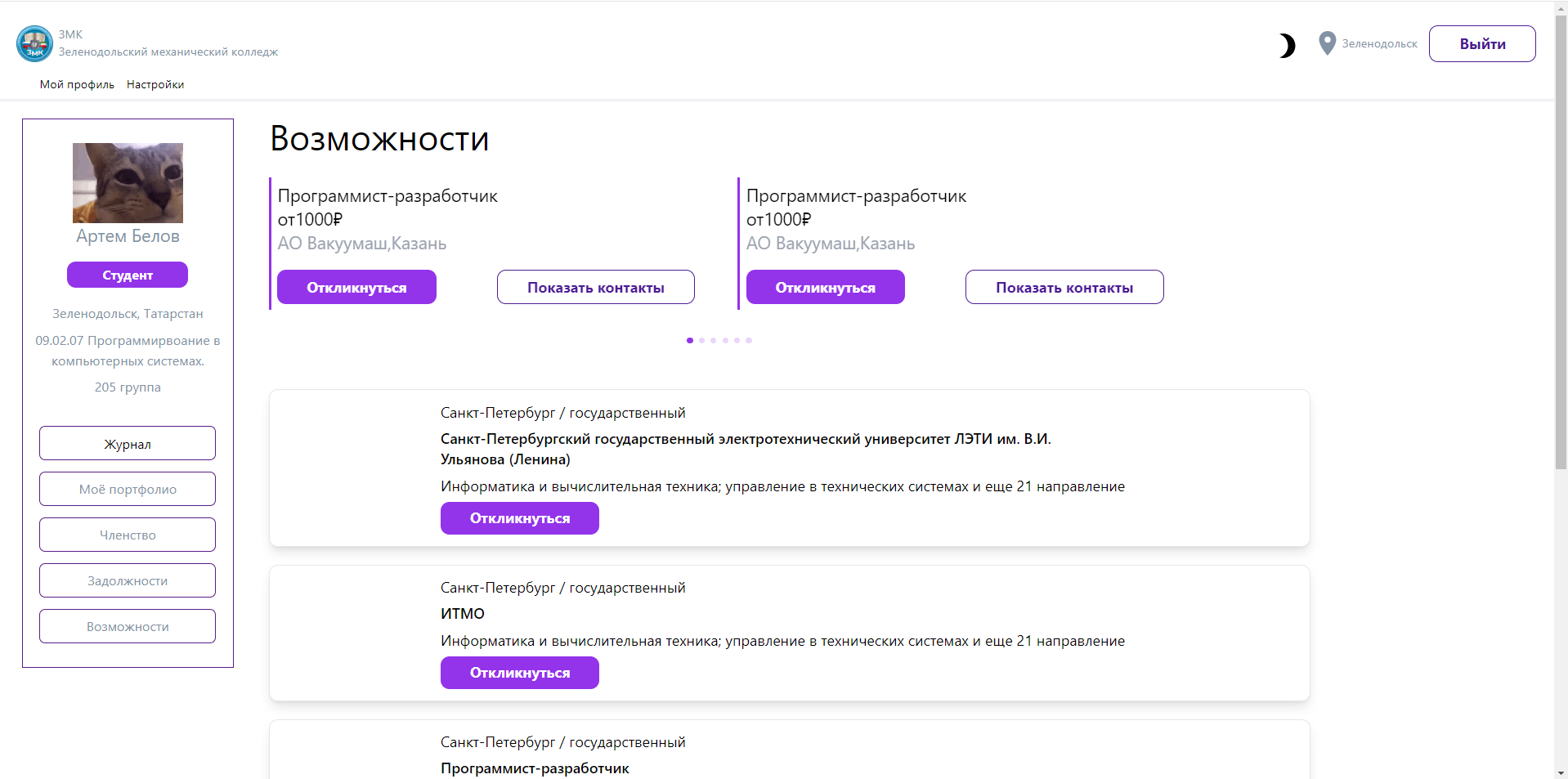
**Страница «Задолженности»**. Интерфейс данной страницы состоит из компонентов: H1 и Table(Рис.23)



H1

Table

Рис.23 Страница «Задолженности»

**Страница «Возможности»**. Интерфейс данной страницы состоит из таких компонентов, как Swiper – слайдер для отображения данных, p – для текста и кнопоки Button. (Рис.24) 

Button

P

Swiper

Button

Рис.24 Страница «Возможности»

**3.7 Описание структуры выходной информации**

Выходными данными являются журнал, долги, членство, достежения и будущее.

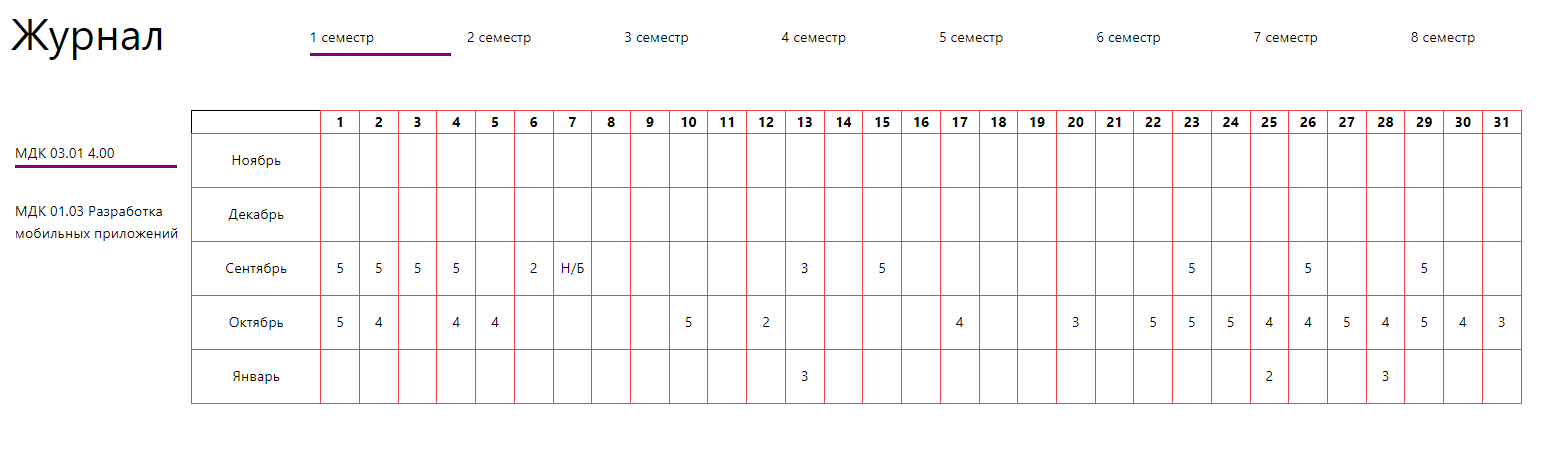


Рис.25 Журнал

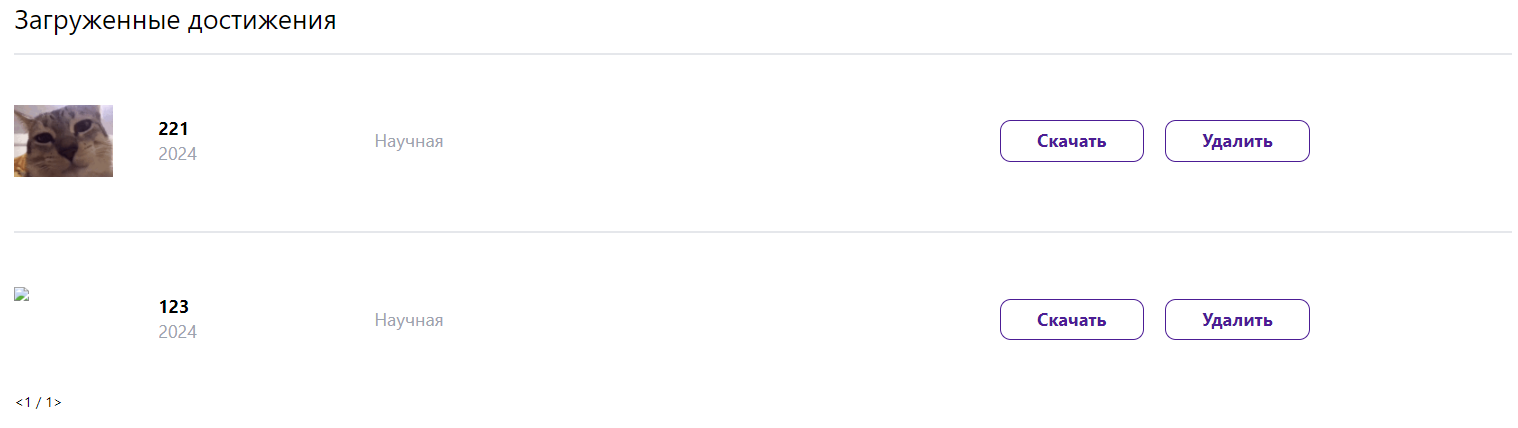


Рис.26 Достижения

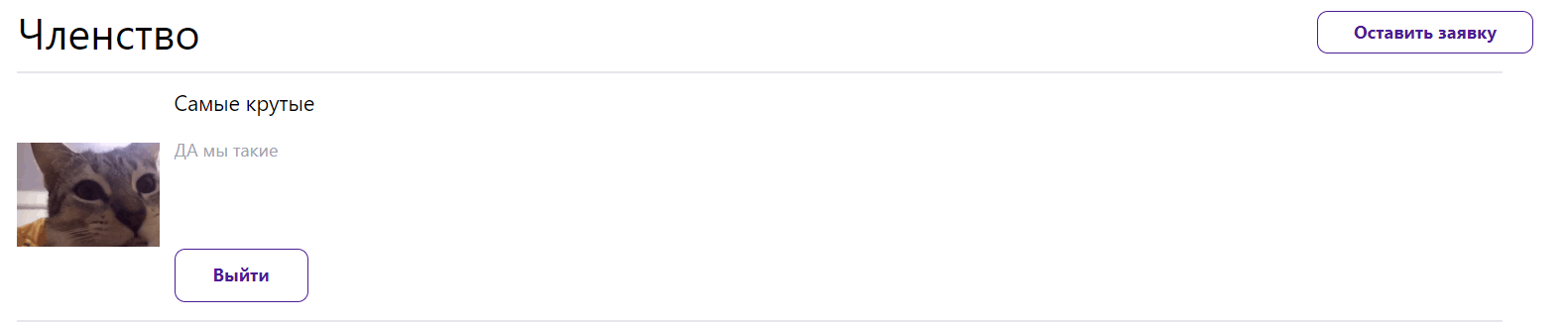


Рис.27 Членство

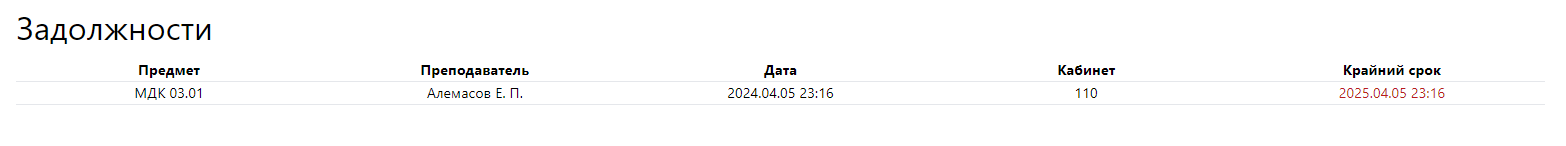


Рис.28 Задолжности

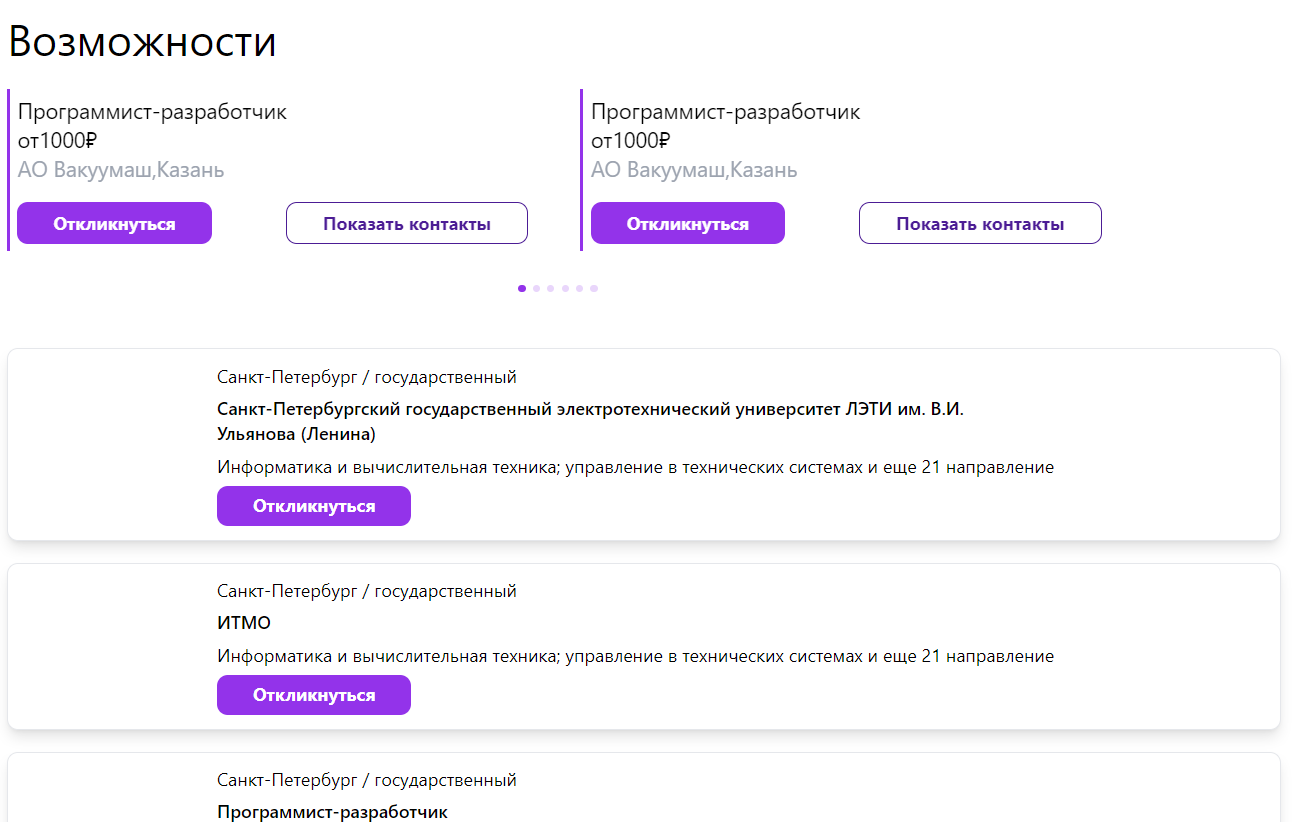


Рис.29 Возможности

**4 РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ**

**4.1 Назначение программного средства**

Назначением программного средства является автоматизация деятельности колледжа.

**4.2 Условия выполнения программного средства**

Условия, при соблюдении которых возможно выполнение работы с программным продуктом:

-Компьютер имеет доступ в интернет.

Подготовительные действия:

-Произвести переход по ссылке <http://prod.dvotch.ru/Login>.

**4.3 Эксплуатация программного средства**

Этот раздел содержит обзор приложений «StudentCabinet 1.0» и ее основных параметров.

Активировать копию программы StudentCabinet 1.0 можно сразу после её установки, когда появится окно подтверждения подлинности Вашей версии. Ключ активации расположен внутри упаковки программного продукта. Для успешного выполнения активации ключ активации необходимо вводить именно в том виде, в котором он предоставлен.

Вводные сведения об интерфейсе пользователя

После запуска приложения отображается стартовое окно приложения (Рис.30). Первая страница представляет собой окно авторизации. После ввода логина и пароля пользователя, его переадресует на соответствующую его роли страницу.

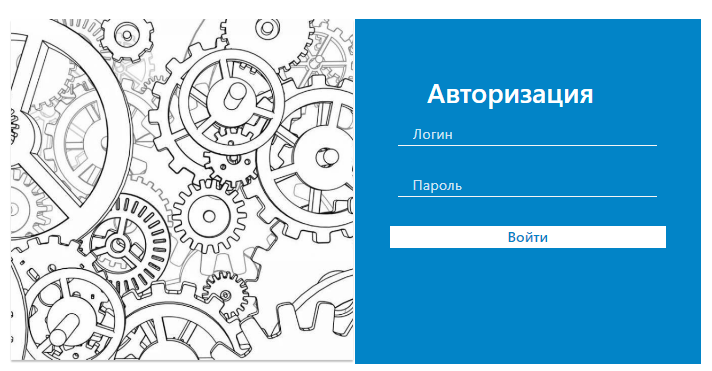


Рис.30 Страница Авторизации

После успешного входа нас встречает Главная страница студента. С этого момента нам становятся доступны такие страницы, как журнал, задолжности, организации, портфолио и возможности (Рис. 31).



Рис.31 Главная страница

Переход по кнопке «Журнал» откроет страницу в которой можно будет выбрать семестр и предмет для просмотра своих оценок(Рис. 32).

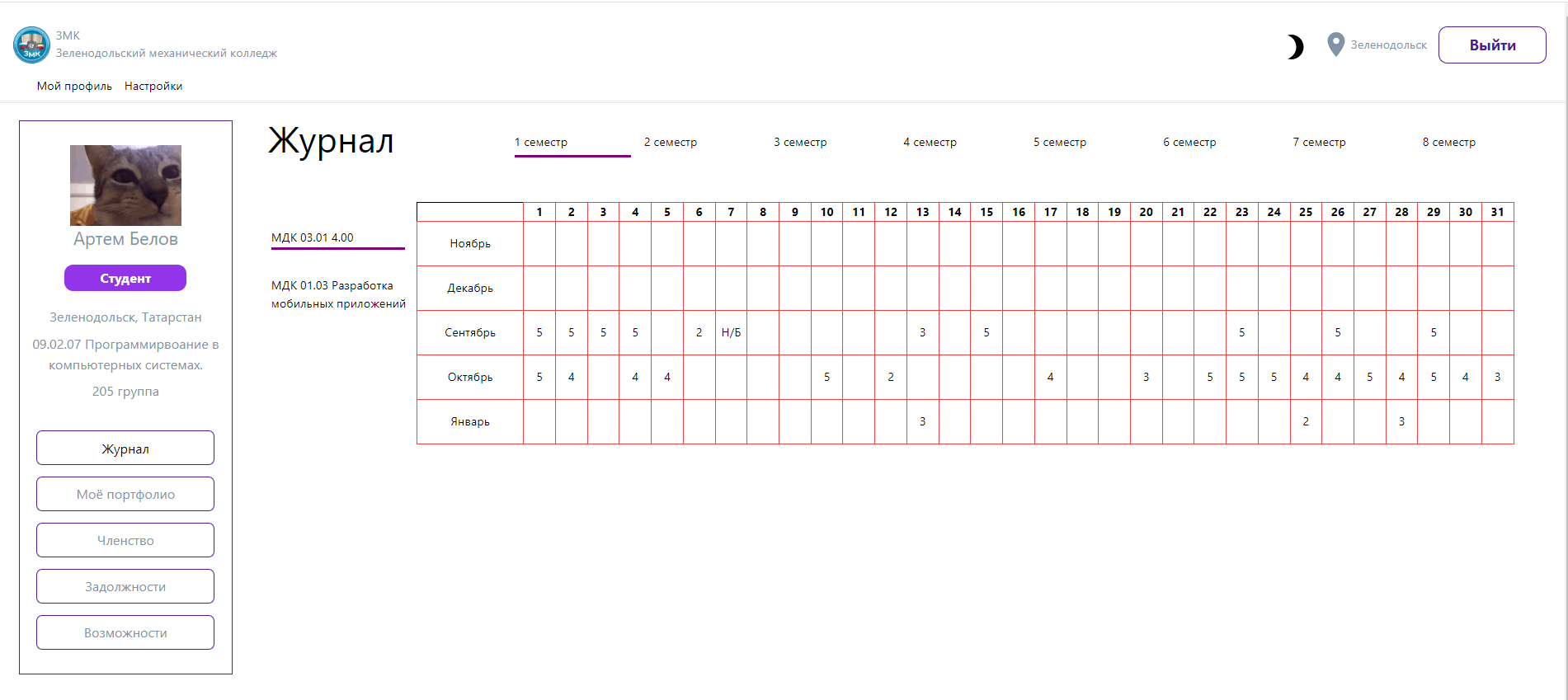


Рис.32 Форма регистрации

Страница «Мое портфолио» предназначена для добавления новых достижений или просмотра уже добавленных, а так же имеется возможность скачивания этих достижений(Рис.33).

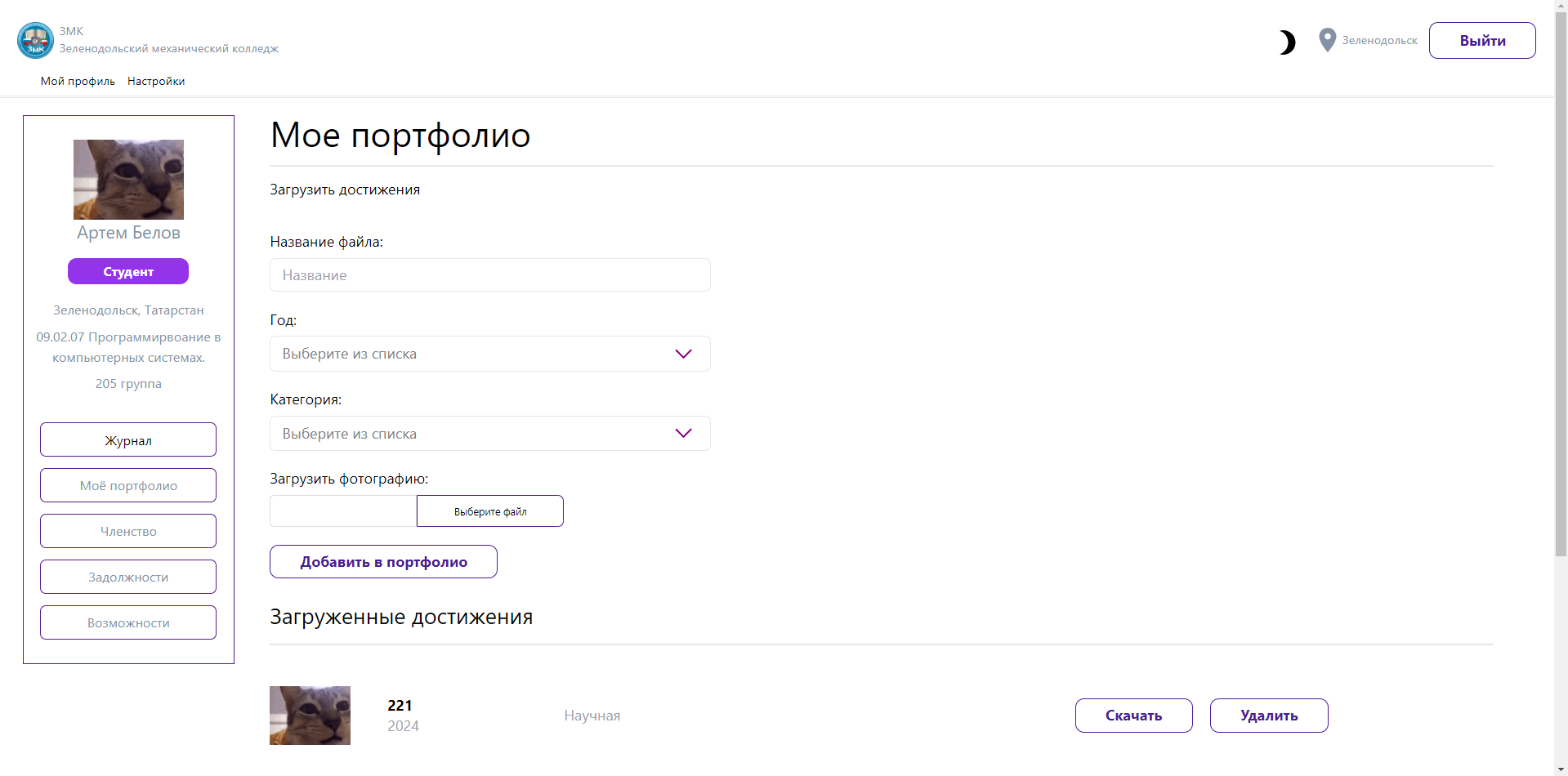


Рис.33 Страница Мое портфолио

Активности в колледже являются его неотъемлемой частью и именно для таких случаев имеется страница «Членство». В данной вкладке можно отправлять заявки на вступление в новые или для просмотра организаций в которые студент уже вступил(Рис.34).

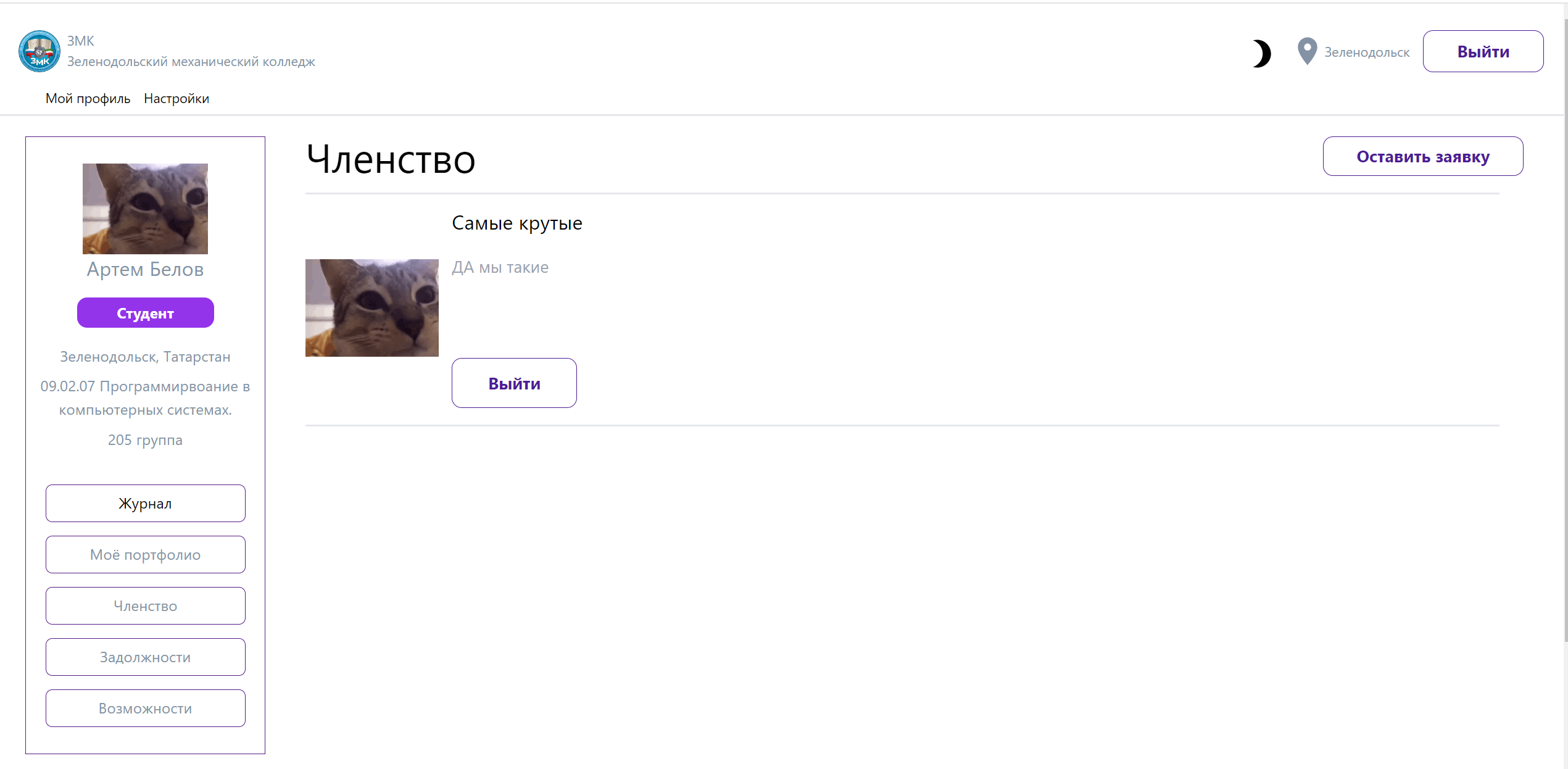


Рис.34 Страница Членство

Для отправки зявки следует начать на кнопку «Оставить заявку» и откроется форма, которую нужно будет заполнить(Рис.35).

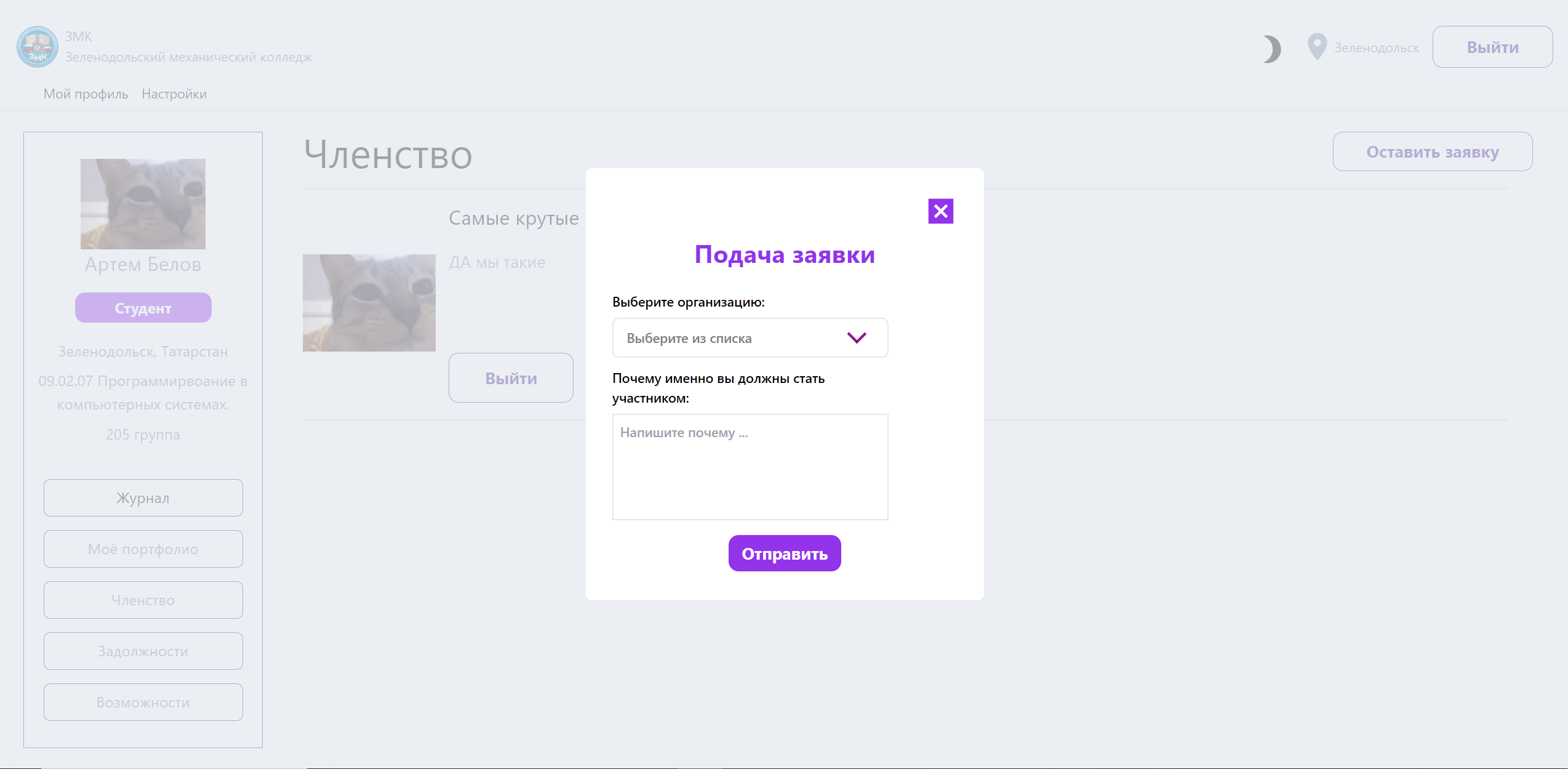


Рис.35 Форма Подача заявки

Страница «Задолжности» используется для просмотра своих долгов по разным предметам с датой их сдачи(Рис.36).

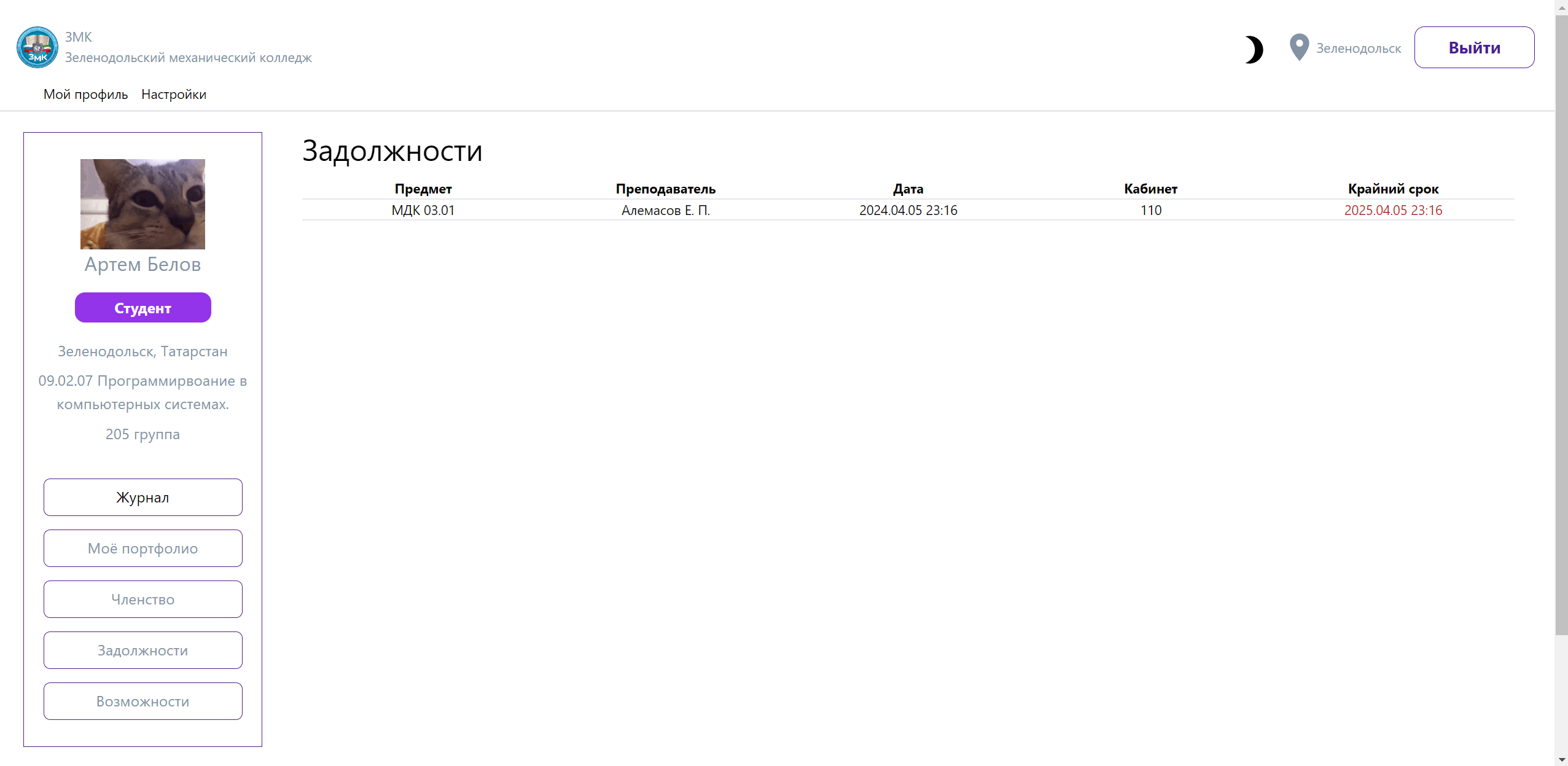


Рис.36 Страница Задолжности

Страница «Возможности» предназначена для помощи выпускникам с трудоустройством или поиском следующего образовательного учреждения(Рис.37).

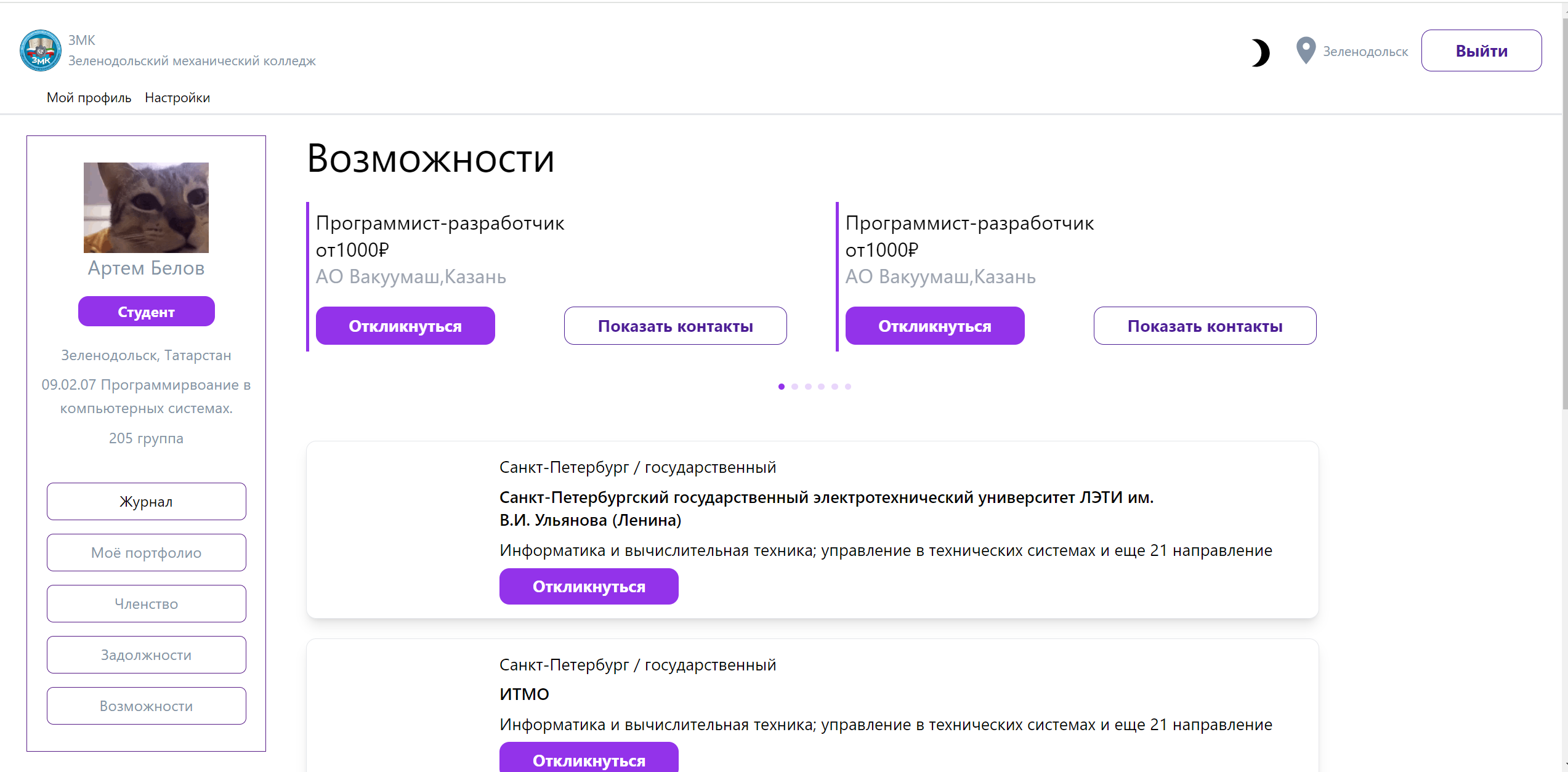


Рис.37 Страница Возможности

Для выхода из системы используется кнопка «Выйти» после нажатия на которую нас перекинет обратно на страницу логина(Рис. 38).

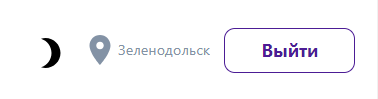


Рис.38 Кнопка выхода

Для переключения темы используется отедльная кнопка, находящаяся в правом верхнем углу 

Работа с StudentCabinet 1.01

После запуска программы Вы перейдете на страницу авторизации. После успешного прохождения авториации вам откроется доступ к продукту. В формах «Достижения» и «Членство» заполняем все необходимые для работы пункты.

После этого кликаем по кнопке «Отправить» или «Добавить в протфолио».

Если Вы желаете удалить любую введенную ранее информацию, Вам необходимо кликнуть по нужной кнопке.

После того, как Вы закончите сеанс работы с программой, если Вы хотите выйти из StudentCabinet 1.01 кликните на соответствующей кнопке «Выйти».

Основные кнопки StudentCabinet 1.01

Кнопки управления «Добавить в портфолио», «Выйти», «Оставить заявку», «Отправить», «Скачать» и «Удалить», которые распологаются на страницах приложения.

Действия этих кнопок приведены в таблице 1:

Таблица 1 – Действие кнопок

|  |  |
| --- | --- |
| Добавить в портфолио | После нажатия на эту кнопку, если все данные введены корректно ваше достижение добавится в базу. |
| Скачать | Кнопка «Скачать» позволяет скачивать фотографии достижений, которые вы уже добавили. |
| Удалить | Кнопка «Удалить» предназначена для удаления записи о вашем достижении из базы. |
| Выйти | Позволяет выйти из организации в которую вы вступили. |
| Оставить заявку | Открытие формы для отправки заявки. |
| Отправить | Отправка заявления на вступление в организацию. |
| Журнал | Перейти на страницу Журнал. |
| Мое портфолио | Перейти на страницу Мое портфолио. |
| Членство | Перейти на страницу Членство. |
| Задолжности | Перейти на страницу Задолжности. |
| Возможности | Перейти на страницу Возможности. |
| Тема | Исользуется для переключения темы с темной на светлую и обратно. |

**4.4 Сообщения пользователю**

Для уверенной и комфортной работы пользователю важно понимать поведение программы, а программе, в свою очередь, поведение пользователя. Для информирования и уточнения намерений пользователей в программах используются сообщения.

При попытке добавления записи пользователь должен корректно заполнить все поля, иначе выводится данное сообщение. (Рис.38)

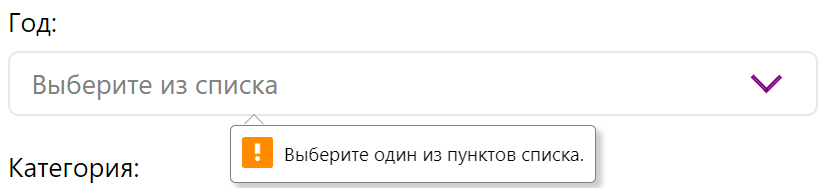


Рис.38 Сообщение Обязательное поле

После добавления новой записи, пользователю выводится сообщение. (Рис.39)



Рис.39 Сообщение Запись добавлена

После обновления записи, пользователю выводится сообщение. (Рис.40)



Рис.40 Сообщение Запись обновлена

**5 РАЗРАБОТКА ТЕСТОВОГО НАБОРА**

**5.1 Определение верхней границы количества тестов**

Для разработанного алгоритма был построен граф.



Рис.41 Потоковый граф

Необходимое количество тестов определяется по формуле V(G) = E-N+2, где E – это число ребер, а N–это число вершин данного графа. Согласно представленному графу E=25, N=19. Далее по формуле рассчитаем необходимое количество тестовых сценариев (тест-кейсов).

V(G)=25-19+2=8.

Для данного программного средства необходимо разработать не менее 8 тестовых сценариев.

Чтобы определить количество тестовых прогонов для программного продукта StudentCabinet 1.0 необходимо выделить требования. Из требований к приложению выделим поддержку 2-х операционных систем с одним основным языком локализации и выполнения 6-ти основных функций. Кроме того, приложение позволяет выполнять 5 функций по взаимодействию с системой (запуск приложения, выход из приложения, сохранение результатов в файл, работа с буфером и т.п.).

Таким образом, полное покрытие требований задаёт набор из 2\*1\*(6+5) =22 тестовых прогонов.

**5.2 Разработка тестовых вариантов**

Таблица 2 – Тестирование корректного добавления информации в Портфолио

|  |  |  |  |
| --- | --- | --- | --- |
| **Название:** | Тестирование корректного добавления информации в таблицу Портфолио | | |
| **Функция:** | Циклы | | |
| **Действие** | | **Ожидаемый результат** | **Результат теста:**   * пройден * провален * заблокирован |
| **Предусловие:** | |  |  |
| Открываем форму Портфолио | | Форма открыта |  |
| **Шаги теста:** | |  |  |
| Заполните форму  Название = Диплом за труды  Год = 2024  Категория = Волонтерство  Фотография = photo.png | | Данные успешно введены |  |
| Нажмите кнопку «Добавить в портфолио» | | Данные успешно сохранены | **Тест пройден** |
| **Постусловие:** | |  |  |
| Форма Портфолио открыта | | Отображается форма Портфолио |  |

Таблица 3 – Тестирование некорректного добавления информации в таблицу Портфолио

|  |  |  |  |
| --- | --- | --- | --- |
| **Название:** | Тестирование некорректного добавления информации в таблицу Портфолио | | |
| **Функция:** | Циклы | | |
| **Действие** | | **Ожидаемый результат** | **Результат теста:**   * пройден * провален * заблокирован |
| **Предусловие:** | |  |  |
| Открываем форму Портфолио | | Форма открыта |  |
| **Шаги теста:** | |  |  |
| Заполните форму  Название =  Год =  Категория =  Фотография = | | Данные успешно введены |  |
| Нажмите кнопку «Добавить в портфолио» | | Кнопка «Добавить в портфолио» заблокирована, так как поле пустое | **Тест пройден** |
| **Постусловие:** | |  |  |
| Форма Портфолио открыта | | Отображается форма Портфолио |  |

Таблица 4 – Тестирование корректного редактирования информации в таблице Портфолио

|  |  |  |  |
| --- | --- | --- | --- |
| **Название:** | Тестирование корректного редактирования информации в таблице Портфолио | | |
| **Функция:** | Циклы | | |
| **Действие** | | **Ожидаемый результат** | **Результат теста:**   * пройден * провален * заблокирован |
| **Предусловие:** | |  |  |
| Открываем форму Портфолио | | Форма открыта |  |
| **Шаги теста:** | |  |  |
| Выберите запись для редактирования и измените данные,  Название = Диплом за труды  Год = 2024  Категория = Волонтерство  Фотография = photo.png | | Данные успешно введены |  |
| Нажмите кнопку «Сохранить» | | Данные успешно сохранены | **Тест пройден** |
| **Постусловие:** | |  |  |
| Форма Портфолио  открыта | | Отображается форма Портфолио |  |

Таблица 5 – Тестирование некорректного редактирования информации в таблице Портфолио

|  |  |  |  |
| --- | --- | --- | --- |
| **Название:** | Тестирование некорректного редактирования информации в таблице Портфолио | | |
| **Функция:** | Циклы | | |
| **Действие** | | **Ожидаемый результат** | **Результат теста:**   * пройден * провален * заблокирован |
| **Предусловие:** | |  |  |
| Открываем форму Портфолио | | Форма открыта |  |
| **Шаги теста:** | |  |  |
| Выберите запись для редактирования и измените данные,  Название =  Год =  Категория =  Фотография = | | Данные успешно введены |  |
| Нажмите кнопку «Сохранить» | | Кнопка «Сохранить» заблокирована, поля пустые | **Тест пройден** |
| **Постусловие:** | |  |  |
| Форма Портфолио  открыта | | Отображается форма Портфолио |  |

Таблица 6 – Тестирование корректного добавления информации в таблицу Организации

|  |  |  |  |
| --- | --- | --- | --- |
| **Название:** | Тестирование корректного добавления информации в таблицу Организации | | |
| **Функция:** | Циклы | | |
| **Действие** | | **Ожидаемый результат** | **Результат теста:**   * пройден * провален * заблокирован |
| **Предусловие:** | |  |  |
| Открываем форму Подача заявки | | Форма открыта |  |
| **Шаги теста:** | |  |  |
| Нажмите кнопку оставить заяку  Заполните форму  Организация= Пульс жизни  ПочемуВыТекст = я активно учавствую в волонтерской деятельности | | Данные успешно введены |  |
| Нажмите кнопку «Отправить» | | Данные успешно сохранены | **Тест пройден** |
| **Постусловие:** | |  |  |
| Форма Подача заявки открыта | | Отображается форма Подача заявки |  |

Таблица 7 – Тестирование некорректного добавления информации в таблицу ОрганизацииПользователей

|  |  |  |  |
| --- | --- | --- | --- |
| **Название:** | Тестирование некорректного добавления информации в таблицу ОрганизацииПользователей | | |
| **Функция:** | Циклы | | |
| **Действие** | | **Ожидаемый результат** | **Результат теста:**   * пройден * провален * заблокирован |
| **Предусловие:** | |  |  |
| Открываем форму Подача заявки | | Форма открыта |  |
| **Шаги теста:** | |  |  |
| Нажмите кнопку Оставить заявку  Заполните форму  Организация=  ПочемуВыТекст = | | Данные успешно введены |  |
| Нажмите кнопку «Отправить» | | Кнопка «Отправить» заблокирована, поля пустые | **Тест пройден** |
| **Постусловие:** | |  |  |
| Форма Подача заявки  открыта | | Отображается форма Подача заявки |  |

Таблица 8 – Тестирование корректного редактирования информации в таблице ОрганизацииПользователей

|  |  |  |  |
| --- | --- | --- | --- |
| **Название:** | Тестирование корректного редактирования информации в таблице ОрганизацииПользователей | | |
| **Функция:** | Циклы | | |
| **Действие** | | **Ожидаемый результат** | **Результат теста:**   * пройден * провален * заблокирован |
| **Предусловие:** | |  |  |
| Открываем форму Заявки | | Форма открыта |  |
| **Шаги теста:** | |  |  |
| Выберите запись для редактирования и измените данные,  Организация= Пульс жизни  ПочемуВыТекст = я активно учавствую в волонтерской деятельности | | Данные успешно введены |  |
| Нажмите кнопку «Сохранить» | | Данные успешно сохранены | **Тест пройден** |
| **Постусловие:** | |  |  |
| Форма Заявки открыта | | Отображается форма Заявки |  |

Таблица 9 – Тестирование некорректного редактирования информации в таблице ОрганизацииПользователей

|  |  |  |  |
| --- | --- | --- | --- |
| **Название:** | Тестирование некорректного редактирования информации в таблице ОрганизацииПользователей | | |
| **Функция:** | Циклы | | |
| **Действие** | | **Ожидаемый результат** | **Результат теста:**   * пройден * провален * заблокирован |
| **Предусловие:** | |  |  |
| Открываем форму Зал | | Форма открыта |  |
| **Шаги теста:** | |  |  |
| Выберите запись для редактирования и измените данные,  Организация=  ПочемуВыТекст = | | Данные успешно введены |  |
| Нажмите кнопку «Сохранить» | | Кнопка «Сохранить» заблокирована, поля пустые | **Тест пройден** |
| **Постусловие:** | |  |  |
| Форма Заявки открыта | | Отображается форма Заявки |  |

**5.3 Составление отчета о тестировании**

Для форм проекта были созданы и проведены тесты, результаты сведены в таблицу:

Таблица 10 – Результаты тестирования

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | Прохождение | | |
| Форма | Число ТП | пройдено | | не пройдено |
| Заявки | 4 | 4 | | 0 |
| Портфолио | 4 | 4 | | 0 |
| ИТОГО | 8 | 8 | 0 | |

По результатам тестирования были выявлены и устранены следующие проблемы:

При выполнении теста таблица №10 Тестирование формы «Портфолио», Некорректный запрос

**Ошибка**

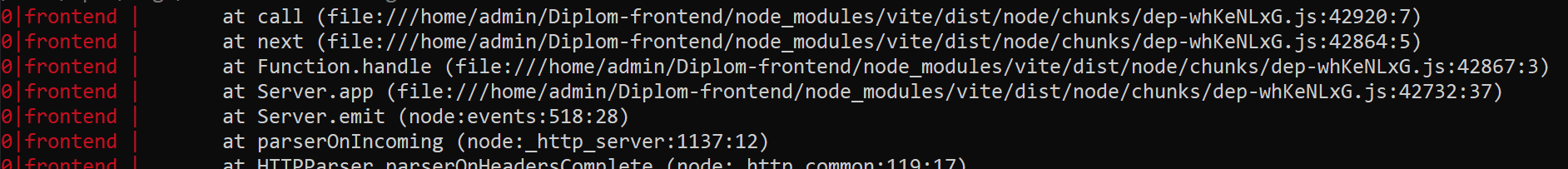
****

Рис.57 Ошибка теста №8

**Причина**

Возможен выбор файла любого разрешения для фотографии(txt, mp4).

**Принятые меры**

Ошибка устранена. Разрешение файла может быть только png или jpeg.

**ЗАКЛЮЧЕНИЕ**

**Оценка качества программного средства с помощью метрик**

Оценка качества программного средства проводится исходя из стандарта оценки качества ISО 9126.

На компьютере следующей конфигурации:

Установленная операционная система – Windows 10 Professional.

Процессор: Intel Pentium G2030 3.0 GHz.

ОЗУ: 4 Gb

Таблица 11 – Требования к количественным характеристикам качества программного средства

|  |  |  |  |
| --- | --- | --- | --- |
| **Характеристики качества** | **Мера** | **Требуемое значение** | **Реальное значение** |
| **Надежность** |  | | |
| *Завершенность:*  *-* наработка на отказ при отсутствии рестарта; | Часы | 10 | 10 |
| *Устойчивость:*  - относительные ресурсы на обеспечение надежности и рестарта. | % | 0,5 | 0,5 |
| *Восстанавливаемость:*  -длительность восстановления. | Минуты | 2 | 2 |
| *Доступность-готовность:*  -относительное время работоспособного функционирования. | Вероятность | 0,998 | 0,998 |
| **Эффективность** |  | | |
| *Временная эффективность:*  *-*время отклика - получение результатов на типовое задание;  - пропускная способность число типовых заданий, исполняемых в единицу времени. | Секунды  Число в минуту | 3  60 | 3  60 |
| *Используемость ресурсов:*  *-* относительная величина использования ресурсов ЭВМ при нормальном функционировании программного средства. |  | 0,001 | 0,001 |

Таблица 12 – Требования к качественным характеристикам программного средства.

|  |  |  |  |
| --- | --- | --- | --- |
| **Характеристики качества** | **Мера** | **Требуемое значение** | **Реальное значение** |
| **Практичность** |  | | |
| *Простота использования:*  **-**среднее время ввода заданий;  -среднее время отклика на задание. | Секунды  Секунды | 10  3 | 10  3 |
| *Изучаемость :*  - трудоемкость изучения применения ПС;  -продолжительность изучения;  -объем эксплуатационной документации; | Чел.-часы  Часы  Страницы | 5  2  15 | 5  2  15 |
| **Сопровождаемость** |  | | |
| *Изменяемость:*  -трудоемкость подготовки изменений;  -длительность подготовки изменений. | Чел.-часы  Часы | 5  5 | 5  5 |
| *Тестируемость:*  *-трудоемкость тестирования изменений;*  *- длительность тестирования изменений.* | Чел.-часы  Часы | 3  3 | 3  3 |
| **Мобильность** |  | | |
| *Адаптируемость:*  - трудоемкость адаптации;  -длительность адаптации. | Чел.-часы  Часы | 0,2  1 | 0,2  1 |
| *Простота установки:*  -трудоемкость инсталляции;  -длительность инсталляции. | Чел.-часы  Часы | 0,2  1 | 0,2  1 |
| *Замещаемость:*  -трудоемкость замены компонентов;  - длительность замены компонентов. | Чел.-часы  Часы | 0,2  0,2 | 0,2  0,2 |

Таблица 13 – Сложность

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Тип объекта | Количество | Вес | | | Итого |
|  |  | Простой | Средний | Сложный |  |
| Экран | 17 | x1 | x2 | **x3** | =51 |
| Отчет | 3 | x2 | x5 | **x8** | =24 |

Таблица 14 – Сложность экрана

|  |  |  |  |
| --- | --- | --- | --- |
| Экраны | Количество клиентских (клт) таблиц данных | | |
| Количество представлений | Всего <4 | Всего <8 | **Всего >8** |
|  | (<2 срв, <3 клт) | (2-3 срв, 3-5 клт) | (>3 срв, >5 клт) |
| <3 | Простой | Простой | Средний |
| 3-7 | Простой | Средний | Сложный |
| **>8** | Средний | Сложный | **Сложный** |

Таблица 15 – Сложность отчета

|  |  |  |  |
| --- | --- | --- | --- |
| Отчеты | Количество Возвратских (клт) таблиц данных | | |
| Количество представлений | Всего <4 | Всего <8 | Всего >8 |
|  | (<2 срв, <3 клт) | (2-3 срв, 3-5 клт) | **(>3 срв, >5 клт)** |
| 0 или 1 | Простой | Простой | Средний |
| **2 или 3** | Простой | Средний | **Сложный** |
| >4 | Средний | Сложный | Сложный |

Производительность = 6000 тысяч строк кода/1 месяца=6000 тысяч строк кода/месяц

Качество = 30 ошибок /6 тысяч строк кода=5 ошибок/тысяч строк кода

**СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

1. Онлайн ресурс: https://learn.javascript.ru/.

2. Трепачёв Дмитрий © 2012-2024 Основной учебник JavaScript 503с.

3. Хавербеке Марейн Выразительный JavaScript 2020 г. 480с.

4. [Майк МакГрат](https://www.google.com/search?sa=X&sca_esv=e60d7d3caff7b7c5&sca_upv=1&cs=0&sxsrf=ADLYWIJSpwMdOmJzq9ztrP62JnjPQB-UEw:1717239307892&q=%D0%9C%D0%B0%D0%B9%D0%BA+%D0%9C%D0%B0%D0%BA%D0%93%D1%80%D0%B0%D1%82&si=ACC90nwLLwns5sISZcdzuISy7t-NHozt8Cbt6G3WNQfC9ekAgC6woQkVPb6SRvxKEAw9WaSGwBjP2g-M_X7z4FX-JSa6GtDynGyVtR7vaUm32H4bFhf4RxYE3mF9xf5ZoNDxv1D8zrq7G3OiQc59cuQtJKJdvjL86jy4-cgKSZFs7t97Tn3qkDZQW_AVP4RJwYtPWqEHM1V0RkrTvgF7L-waQStid8iqIY8v6gTiTttEZm-UeZsLMTs%3D&ved=2ahUKEwjAg8P3nrqGAxU9AxAIHYQtAiYQmxMoAHoECBEQAg) JavaScript для начинающих 2023г 232с.

5. [Борис Новиков](https://www.google.com/search?sa=X&sca_esv=e60d7d3caff7b7c5&sca_upv=1&cs=0&sxsrf=ADLYWIILbLQR2ZFK_Y8AUm6bd1kvgNnmXw:1717239423682&q=%D0%91%D0%BE%D1%80%D0%B8%D1%81+%D0%9D%D0%BE%D0%B2%D0%B8%D0%BA%D0%BE%D0%B2&si=ACC90nwLLwns5sISZcdzuISy7t-NHozt8Cbt6G3WNQfC9ekAgAePtGtR4XSjxyn-iaT5GtZCePrGlxeOvNqLsPidxlNoUAnTUiLbvcYvkOuELFVoR9gJ6nX0ME7XZnJZ67YtRzPLL6uzKDpGQlyENc-7wrrcC7VnGms9j10YLH_aNIItMnevWlw_etvVWPlcBFnT4xmfO9FnlNn83DmahEtG394x487QLusHcEB2aXW8X-M8v0CaMD70IAiHic4-xL_uSBcblWCYfNnnJeeFT_5vA__0rpNTqA%3D%3D&ved=2ahUKEwiojd6un7qGAxWiIRAIHbbGNFIQmxMoAHoECBYQAg), [Екатерина Горшкова](https://www.google.com/search?sa=X&sca_esv=e60d7d3caff7b7c5&sca_upv=1&cs=0&sxsrf=ADLYWIILbLQR2ZFK_Y8AUm6bd1kvgNnmXw:1717239423682&q=%D0%95%D0%BA%D0%B0%D1%82%D0%B5%D1%80%D0%B8%D0%BD%D0%B0+%D0%93%D0%BE%D1%80%D1%88%D0%BA%D0%BE%D0%B2%D0%B0&si=ACC90nwLLwns5sISZcdzuISy7t-NHozt8Cbt6G3WNQfC9ekAgBgXZl0kLft9PIdUEuWftf1zutkvH7E0kpJBDsiyXEvGc3Go32tqkgrU5bIk2sk5ndJCtaID42yMAOQi9Ekyog2B2f2RkJYNni0MAjCQLnf0GiMyHH5UXpyJKFa9bO6UeQ39ryM8KODgG4V1o2HnSburTFLdMQnwLffvNodEHtT60aZwOvpHjGFZap92LVV3AN56wMEKQL8vM08zc_Dbaaquqv2vB9wJZ_MgxfUSAgRHUykbL8hYWpXuhjf127TdNkpN8L4%3D&ved=2ahUKEwiojd6un7qGAxWiIRAIHbbGNFIQmxMoAXoECBYQAw) Основы технологий баз данных 2020г. 582с.

6. Салахалдин Джуба, Волков Андрей Изучаем PostgreSQL 10 2021г. 400с.

7. Мартин Роберт Чистый код: создание, анализ и рефакторинг 2024г 464с.

8. ГОСТ 19.201-78.ЕСПД. Техническое задание. Требования к содержанию и оформлению

9. ГОСТ 19.102-77.ЕСПД. Стадии разработки

10. ГОСТ 19.505-79.ЕСПД. Руководство оператора. Требования к содержанию и оформлению

11. ГОСТ 19.002-80.ЕСПД. Схемы алгоритмов и программ. Правила выполнения

12. ГОСТ 19.404-79.ЕСПД. Пояснительная записка. Требования к содержанию и оформлению

13. Бхаргава Адитья Грокаем алгоритмы 2024г. 288с.

14. Флэнаган Дэвид JavaScript. Полное руководство 2021г. 720с.

15. Дронов В. А. React 17. Разработка веб-приложений на JavaScript 2022г. 384с.

16. Порселло Е., Бэнкс А. React: современные шаблоны для разработки приложений 2023г 320с.

**ПРИЛОЖЕНИЕ А**

ПМ студента колледжа

Текст кода программного средства

460.ЗМК.00002-01 12 01

**Текст кода программного средства**

**import { Outlet, useNavigate } from "react-router-dom";**

**import { Aside, Header } from "../../widgets";**

**import { useContext, useEffect } from "react";**

**import { RoleContext } from "../context";**

**import { Footer } from "../../widgets/footer/ui";**

**export const RootLayot = () => {**

**const navigate = useNavigate();**

**const { setRole } = useContext(RoleContext);**

**useEffect(() => {**

**const TokenUser = localStorage.getItem("token");**

**if (!TokenUser) return navigate("/Login");**

**setRole([TokenUser]);**

**}, []);**

**return (**

**<div className="h-screen">**

**<div className="grid grid-cols-root grid-rows-root gap-y-6 gap-x-12 dark:bg-slate-900 h-fit">**

**<Header />**

**<Aside />**

**<Outlet />**

**<Footer />**

**</div>**

**</div>**

**);**

**};**

**import { Navigate, Outlet } from "react-router-dom";**

**import { decodeJwt } from "../../shared/helpers/decodeJwt";**

**export const AdminGuardRoute = () => {**

**const token = decodeJwt();**

**const isTeacher = token.roles.includes("RESOURCES\_DEPARTMENT");**

**return isTeacher ? <Outlet /> : <Navigate to="/forbidden" replace />;**

**};**

**import {**

**Route,**

**createBrowserRouter,**

**createRoutesFromElements,**

**} from "react-router-dom";**

**import { RootLayot } from "../layout/Root";**

**import { Statement } from "../../widgets/statement/ui";**

**import { StatementTeacher } from "../../widgets/teacher/statement/ui";**

**import Login from "../../widgets/login/ui";**

**import { NotFound } from "../../pages/404/NotFound";**

**import { Settings } from "../../widgets";**

**import { StudentGuardRoute } from "./StudentGuardRoute";**

**import { Forbidden } from "../../pages/403/Forbidden";**

**import { PortfolioPage } from "../../pages/portfolio/ui";**

**import { Credit } from "../../widgets/teacher/credit/ui";**

**import { TeacherGuardRoute } from "./TeacherGuardRoute";**

**import { OrganizationPage } from "../../pages/organization/ui";**

**import { StudentCreditPage } from "../../pages/creditStudentPage/ui";**

**import { FuturePage } from "../../pages/futurePage/ui";**

**import { AdminGuardRoute } from "./AdminGuardRoute";**

**export const route = createBrowserRouter(**

**createRoutesFromElements(**

**<Route>**

**<Route element={<Login />} path="/Login" />**

**<Route element={<RootLayot />} path="/">**

**<Route element={<StudentGuardRoute />}>**

**<Route element={<Statement />} path="student/statement" />**

**<Route element={<OrganizationPage />} path="student/organizations" />**

**<Route element={<StudentCreditPage />} path="student/credit" />**

**<Route element={<FuturePage />} path="student/future" />**

**</Route>**

**<Route element={<TeacherGuardRoute />}>**

**<Route element={<Credit />} path="teacher/credit" />**

**<Route element={<StatementTeacher />} path="teacher/statement" />**

**</Route>**

**<Route element={<AdminGuardRoute />}>**

**<Route element={<Settings />} path="settings" />**

**</Route>**

**<Route element={<PortfolioPage />} path="portfolio" />**

**</Route>**

**<Route element={<Forbidden />} path="forbidden" />**

**<Route element={<NotFound />} path="\*" />**

**</Route>**

**)**

**);**

**import { Navigate, Outlet } from "react-router-dom";**

**import { decodeJwt } from "../../shared/helpers/decodeJwt";**

**export const StudentGuardRoute = () => {**

**const token = decodeJwt();**

**const isStudent = token.roles.includes("STUDENT");**

**return isStudent ? <Outlet /> : <Navigate to="/forbidden" replace />;**

**};**

**import { Navigate, Outlet } from "react-router-dom";**

**import { decodeJwt } from "../../shared/helpers/decodeJwt";**

**export const TeacherGuardRoute = () => {**

**const token = decodeJwt();**

**const isTeacher = token.roles.includes("TEACHER");**

**return isTeacher ? <Outlet /> : <Navigate to="/forbidden" replace />;**

**};**

**import { useNavigate } from "react-router-dom";**

**import { Button } from "../../shared/components";**

**import Svg403 from "./Svg403";**

**export const Forbidden = () => {**

**const navigate = useNavigate();**

**const goBack = () => navigate(-1);**

**return (**

**<div>**

**<Svg403 />**

**<h1>Нет доступа</h1>**

**<p>**

**Если вы уверены, что вам нужна эта страница, запросите доступ у**

**администратора**

**</p>**

**<Button onClick={goBack} variant="outlined">**

**Вернуться назад**

**</Button>**

**</div>**

**);**

**};**

**import Svg404 from "./Svg404";**

**export const NotFound = () => {**

**return (**

**<div>**

**<Svg404 width="100%" />**

**</div>**

**);**

**};**

**import axios from "axios";**

**import {**

**STUDENT\_CREDIT\_URL,**

**STUDENT\_LESSON\_URL,**

**STUDENT\_TEACHER\_URL,**

**TOKEN,**

**} from "../../../shared/const";**

**import { useQuery } from "@tanstack/react-query";**

**export const fetchStudentCredits = async () => {**

**const { data: credits } = await axios.get<Credits[]>(STUDENT\_CREDIT\_URL, {**

**headers: { Authorization: TOKEN() },**

**});**

**return Promise.all(**

**credits.map(async (elem) => {**

**const { data: lesson } = await axios.get<Lesson>(**

**STUDENT\_LESSON\_URL + "/" + elem.lessonId,**

**{ headers: { Authorization: TOKEN() } }**

**);**

**const { data: teacher } = await axios.get<Teacher>(**

**STUDENT\_TEACHER\_URL + "/" + lesson.userId,**

**{ headers: { Authorization: TOKEN() } }**

**);**

**const returnObj = {**

**lesson: lesson.name,**

**name: teacher.name,**

**surname: teacher.surname,**

**patronymic: teacher.patronymic,**

**office: elem.office,**

**date: elem.date,**

**deadLine: elem.deadLine,**

**id: elem.id,**

**};**

**return returnObj;**

**})**

**);**

**};**

**export const useGetStudentCredits = () =>**

**useQuery({**

**queryKey: ["my credits"],**

**queryFn: fetchStudentCredits,**

**});**

**interface Credits {**

**id: string;**

**lessonId: string;**

**userId: string;**

**deadLine: Date;**

**date: Date;**

**office: 110;**

**}**

**interface Teacher {**

**name: string;**

**surname: string;**

**patronymic: string;**

**email: string;**

**phoneNumber: string;**

**}**

**interface Lesson {**

**name: string;**

**userId: string;**

**}**

**import { formatDateTime } from "../../widgets/teacher/credit/api";**

**import { useGetStudentCredits } from "./api/useGetCredits";**

**export const StudentCreditPage = () => {**

**const { data: credits, isLoading } = useGetStudentCredits();**

**console.log(credits);**

**return (**

**<div className="">**

**<h1 className="text-4xl dark:text-white mb-4">Задолжности</h1>**

**<div className="flex-col flex justify-center">**

**<table className=" ">**

**<thead className="">**

**<tr className="border-b-2 border-Neutral-900  dark:text-white">**

**<th className="w-60">Предмет </th>**

**<th className="w-60">Преподаватель</th>**

**<th className="w-60">Дата</th>**

**<th className="w-60">Кабинет</th>**

**<th className="w-60">Крайний срок</th>**

**</tr>**

**</thead>**

**<tbody className="dark:text-white">**

**{isLoading ? (**

**<tr>**

**<td>Loading...</td>**

**</tr>**

**) : (**

**credits &&**

**credits.map((credit, index) => (**

**<tr key={index} className="border-b-2 border-Neutral-900 ">**

**<td className="text-center">{credit.lesson}</td>**

**<td className="text-center">**

**{credit.surname +**

**" " +**

**credit.name.slice(0, 1) +**

**". " +**

**credit.patronymic.slice(0, 1) +**

**"."}**

**</td>**

**<td className="text-center">**

**{formatDateTime(credit.date.toString())}**

**</td>**

**<td className="text-center">{credit.office}</td>**

**<td className="text-center text-red-700">**

**{formatDateTime(credit.deadLine.toString())}**

**</td>**

**</tr>**

**))**

**)}**

**</tbody>**

**</table>**

**<div className=" flex justify-center "></div>**

**</div>**

**</div>**

**);**

**};**

**import axios from "axios";**

**import { STUDENT\_FUTURE\_URL, TOKEN } from "../../../shared/const";**

**import { useQuery } from "@tanstack/react-query";**

**export const fetchFutures = () =>**

**axios.get<IFuture>(STUDENT\_FUTURE\_URL, {**

**headers: { Authorization: TOKEN() },**

**});**

**export const useGetFutures = () =>**

**useQuery({**

**queryKey: ["my futures"],**

**queryFn: fetchFutures,**

**select: ({ data }) => data,**

**});**

**interface IFuture {**

**works: IWork[];**

**learns: ILearn[];**

**}**

**interface IWork {**

**id: string;**

**name: string;**

**place: string;**

**cost: number;**

**phone: string;**

**url: string;**

**}**

**interface ILearn {**

**id: string;**

**name: string;**

**place: string;**

**cost: number;**

**description: string;**

**phone: string;**

**url: string;**

**photo: string;**

**}**

**import { FutureCard } from "../../widgets/future/futureCard/ui";**

**import { FutureSwiper } from "../../widgets/future/futureSwiper/ui";**

**import { useGetFutures } from "./api/useGetFutures";**

**export const FuturePage = () => {**

**const { data } = useGetFutures();**

**return (**

**<div>**

**<h1 className=" text-5xl dark:text-white">Возможности</h1>**

**{data?.works && <FutureSwiper data={data.works} />}**

**{data?.learns && <FutureCard data={data.learns} />}**

**</div>**

**);**

**};**

**export interface IOrganizations {**

**name: string;**

**id: string;**

**description: string;**

**photo: string;**

**}**

**import axios from "axios";**

**import { IOrganizations } from "./interfaces";**

**import { STUDENT\_ORGANIZATION\_URL, TOKEN } from "../../../shared/const";**

**import { useQuery } from "@tanstack/react-query";**

**export const fetchMyOrganizations = () => {**

**return axios.get<IOrganizations[]>(STUDENT\_ORGANIZATION\_URL, {**

**headers: {**

**Authorization: TOKEN(),**

**},**

**});**

**};**

**export const useGetMyOrganizations = () =>**

**useQuery({**

**queryKey: ["organizations"],**

**queryFn: fetchMyOrganizations,**

**select: ({ data }) => data,**

**});**

**import { useState } from "react";**

**import { Button, Hr } from "../../shared/components";**

**import { OrganizationCard } from "../../widgets/organizationCard/ui";**

**import { OrganizationModal } from "../../widgets/organizationModal/ui";**

**import { useGetMyOrganizations } from "./api/useGetMyOrganizations";**

**export const OrganizationPage = () => {**

**const { data, isLoading } = useGetMyOrganizations();**

**const [openModal, setOpenModal] = useState(false);**

**const modalElement = document.getElementById("modal") as HTMLDialogElement;**

**const onAnimationEnd = () => {**

**modalElement.classList.remove("hide");**

**modalElement.close();**

**modalElement.removeEventListener("animationend", onAnimationEnd);**

**};**

**const handleClickOpenModal = () => {**

**setOpenModal(true);**

**document.body.classList.add("overflow-hidden");**

**document**

**.querySelector("#react-portal-modal-organization")**

**?.classList.remove("hidden");**

**};**

**const handleClickCloseModal = () => {**

**setOpenModal(!openModal);**

**document**

**.querySelector("#react-portal-modal-organization")**

**?.classList.add("hidden");**

**document.body.classList.remove("overflow-hidden");**

**modalElement.addEventListener("animationend", onAnimationEnd);**

**};**

**return (**

**<div className="flex flex-col">**

**<div className="flex justify-between ">**

**<h1 className=" text-5xl dark:text-white">Членство</h1>**

**<Button variant="outlined" onClick={handleClickOpenModal}>**

**Оставить заявку**

**</Button>**

**</div>**

**<Hr />**

**<main>**

**{isLoading ? (**

**<div>Загрузка</div>**

**) : (**

**data &&**

**data.map((elem) => (**

**<OrganizationCard**

**key={elem.id}**

**description={elem.description}**

**name={elem.name}**

**photo={elem.photo}**

**id={elem.id}**

**/>**

**))**

**)}**

**</main>**

**<OrganizationModal open={openModal} onClose={handleClickCloseModal} />**

**</div>**

**);**

**};**

**import { Hr } from "../../shared/components";**

**import { AchievementForm, AchievementList } from "../../widgets/achievement";**

**export const PortfolioPage = () => {**

**return (**

**<div>**

**<h1 className=" text-5xl dark:text-white">Мое портфолио</h1>**

**<Hr />**

**<h3 className="text-xl mt-4 dark:text-white">Загрузить достижения</h3>**

**<AchievementForm />**

**<AchievementList />**

**</div>**

**);**

**};**

**import clsx from "clsx";**

**import React from "react";**

**interface props {**

**children: React.ReactNode;**

**onClick?: React.MouseEventHandler<HTMLButtonElement>;**

**className?: string;**

**value?: string;**

**type?: "submit" | "reset" | "button";**

**variant: "text" | "contained" | "outlined";**

**}**

**const typeButton = (type: "text" | "contained" | "outlined") => {**

**if (type === "contained")**

**return "bg-purple-600 text-white px-10 py-2 dark:text-white dark:bg-rose-600";**

**if (type === "text") return "bg-white text-violet-900 px-0 py-0";**

**return "border-[1px] border-violet-900 text-violet-900 px-10 py-2 dark:text-white dark:border-red-600";**

**};**

**export const Button = React.memo(**

**({ children, onClick, className, value, type, variant }: props) => {**

**const baseStyle = typeButton(variant);**

**return (**

**<button**

**className={clsx("rounded-xl  font-bold text-xl", baseStyle, className)}**

**onClick={onClick}**

**type={type}**

**value={value}**

**>**

**{children}**

**</button>**

**);**

**}**

**);**

**import clsx from "clsx";**

**export const Hr = ({ className }: props) => {**

**return (**

**<hr**

**className={clsx("border-b-2 border-gray-200 w-[98%] mt-5", className)}**

**/>**

**);**

**};**

**interface props {**

**className?: string;**

**}**

**import clsx from "clsx";**

**import { Link } from "react-router-dom";**

**interface props {**

**children: React.ReactNode;**

**to: string;**

**className?: string;**

**}**

**export const BaseLink = ({ children, to, className }: props) => {**

**return (**

**<Link**

**to={to}**

**className={clsx(**

**"border-[1px] border-purple-900 py-2 dark:border-rose-600 dark:text-white  grow grid place-items-center rounded-lg mt-4 hover:opacity-50",**

**className**

**)}**

**>**

**{children}**

**</Link>**

**);**

**};**

**import { createPortal } from "react-dom";**

**interface props {**

**children: React.ReactNode;**

**wrapperId: string;**

**}**

**const createWrapperAndAppendToBody = (wrapperId: string) => {**

**const wrapperElement = document.createElement("div");**

**wrapperElement.setAttribute("id", wrapperId);**

**document.body.appendChild(wrapperElement);**

**return wrapperElement;**

**};**

**const ReactPortal = ({**

**children,**

**wrapperId = "react-portal-wrapper",**

**}: props) => {**

**let element = document.getElementById(wrapperId);**

**if (!element) {**

**element = createWrapperAndAppendToBody(wrapperId);**

**}**

**return createPortal(children, element);**

**};**

**export default ReactPortal;**

**export const BASE\_URL = "http://prodd.dvotch.ru:3001/api";**

**export const CATEGORIES\_URL = BASE\_URL + "/category";**

**export const PORTFOLIO\_URL = BASE\_URL + "/portfolio";**

**export const ORGANIZATION\_URL = BASE\_URL + "/organizations";**

**export const SPECIALIZATION\_URL = BASE\_URL + "/specializations";**

**export const STUDENT\_URL = BASE\_URL + "/student";**

**export const STUDENT\_STATEMENT\_URL = STUDENT\_URL + "/statement";**

**export const STUDENT\_PORTFOLIO\_URL = STUDENT\_URL + "/portfolio";**

**export const STUDENT\_ORGANIZATION\_URL = STUDENT\_URL + "/organization";**

**export const STUDENT\_CREDIT\_URL = STUDENT\_URL + "/credits";**

**export const STUDENT\_TEACHER\_URL = STUDENT\_URL + "/user";**

**export const STUDENT\_LESSON\_URL = STUDENT\_URL + "/lesson";**

**export const STUDENT\_FUTURE\_URL = STUDENT\_URL + "/future";**

**export const STUDENT\_MARK\_URL = STUDENT\_URL + "/mark";**

**export const TEACHER\_URL = BASE\_URL + "/teacher";**

**export const TEACHER\_STATEMENT\_URL = TEACHER\_URL + "/statement";**

**export const ADMIN\_URL = BASE\_URL + "/adm";**

**export const ADMIN\_FUTURE\_URL = ADMIN\_URL + "/future";**

**export const TOKEN = () => localStorage.getItem("token");**

**import { TOKEN } from "../const";**

**import { jwtDecode } from "Jwt-decode";**

**import { JwtPayload } from "../interfaces";**

**export const decodeJwt = () => {**

**return jwtDecode<JwtPayload>(TOKEN() || "");**

**};**

**export interface JwtPayload {**

**id: string;**

**email: string;**

**roles: roles[];**

**organizationId: string;**

**group: number;**

**specialization: string;**

**name: string;**

**surname: string;**

**}**

**export type roles = "STUDENT" | "TEACHER" | "RESOURCES\_DEPARTMENT";**

**import { useQuery } from "@tanstack/react-query";**

**import axios from "axios";**

**import { CATEGORIES\_URL, TOKEN } from "../../../../shared/const";**

**export const useGetCategories = () => {**

**return useQuery({**

**queryKey: ["categories"],**

**queryFn: fetchCategories,**

**select: ({ data }) => {**

**return data;**

**},**

**});**

**};**

**export const fetchCategories = async () => {**

**return await axios.get<Categories[]>(CATEGORIES\_URL, {**

**headers: {**

**Authorization: TOKEN(),**

**},**

**});**

**};**

**interface Categories {**

**id: string;**

**name: string;**

**}**

**import { QueryClient, useMutation } from "@tanstack/react-query";**

**import axios from "axios";**

**import { Inputs } from "../ui";**

**import { PORTFOLIO\_URL, TOKEN } from "../../../../shared/const";**

**type dto = {**

**userId: string;**

**} & Inputs;**

**export const postPortfolio = (dto: dto) => {**

**return axios.post(PORTFOLIO\_URL, dto, {**

**headers: {**

**"Content-Type": `multipart/form-data`,**

**Authorization: TOKEN(),**

**},**

**});**

**};**

**export const useUploadPortfolio = (client: QueryClient) =>**

**useMutation({**

**mutationFn: (dto: dto) => {**

**return postPortfolio(dto);**

**},**

**onSuccess: () => {**

**client.invalidateQueries({ queryKey: ["portfolio list"] });**

**},**

**onSettled: () => {**

**client.invalidateQueries({ queryKey: ["portfolio list"] });**

**},**

**});**

**export const getYears = () => {**

**const date = new Date();**

**const returnDate = [];**

**for (let i = 0; i < 3; i++) returnDate.push(date.getFullYear() - i);**

**return returnDate;**

**};**

**export const years = getYears();**

**import { useForm, SubmitHandler } from "react-hook-form";**

**import { years } from "./config/years";**

**import { useEffect, useState } from "react";**

**import { useGetCategories } from "./api/categories";**

**import { useUploadPortfolio } from "./api/postPortfolio";**

**import { useQueryClient } from "@tanstack/react-query";**

**import { decodeJwt } from "../../../shared/helpers/decodeJwt";**

**import { Button } from "../../../shared/components";**

**export type Inputs = {**

**name: string;**

**year: number;**

**categoryId: string;**

**photo: FileList | Blob;**

**};**

**export const AchievementForm = () => {**

**const { register, handleSubmit, reset } = useForm<Inputs>();**

**const [fileName, setFileName] = useState("");**

**const { data: categories } = useGetCategories();**

**const queryClient = useQueryClient();**

**const { mutate: postPortfolio } = useUploadPortfolio(queryClient);**

**useEffect(() => {**

**const inputFileText = document.querySelector(**

**"input[type=file]"**

**) as HTMLInputElement;**

**inputFileText.addEventListener("change", () => {**

**if (inputFileText.files && inputFileText.files[0])**

**setFileName(inputFileText.files[0].name);**

**});**

**}, []);**

**const onSubmit: SubmitHandler<Inputs> = (data) => {**

**if (data.photo instanceof FileList) {**

**data.photo = new Blob([data.photo[0]], { type: "image/png" });**

**const userId = decodeJwt().id;**

**postPortfolio({ userId, ...data });**

**reset();**

**}**

**};**

**return (**

**<form**

**onSubmit={handleSubmit(onSubmit)}**

**className="mt-11 flex flex-col gap-6 text-xl w-[600px] dark:text-white"**

**id="addAchievement"**

**>**

**<label className="">**

**Название файла:**

**<br />**

**<input**

**type="text"**

**placeholder="Название"**

**{...register("name")}**

**className="border-gray-200 border-2 rounded-lg px-4 py-2 mt-2 w-[600px] dark:border-rose-600"**

**/>**

**</label>**

**<label>**

**Год:**

**<div className="select mt-2 dark:before:border-rose-600 dark:after:border-rose-600 dark:border-rose-600">**

**<select {...register("year")} className="indent-4" required>**

**<option value="" hidden>**

**Выберите из списка**

**</option>**

**{years.map((year) => (**

**<option value={+year} key={year}>**

**{year}**

**</option>**

**))}**

**</select>**

**</div>**

**</label>**

**<label>**

**Категория:**

**<div className="select mt-2 dark:before:border-rose-600 dark:after:border-rose-600 dark:border-rose-600">**

**<select**

**{...register("categoryId")}**

**className="indent-4"**

**required**

**defaultValue=""**

**>**

**<option value="" hidden>**

**Выберите из списка**

**</option>**

**{categories &&**

**categories.map((category) => (**

**<option value={category.id} key={category.id}>**

**{category.name}**

**</option>**

**))}**

**</select>**

**</div>**

**</label>**

**<label className="input-file w-[400px]" htmlFor="file">**

**Загрузить фотографию:**

**<br />**

**<div className="mt-2 flex ">**

**<span className="input-file-text flex-1 dark:border-rose-600 ">**

**<div className="w-[95%] text-nowrap overflow-hidden ">**

**{fileName}**

**</div>**

**</span>**

**<input**

**type="file"**

**className="dark:border-rose-600"**

**id="file"**

**{...register("photo")}**

**/>**

**<span className="input-file-btn dark:text-white dark:border-rose-600">**

**Выберите файл**

**</span>**

**</div>**

**</label>**

**<Button className="w-fit" type="submit" variant="outlined">**

**Добавить в портфолио**

**</Button>**

**</form>**

**);**

**};**

**import axios from "axios";**

**import { PORTFOLIO\_URL, TOKEN } from "../../../../shared/const";**

**import { QueryClient, useMutation } from "@tanstack/react-query";**

**export const deleteUserAchievement = (id: string) => {**

**return axios.delete(PORTFOLIO\_URL + "/" + id, {**

**headers: {**

**Authorization: TOKEN(),**

**},**

**});**

**};**

**export const useDeleteAchievement = (queryClient: QueryClient) =>**

**useMutation({**

**mutationFn: (id: string) => deleteUserAchievement(id),**

**onSettled: () =>**

**queryClient.invalidateQueries({ queryKey: ["portfolio list"] }),**

**onSuccess: () =>**

**queryClient.invalidateQueries({ queryKey: ["portfolio list"] }),**

**});**

**import axios from "axios";**

**import {**

**CATEGORIES\_URL,**

**STUDENT\_PORTFOLIO\_URL,**

**TOKEN,**

**} from "../../../../shared/const";**

**import { useQuery } from "@tanstack/react-query";**

**export const fetchMyAchievements = async (page: number) => {**

**const data = await axios.get<AchievementResponse>(**

**STUDENT\_PORTFOLIO\_URL + "?page=" + page,**

**{**

**headers: {**

**Authorization: TOKEN(),**

**},**

**}**

**);**

**return data.data;**

**};**

**export const fetchMCategories = async () => {**

**return await axios.get<Categories[]>(CATEGORIES\_URL, {**

**headers: {**

**Authorization: TOKEN(),**

**},**

**});**

**};**

**export const useGetAchievements = (page: number) => {**

**return useQuery({**

**queryKey: ["portfolio list", page],**

**queryFn: async () => {**

**const achievements = await fetchMyAchievements(page);**

**const categories = (await fetchMCategories()).data;**

**return achievements.data.map((elem) => {**

**return {**

**...elem,**

**categoryId: categories.find(**

**(category) => category.id === elem.categoryId**

**)?.name,**

**lastPage: achievements.meta.totalPage,**

**};**

**});**

**},**

**});**

**};**

**interface AchievementResponse {**

**data: Achievement[];**

**meta: {**

**totalPage: number;**

**};**

**}**

**interface Achievement {**

**id: string;**

**categoryId: string;**

**year: number;**

**name: string;**

**photo: string;**

**}**

**interface Categories {**

**id: string;**

**name: string;**

**}**

**import React from "react";**

**type PaginationProps = {**

**onNextPageClick: () => void;**

**onPrevPageClick: () => void;**

**disable: {**

**left: boolean;**

**right: boolean;**

**};**

**nav?: {**

**current: number;**

**total: number;**

**};**

**};**

**export const Pagination = React.memo((props: PaginationProps) => {**

**const { nav = null, disable, onNextPageClick, onPrevPageClick } = props;**

**const handleNextPageClick = () => {**

**onNextPageClick();**

**};**

**const handlePrevPageClick = () => {**

**onPrevPageClick();**

**};**

**return (**

**<div className="paginator">**

**<button**

**className="dark:text-white"**

**type="button"**

**onClick={handlePrevPageClick}**

**disabled={disable.left}**

**>**

**{"<"}**

**</button>**

**{nav && (**

**<span className="dark:text-white">**

**{nav.current + 1} / {nav.total}**

**</span>**

**)}**

**<button**

**className="dark:text-white"**

**type="button"**

**onClick={handleNextPageClick}**

**disabled={disable.right}**

**>**

**{">"}**

**</button>**

**</div>**

**);**

**});**

**import { useCallback, useState } from "react";**

**import { Button, Hr } from "../../../shared/components";**

**import { useGetAchievements } from "./api/useGetAchievements";**

**import { useDeleteAchievement } from "./api/useDeleteAchievement";**

**import { useQueryClient } from "@tanstack/react-query";**

**import { Pagination } from "./components/paginationButton";**

**export const AchievementList = () => {**

**const [page, setPage] = useState(0);**

**const { data, isLoading } = useGetAchievements(page);**

**const handleClickDeleteAchievement = (**

**event: React.SyntheticEvent<HTMLButtonElement>**

**) => {**

**if (event.target instanceof HTMLButtonElement)**

**deleteAchievement(event.target.value);**

**};**

**const quertClient = useQueryClient();**

**const { mutate: deleteAchievement } = useDeleteAchievement(quertClient);**

**const ROWS\_PER\_PAGE = 2;**

**const getTotalPageCount = (rowCount: number): number =>**

**Math.ceil(rowCount / ROWS\_PER\_PAGE);**

**const handleNextPageClick = useCallback(() => {**

**const current = page;**

**const next = current + 1;**

**const total = data ? getTotalPageCount(data[0]?.lastPage) : current;**

**console.log(next, total);**

**setPage(next < total ? next : current);**

**}, [page, data]);**

**const handlePrevPageClick = useCallback(() => {**

**const current = page;**

**const prev = current - 1;**

**setPage(prev >= 0 ? prev : current);**

**}, [page]);**

**return (**

**<>**

**<h2 className="text-3xl mt-8 dark:text-white">Загруженные достижения</h2>**

**{isLoading ? (**

**<div className="h-80">Loading...</div>**

**) : (**

**data &&**

**data.map((achievement) => (**

**<div key={achievement.id}>**

**<Hr />**

**<div className="grid h-40 w-10/12 grid-cols-achievementList items-center text-xl mt-4">**

**<img**

**src={achievement.photo}**

**alt="фотография достижения"**

**width={110}**

**className="text-white"**

**/>**

**<div>**

**<h4 className="font-bold dark:text-white">**

**{achievement.name}**

**</h4>**

**<span className="text-gray-400 dark:text-sub-100">**

**{achievement.year}**

**</span>**

**</div>**

**<p className="text-gray-400 dark:text-sub-100">**

**{achievement.categoryId}**

**</p>**

**<div className="flex gap-6">**

**<Button variant="outlined">Скачать</Button>**

**<Button**

**variant="outlined"**

**value={achievement.id}**

**onClick={handleClickDeleteAchievement}**

**>**

**Удалить**

**</Button>**

**</div>**

**</div>**

**</div>**

**))**

**)}**

**<div className="flex gap-6 achievement-list">**

**{data && data?.length > 0 && (**

**<Pagination**

**onNextPageClick={handleNextPageClick}**

**onPrevPageClick={handlePrevPageClick}**

**disable={{**

**left: page === 0,**

**right: page === getTotalPageCount(data[0].lastPage),**

**}}**

**nav={{ current: page, total: getTotalPageCount(data[0].lastPage) }}**

**/>**

**)}**

**</div>**

**</>**

**);**

**};**

**export interface ISpecialization {**

**id: string;**

**name: string;**

**description: string;**

**}**

**export interface IPostFuture {**

**name: string;**

**cost: number;**

**description: string | null;**

**phone: string | null;**

**photo: Blob | FileList | null;**

**place: string;**

**specializationId: string;**

**url: string;**

**work: boolean;**

**}**

**import axios from "axios";**

**import { ISpecialization } from "./interfaces";**

**import { SPECIALIZATION\_URL, TOKEN } from "../../../shared/const";**

**import { useQuery } from "@tanstack/react-query";**

**export const fetchSpecializations = () =>**

**axios.get<ISpecialization[]>(SPECIALIZATION\_URL, {**

**headers: { Authorization: TOKEN() },**

**});**

**export const useGetSpecializations = () =>**

**useQuery({**

**queryKey: ["all specializations"],**

**queryFn: fetchSpecializations,**

**select: ({ data }) => data,**

**});**

**import axios from "axios";**

**import { ADMIN\_FUTURE\_URL } from "../../../shared/const";**

**import { IPostFuture } from "../../addFuture/api/interfaces";**

**import { useMutation } from "@tanstack/react-query";**

**export const PostFuture = (data: IPostFuture) =>**

**axios.post(ADMIN\_FUTURE\_URL, data, {**

**headers: {**

**"Content-Type": `multipart/form-data`,**

**Authorization:**

**"Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6ImVmNWE1OWQ3LTdlNjYtNDdkZi04MzcxLWUwMmNmNDA4OGM0YSIsImVtYWlsIjoiYWRtMiIsInJvbGVzIjpbIlJFU09VUkNFU19ERVBBUlRNRU5UIl0sInNwZWNpYWxpemF0aW9uIjoiMDkuMDIuMDcg0J\_RgNC-0LPRgNCw0LzQvNC40YDQstC-0LDQvdC40LUg0LIg0LrQvtC80L\_RjNGO0YLQtdGA0L3Ri9GFINGB0LjRgdGC0LXQvNCw0YUuIiwiZ3JvdXAiOjIwNSwibmFtZSI6ItCQ0JTQnNCY0J0iLCJzdXJuYW1lIjoi0J3QlSDQo9CU0JDQm9Cv0KLQrCIsImlhdCI6MTcxNTA4MDY4MiwiZXhwIjoxNzE1Njg1NDgyfQ.qSopLNdgYndlG1Ea9QRk-IBBjB4CtwsaU8-UjlpcfQo",**

**},**

**});**

**export const usePostFuture = () =>**

**useMutation({**

**mutationFn: (data: IPostFuture) => PostFuture(data),**

**});**

**import { SubmitHandler, useForm, useWatch } from "react-hook-form";**

**import { Button } from "../../shared/components";**

**import { useEffect, useState } from "react";**

**import { useGetSpecializations } from "./api/useGetSpecializations";**

**import { usePostFuture } from "./api/usePostFuture";**

**export type Inputs = {**

**name: string;**

**description: string | null;**

**cost: number;**

**phone: string | null;**

**place: string;**

**specializationId: string;**

**url: string;**

**work: boolean;**

**photo: FileList | Blob | null;**

**};**

**export const AddFutureForm = () => {**

**const { register, handleSubmit, control, reset } = useForm<Inputs>({**

**defaultValues: { description: null, phone: null, photo: null },**

**});**

**const { data: specializations } = useGetSpecializations();**

**const { mutate: postFuture } = usePostFuture();**

**const [fileName, setFileName] = useState("");**

**const checked = useWatch({**

**control,**

**name: "work",**

**defaultValue: false,**

**});**

**useEffect(() => {**

**const inputFileText = document.querySelector("#filefuture");**

**if (inputFileText instanceof HTMLInputElement)**

**inputFileText.addEventListener("change", () => {**

**if (inputFileText.files && inputFileText.files[0])**

**setFileName(inputFileText.files[0].name);**

**});**

**}, [checked]);**

**const onSubmit: SubmitHandler<Inputs> = (data) => {**

**data.cost = +data.cost;**

**const inputFile = document.querySelector("#filefuture");**

**if (inputFile instanceof HTMLInputElement) {**

**const file = inputFile.files;**

**if (file) {**

**data.photo = new Blob([file[0]], { type: "image/png" });**

**}**

**}**

**postFuture(data);**

**reset();**

**};**

**return (**

**<form**

**onSubmit={handleSubmit(onSubmit)}**

**className="w-[520px] mt-8 flex flex-col gap-3"**

**>**

**<h2 className="text-purple-600 text-3xl font-bold place-self-center">**

**Зарегистрировать будущее**

**</h2>**

**<label>**

**Рабочее место:**

**<input type="checkbox" {...register("work")} className="ml-2" />**

**</label>**

**<label>**

**Введите название:**

**<input**

**{...register("name")}**

**className="w-full border-2 border-gray-200 p-2 mt-2"**

**required**

**placeholder="Введите название организации"**

**/>**

**</label>**

**<label>**

**{checked ? "Оплата за работу" : "Стоимость обучения"}:**

**<input**

**placeholder={checked ? "Оплата" : "Cтоимость"}**

**{...register("cost")}**

**required**

**type="number"**

**className="w-full border-2 border-gray-200 p-2 mt-2"**

**/>**

**</label>**

**<label>**

**Адрес сайта:**

**<input**

**placeholder="Адрес сайта"**

**{...register("url")}**

**required**

**className="w-full border-2 border-gray-200 p-2 mt-2"**

**/>**

**</label>**

**<label>**

**Примерное месторасположение:**

**<input**

**placeholder="Где находится организация"**

**{...register("place")}**

**required**

**className="w-full border-2 border-gray-200 p-2 mt-2"**

**/>**

**</label>**

**<label>**

**Специальность:**

**<div className="select mt-2 w-full">**

**<select**

**{...register("specializationId")}**

**className="indent-4"**

**required**

**defaultValue=""**

**>**

**<option value="" hidden>**

**Выберите из списка**

**</option>**

**{specializations &&**

**specializations.map((spec) => (**

**<option value={spec.id} key={spec.id}>**

**{spec.name}**

**</option>**

**))}**

**</select>**

**</div>**

**</label>**

**{!checked && (**

**<label>**

**Описание учебного заведения:**

**<div className="mt-2">**

**<textarea**

**placeholder="Расскажите об учебном заведении ..."**

**{...register("description", { value: null })}**

**required**

**className="w-full border-2 border-gray-200 h-32 resize-none p-2"**

**/>**

**</div>**

**</label>**

**)}**

**{checked && (**

**<label>**

**Номер телефона:**

**<input**

**placeholder="Номер телефона"**

**{...register("phone")}**

**required**

**className="w-full border-2 border-gray-200 p-2 mt-2"**

**/>**

**</label>**

**)}**

**{!checked && (**

**<section>**

**<label className="input-file w-[400px]">**

**Загрузить фотографию:**

**<br />**

**<div className="mt-2 flex">**

**<span className="input-file-text flex-1 ">**

**<div className="w-[95%] text-nowrap overflow-hidden">**

**{fileName}**

**</div>**

**</span>**

**<input**

**type="file"**

**className=""**

**id="filefuture"**

**{...register("photo")}**

**/>**

**<span className="input-file-btn">Выберите файл</span>**

**</div>**

**</label>**

**</section>**

**)}**

**<menu className="flex mt-4">**

**<Button variant="contained" className="px-4" type="submit">**

**Отправить**

**</Button>**

**</menu>**

**</form>**

**);**

**};**

**interface IPostOrganization {**

**name: string;**

**description: string;**

**logo: Blob | FileList;**

**}**

**import axios from "axios";**

**import { ORGANIZATION\_URL } from "../../../shared/const";**

**import { useMutation } from "@tanstack/react-query";**

**export const postOrganization = (data: IPostOrganization) => {**

**const form = new FormData();**

**form.append("logo", data.logo as Blob);**

**form.append("name", data.name);**

**form.append("description", data.description);**

**return axios.post<IPostOrganization>(ORGANIZATION\_URL, form, {**

**headers: {**

**"Content-Type": `multipart/form-data`,**

**Authorization:**

**"Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6ImVmNWE1OWQ3LTdlNjYtNDdkZi04MzcxLWUwMmNmNDA4OGM0YSIsImVtYWlsIjoiYWRtMiIsInJvbGVzIjpbIlJFU09VUkNFU19ERVBBUlRNRU5UIl0sInNwZWNpYWxpemF0aW9uIjoiMDkuMDIuMDcg0J\_RgNC-0LPRgNCw0LzQvNC40YDQstC-0LDQvdC40LUg0LIg0LrQvtC80L\_RjNGO0YLQtdGA0L3Ri9GFINGB0LjRgdGC0LXQvNCw0YUuIiwiZ3JvdXAiOjIwNSwibmFtZSI6ItCQ0JTQnNCY0J0iLCJzdXJuYW1lIjoi0J3QlSDQo9CU0JDQm9Cv0KLQrCIsImlhdCI6MTcxNDgyNjUwOCwiZXhwIjoxNzE1NDMxMzA4fQ.gH4C31QtEeEX-v7ostYOpd5YZWp3QbX\_lx\_5Ak0o\_EY",**

**},**

**});**

**};**

**export const usePostOrganization = () =>**

**useMutation({**

**mutationFn: (data: IPostOrganization) => postOrganization(data),**

**});**

**import { SubmitHandler, useForm } from "react-hook-form";**

**import { Button } from "../../shared/components";**

**import { useEffect, useState } from "react";**

**import { usePostOrganization } from "./api/usePostOrganization";**

**export type Inputs = {**

**name: string;**

**description: string;**

**logo: FileList | Blob;**

**};**

**export const AddOrganizationForm = () => {**

**const { register, handleSubmit, reset } = useForm<Inputs>();**

**const { mutate: postOrganization } = usePostOrganization();**

**const [fileName, setFileName] = useState("");**

**useEffect(() => {**

**const inputFileText = document.querySelector("#fileorg");**

**if (inputFileText instanceof HTMLInputElement)**

**inputFileText.addEventListener("change", () => {**

**if (inputFileText.files && inputFileText.files[0])**

**setFileName(inputFileText.files[0].name);**

**});**

**}, []);**

**const onSubmit: SubmitHandler<Inputs> = (data) => {**

**const inputFile: HTMLInputElement = document.querySelector("#fileorg")!;**

**const file = inputFile.files;**

**if (file) {**

**data.logo = new Blob([file[0]], { type: "image/png" });**

**postOrganization(data);**

**}**

**reset();**

**};**

**return (**

**<form**

**onSubmit={handleSubmit(onSubmit)}**

**className="w-[520px] mt-8 flex flex-col gap-3"**

**>**

**<h2 className="text-purple-600 text-3xl font-bold place-self-center">**

**Зарегистрировать организацию**

**</h2>**

**<section className="mt-4 w-4/5">**

**<label>**

**Введите название:**

**<div className=" mt-2 w-full ">**

**<input**

**{...register("name")}**

**className="w-full border-2 p-1"**

**required**

**placeholder="введите название организации"**

**/>**

**</div>**

**</label>**

**</section>**

**<section className="w-4/5">**

**<label>**

**Описание организации:**

**<div className="mt-2">**

**<textarea**

**placeholder="Расскажите о себе ..."**

**{...register("description")}**

**required**

**className="w-full border-2 border-gray-200 h-32 resize-none p-2"**

**/>**

**</div>**

**</label>**

**</section>**

**<section>**

**<label className="input-file w-[400px]">**

**Загрузить фотографию:**

**<br />**

**<div className="mt-2 flex">**

**<span className="input-file-text flex-1 ">**

**<div className="w-[95%] text-nowrap overflow-hidden">**

**{fileName}**

**</div>**

**</span>**

**<input**

**type="file"**

**className=""**

**id="fileorg"**

**{...register("logo")}**

**/>**

**<span className="input-file-btn">Выберите файл</span>**

**</div>**

**</label>**

**</section>**

**<menu className="flex mt-4">**

**<Button variant="contained" className="px-4" type="submit">**

**Отправить**

**</Button>**

**</menu>**

**</form>**

**);**

**};**

**import axios from "axios";**

**import { useQuery } from "@tanstack/react-query";**

**import { TOKEN } from "../../../shared/const";**

**export const getLogo = async () => {**

**return await axios.get<string>("http://prod.dvotch.ru:3001/api/user/logo", {**

**headers: {**

**Authorization: TOKEN(),**

**},**

**});**

**};**

**export const useGetLogo = () => {**

**return useQuery({**

**queryKey: ["logo"],**

**queryFn: getLogo,**

**select: ({ data }) => {**

**return data;**

**},**

**});**

**};**

**import axios from "axios";**

**import { IUserMe } from "../interface";**

**import { useQuery } from "@tanstack/react-query";**

**export const fetchUserMe = async () => {**

**const TOKEN = localStorage.getItem("token");**

**return await axios.post<IUserMe[]>("http://prod.dvotch.ru:3001/api/user/me", {**

**headers: {**

**Authorization: TOKEN,**

**},**

**});**

**};**

**export const useUserMe = () => {**

**return useQuery({**

**queryKey: ["infoMe"],**

**queryFn: () => fetchUserMe(),**

**select: ({ data }) => data,**

**});**

**};**

**export const LinksGeneral: LinkData[] = [**

**{**

**path: "/portfolio",**

**text: "Моё портфолио",**

**},**

**];**

**export const LinksStudent: LinkData[] = [**

**{**

**path: "/student/statement",**

**text: "Журнал",**

**},**

**...LinksGeneral,**

**{**

**path: "/student/organizations",**

**text: "Членство",**

**},**

**{**

**path: "/student/credit",**

**text: "Задолжности",**

**},**

**{**

**path: "/student/future",**

**text: "Возможности",**

**},**

**];**

**export const LinksTeacher: LinkData[] = [**

**{**

**path: "/teacher/statement",**

**text: "Журнал",**

**},**

**...LinksGeneral,**

**{**

**path: "/teacher/credit",**

**text: "Задолжности",**

**},**

**];**

**export interface LinkData {**

**path: string;**

**text?: string;**

**}**

**export enum Role {**

**"STUDENT" = "Студент",**

**"TEACHER" = "Учитель",**

**"RESOURCES\_DEPARTMENT" = "Отдел кадров",**

**}**

**export interface IUserMe {**

**id: string;**

**ebail: string;**

**roles: string;**

**organizationId: string;**

**specialization: string;**

**group: string;**

**name: string;**

**surname: string;**

**}**

**import { useGetLogo } from "./api/logo";**

**import { decodeJwt } from "../../shared/helpers/decodeJwt";**

**import { LinksStudent, LinksTeacher } from "./config/links";**

**import { BaseLink } from "../../shared/components";**

**import clsx from "clsx";**

**import { Role } from "./config/role";**

**export const Aside = () => {**

**const { data, isLoading, isRefetching } = useGetLogo();**

**const decode = decodeJwt();**

**const linksMenu = decode.roles.includes("STUDENT")**

**? LinksStudent**

**: LinksTeacher;**

**return (**

**<aside className=" py-8 border-[1px] border-purple-900 text-sub-100 text-lg h-fit grid place-items-center mb-4 ml-8 row-start-2 dark:border-rose-600">**

**{isLoading || isRefetching ? (**

**<div>Loading image...</div>**

**) : (**

**<img src={data} id="img" width={150} />**

**)}**

**<h2 className="text-sub-100 text-2xl dark:text-white">**

**{decode.name + " " + decode.surname}**

**</h2>**

**<span className="px-12 py-1 bg-purple-600  text-white font-bold rounded-xl mt-5 dark:bg-purple-900">**

**{Role[decode.roles[0]]}**

**</span>**

**<p className="mt-5">Зеленодольск, Татарстан</p>**

**{decode.specialization && (**

**<p className="w-64 text-center mt-2">{decode.specialization}</p>**

**)}**

**{decode.group && <p className="mt-2">{decode.group + " группа"}</p>}**

**<ul className="mt-6 w-60 ">**

**{linksMenu.map((link, index) => (**

**<li key={index}>**

**<BaseLink**

**to={link.path}**

**className={clsx(index === 0 ? "text-black" : "")}**

**>**

**{link.text}**

**</BaseLink>**

**</li>**

**))}**

**</ul>**

**</aside>**

**);**

**};**

**export const Footer = () => {**

**return (**

**<footer className="text-white py-8 bg-gradient-to-b from-white to-purple-200 h-40 opacity-100 dark:opacity-100 dark:bg-gradient-to-b dark:from-slate-900 dark:to-gray-600 row-start-3 col-span-3">**

**<div className="container mx-auto flex flex-row md:flex-row items-center justify-center">**

**<div>**

**<h3 className="text-lg font-bold  fonst-sans text-black opacity-50 dark:text-white">**

**Зеленодольский механический колледж © 1939-2024.**

**</h3>**

**</div>**

**<p className="text-sm text-black dark:text-white opacity-50">**

**Все права защищены. Если не указано иное, все материалы сайта доступны**

**по лицензии Creative Commons CC-BY-SA 4.0. Прочтите нашу политику**

**конфиденциальности и пользовательское соглашение.**

**</p>**

**</div>**

**</footer>**

**);**

**};**

**import { Button } from "../../../shared/components";**

**import { ILearn } from "../interfaces";**

**export const FutureCard = ({ data }: { data: ILearn[] }) => {**

**return (**

**<div className="mt-12 flex flex-col gap-6 w-5/6 ">**

**{data.map((elem) => {**

**return (**

**<div**

**key={elem.id}**

**className="bg-white flex rounded-xl text-xl gap-4 p-4 border-[1px] shadow-lg dark:bg-slate-900 dark:border-rose-600"**

**>**

**<div className="relative">**

**<img src={elem.photo} alt="" width={200} />**

**<p className="absolute text-white bottom-3 left-3">**

**{"от " + elem.cost + " ₽/ год"}**

**</p>**

**</div>**

**<div className="flex flex-col gap-2 dark:text-white">**

**<h2>{elem.place}</h2>**

**<p className="font-medium w-5/6">{elem.name}</p>**

**<p>{elem.description}</p>**

**<Button variant="contained" className="w-fit">**

**<a href={elem.url} target="\_blank">**

**Откликнуться**

**</a>**

**</Button>**

**</div>**

**</div>**

**);**

**})}**

**</div>**

**);**

**};**

**import { Swiper, SwiperSlide } from "swiper/react";**

**import { Pagination } from "swiper/modules";**

**import { Button } from "../../../shared/components";**

**import { IWork } from "../interfaces";**

**import { CSSProperties, useContext, useState } from "react";**

**import "swiper/css";**

**import "swiper/css/pagination";**

**import { RoleContext } from "../../../app/context";**

**export const FutureSwiper = ({ data }: { data: IWork[] }) => {**

**const [showPhone, setShowPhone] = useState(false);**

**const { theme } = useContext(RoleContext);**

**const paginationColor = theme ? "rgb(225 29 72 )" : "rgb(147 51 234 )";**

**return (**

**<div className="w-[1224px] mt-8">**

**<Swiper**

**style={**

**{**

**"--swiper-pagination-color": paginationColor,**

**"--swiper-pagination-bullet-inactive-color": paginationColor,**

**} as CSSProperties**

**}**

**spaceBetween={50}**

**slidesPerView={2}**

**className="w-full h-60 "**

**modules={[Pagination]}**

**pagination={{**

**clickable: true,**

**}}**

**grabCursor**

**>**

**{data.map((elem) => {**

**return (**

**<SwiperSlide style={{ width: "680px" }} key={elem.id}>**

**<div className="flex justify-between flex-col h-3/4 border-l-4 border-l-purple-600 p-2 dark:border-rose-600">**

**<div className="text-2xl dark:text-white">**

**<h2>{elem.name}</h2>**

**<h2>{"от" + elem.cost + "₽"}</h2>**

**<p className="text-gray-400 dark:text-sub-100">**

**{elem.place}**

**</p>**

**</div>**

**<div className="flex justify-between text-base">**

**<Button variant="contained">**

**<a href={elem.url} target="\_blank">**

**Откликнуться**

**</a>**

**</Button>**

**<Button**

**variant="outlined"**

**onClick={() => setShowPhone(!showPhone)}**

**>**

**{showPhone ? elem.phone : "Показать контакты"}**

**</Button>**

**</div>**

**</div>**

**</SwiperSlide>**

**);**

**})}**

**</Swiper>**

**</div>**

**);**

**};**

**export interface IWork {**

**id: string;**

**name: string;**

**place: string;**

**cost: number;**

**phone: string;**

**url: string;**

**}**

**export interface ILearn {**

**id: string;**

**name: string;**

**place: string;**

**cost: number;**

**description: string;**

**phone: string;**

**url: string;**

**photo: string;**

**}**

**import Logo from "../../shared/assets/zmk.png";**

**import Brightness3Icon from "@mui/icons-material/Brightness3";**

**import RoomIcon from "@mui/icons-material/Room";**

**import { Link, useNavigate } from "react-router-dom";**

**import { Button } from "../../shared/components";**

**import { QueryCache } from "@tanstack/react-query";**

**import { useContext, useEffect, useState } from "react";**

**import { RoleContext } from "../../app/context";**

**export const Header = () => {**

**const navigate = useNavigate();**

**const cache = new QueryCache();**

**const { setTheme: setThemeContext } = useContext(RoleContext);**

**const handleLogout = () => {**

**localStorage.removeItem("token");**

**navigate("/Login");**

**cache.clear();**

**};**

**const handleRedirect = () => {**

**window.open(**

**"https://yandex.ru/maps/org/zelenodolskiy\_mekhanicheskiy\_kolledzh/1168389123/?ll=48.488733%2C55.851040&z=16.49",**

**"\_blank"**

**);**

**};**

**const [theme, setTheme] = useState("light");**

**useEffect(() => {**

**if (theme === "dark") {**

**document.documentElement.classList.add("dark");**

**setThemeContext("dark");**

**} else {**

**document.documentElement.classList.remove("dark");**

**setThemeContext("");**

**}**

**}, [theme]);**

**const handleThemeSwitch = () => {**

**setTheme(theme === "dark" ? "light" : "dark");**

**};**

**return (**

**<div className="  border-b-2 border-gray-200 col-span-3">**

**<header className="px-6 pt-8 pb-2 border-b-2 border-gray-200  col-span-2 dark:bg-slate-900">**

**<div className="flex justify-between ">**

**<div className="flex ">**

**<Link to="/">**

**<img**

**className="dark:mix-blend-screen hover:opacity-50 "**

**src={Logo}**

**width={50}**

**/>**

**</Link>**

**<p className="fonst-sans text-base text-sub-100 ml-2">**

**ЗМК <br /> Зеленодольский механический колледж**

**</p>**

**</div>**

**<div className="flex">**

**{theme === "light" ? (**

**<button className="w-10 h-10 mt-2 " onClick={handleThemeSwitch}>**

**<Brightness3Icon**

**sx={{ fontSize: 40 }}**

**className="visible hover:opacity-50 "**

**/>**

**</button>**

**) : (**

**<button className="w-10 h-10 mt-2 " onClick={handleThemeSwitch}>**

**<svg**

**className="w-8 h-8 text-orange-600  hover:opacity-50  dark:text-white"**

**viewBox="0 0 24 24"**

**fill="none"**

**stroke="currentColor"**

**stroke-width="2"**

**stroke-linecap="round"**

**stroke-linejoin="round"**

**>**

**<circle cx="12" cy="12" r="5" />**

**<line x1="12" y1="1" x2="12" y2="3" />**

**<line x1="12" y1="21" x2="12" y2="23" />**

**<line x1="4.22" y1="4.22" x2="5.64" y2="5.64" />**

**<line x1="18.36" y1="18.36" x2="19.78" y2="19.78" />**

**<line x1="1" y1="12" x2="3" y2="12" />**

**<line x1="21" y1="12" x2="23" y2="12" />**

**<line x1="4.22" y1="19.78" x2="5.64" y2="18.36" />**

**<line x1="18.36" y1="5.64" x2="19.78" y2="4.22" />**

**</svg>**

**</button>**

**)}**

**<div className="fonst-sans text-sub-100 flex justify-center items-center">**

**<RoomIcon sx={{ fontSize: 40 }} className="ml-4" />**

**<span onClick={handleRedirect} className="">**

**<button className="mr-4 dark:text-white hover:opacity-50 ">**

**Зеленодольск**

**</button>**

**</span>**

**</div>**

**<Button**

**className="hover:opacity-50  "**

**onClick={handleLogout}**

**variant="outlined"**

**>**

**Выйти**

**</Button>**

**</div>**

**</div>**

**<nav className="mt-4 ml-8">**

**<Link**

**to="/"**

**className="fonst-sans text-base text-text-100 align-middle hover:opacity-50   pb-2 dark:text-white "**

**>**

**Мой профиль**

**</Link>**

**<Link**

**to="/settings"**

**className="fonst-sans text-base text-text-100 align-middle ml-4 hover:opacity-50  pb-2 dark:text-white "**

**>**

**Настройки**

**</Link>**

**</nav>**

**</header>**

**</div>**

**);**

**};**

**import { useState, ChangeEvent, FormEvent } from "react";**

**import shecterin from "../../shared/assets/Прямоугольник 2.png";**

**import { jwtDecode } from "Jwt-decode";**

**import { useNavigate } from "react-router-dom";**

**import { JwtPayload, roles } from "../../shared/interfaces";**

**const AuthForm = () => {**

**const navigate = useNavigate();**

**const [formData, setFormData] = useState({**

**email: "",**

**password: "",**

**});**

**const handleLogin = (roles: roles[]) => {**

**if (roles.includes("STUDENT") || roles.includes("TEACHER")) {**

**return navigate("/", { replace: true });**

**}**

**return navigate("/settings", { replace: true });**

**};**

**const handleChange = (e: ChangeEvent<HTMLInputElement>) => {**

**const { name, value } = e.target;**

**setFormData({**

**...formData,**

**[name]: value,**

**});**

**};**

**const handleSubmit = (e: FormEvent<HTMLFormElement>) => {**

**e.preventDefault();**

**fetch("http://prod.dvotch.ru:3001/api/auth/login", {**

**method: "POST",**

**headers: {**

**"Content-Type": "application/json",**

**},**

**body: JSON.stringify(formData),**

**})**

**.then((response) => response.json())**

**.then((response) => {**

**const token = response;**

**if (token.accessToken) {**

**localStorage.setItem("token", token.accessToken);**

**const decoded = jwtDecode<JwtPayload>(token.accessToken);**

**return handleLogin(decoded.roles);**

**}**

**alert("Неверный логин или пароль");**

**})**

**.catch((error) => {**

**console.error(error);**

**});**

**};**

**return (**

**<div className="flex items-center justify-center  h-screen">**

**<div className="">**

**<img src={shecterin} alt="" className="size-96 " />**

**</div>**

**<div className="bg-sky-600 size-96 flex flex-col  gap-4 ">**

**<div className="text-white text-3xl font-medium mt-16 ml-20">**

**Авторизация**

**</div>**

**<form**

**onSubmit={handleSubmit}**

**className="flex flex-col gap-8 items-center"**

**>**

**<div className="border-b-2 border-neutral-100 w-72">**

**<input**

**className="placeholder-white placeholder-opacity-80 ml-4 text-white"**

**name="email"**

**type="text"**

**value={formData.email}**

**onChange={handleChange}**

**placeholder="Логин"**

**/>**

**</div>**

**<div className="border-b-2 border-neutral-100 w-72">**

**<input**

**className="placeholder-white placeholder-opacity-80 ml-4 text-white"**

**name="password"**

**type="password"**

**value={formData.password}**

**onChange={handleChange}**

**placeholder="Пароль"**

**/>**

**</div>**

**<button**

**type="submit"**

**className="bg-white text-sky-600 size-4/5 font-medium"**

**>**

**Войти**

**</button>**

**</form>**

**</div>**

**</div>**

**);**

**};**

**export default AuthForm;**

**import { QueryClient } from "@tanstack/react-query";**

**import { Button, Hr } from "../../shared/components";**

**import { useLeaveFromOrganization } from "../organizationModal/api/useLeaveFromOrganization";**

**import { memo } from "react";**

**interface props {**

**name: string;**

**description: string;**

**photo: string;**

**id: string;**

**}**

**export const OrganizationCard = memo(**

**({ name, description, photo, id }: props) => {**

**const queryClient = new QueryClient();**

**const { mutate: leaveFromOrganization } = useLeaveFromOrganization();**

**const handleClickLeave = () => {**

**leaveFromOrganization(id);**

**queryClient.invalidateQueries({ queryKey: ["organizations"] });**

**};**

**return (**

**<>**

**<div className="grid grid-rows-organizationCard grid-cols-organizationCard h-60 gap-x-4 mt-4 gap-4">**

**<img**

**src={photo}**

**alt="Лого организации"**

**className="row-span-3 self-center dark:text-white"**

**/>**

**<h2 className="col-span-3 text-2xl dark:text-white">{name}</h2>**

**<p className="col-span-3 text-xl text-gray-400 overflow-auto">**

**{description}**

**</p>**

**<Button variant="outlined" onClick={handleClickLeave}>**

**Выйти**

**</Button>**

**</div>**

**<Hr />**

**</>**

**);**

**}**

**);**

**import axios from "axios";**

**import { IOrganizations } from "../../../pages/organization/api/interfaces";**

**import { ORGANIZATION\_URL, TOKEN } from "../../../shared/const";**

**import { useQuery } from "@tanstack/react-query";**

**export const fetchAllOrganization = () => {**

**return axios.get<IOrganizations[]>(ORGANIZATION\_URL, {**

**headers: { Authorization: TOKEN() },**

**});**

**};**

**export const useGetOrganizations = () =>**

**useQuery({**

**queryKey: ["all organizations"],**

**queryFn: fetchAllOrganization,**

**select: ({ data }) => data,**

**});**

**import axios from "axios";**

**import { STUDENT\_ORGANIZATION\_URL, TOKEN } from "../../../shared/const";**

**import { useMutation } from "@tanstack/react-query";**

**export const deleteUserOrganization = (id: string) => {**

**return axios.delete(STUDENT\_ORGANIZATION\_URL + "/" + id, {**

**headers: { Authorization: TOKEN() },**

**});**

**};**

**export const useLeaveFromOrganization = () =>**

**useMutation({**

**mutationFn: (id: string) => deleteUserOrganization(id),**

**});**

**import axios from "axios";**

**import { STUDENT\_ORGANIZATION\_URL, TOKEN } from "../../../shared/const";**

**import { useMutation } from "@tanstack/react-query";**

**interface IApplication {**

**organizationId: string;**

**whyYouText: string;**

**}**

**export const PostApplicationToOrganization = (data: IApplication) => {**

**return axios.post<IApplication>(**

**STUDENT\_ORGANIZATION\_URL,**

**{ ...data, status: false },**

**{**

**headers: {**

**Authorization: TOKEN(),**

**"Content-Type": `application/json`,**

**},**

**}**

**);**

**};**

**export const usePostApplication = () =>**

**useMutation({**

**mutationFn: (data: IApplication) => PostApplicationToOrganization(data),**

**});**

**import { SubmitHandler, useForm } from "react-hook-form";**

**import { useGetOrganizations } from "./api/useGetOrganizations";**

**import { Button } from "../../shared/components";**

**import DisabledByDefaultIcon from "@mui/icons-material/DisabledByDefault";**

**import ReactPortal from "../../shared/components/portals/portal";**

**import { usePostApplication } from "./api/usePostApplication";**

**export type Inputs = {**

**organizationId: string;**

**whyYouText: string;**

**};**

**interface props {**

**open: boolean;**

**onClose: React.MouseEventHandler;**

**}**

**export const OrganizationModal = ({ open, onClose }: props) => {**

**const { register, handleSubmit, reset } = useForm<Inputs>();**

**const { data: organizations } = useGetOrganizations();**

**const { mutate: postApplication } = usePostApplication();**

**const onSubmit: SubmitHandler<Inputs> = (data) => {**

**postApplication(data);**

**reset();**

**};**

**if (!open) return null;**

**return (**

**<ReactPortal wrapperId="react-portal-modal-organization">**

**<div className="w-screen h-screen absolute bg-slate-200/70 top-0 left-0 grid place-items-center ">**

**<div className="p-4 w-[480px] h-[520px] bg-white dark:bg-slate-800 rounded-lg">**

**<form**

**method="dialog"**

**onSubmit={handleSubmit(onSubmit)}**

**className="p-4 flex flex-col w-full gap-3 font-medium"**

**>**

**<button**

**onClick={onClose}**

**className="text-purple-600 place-self-end dark:text-white"**

**>**

**<DisabledByDefaultIcon sx={{ fontSize: 40 }} />**

**</button>**

**<h2 className="text-purple-600 text-3xl font-bold place-self-center dark:text-white">**

**Подача заявки**

**</h2>**

**<section className="mt-4 w-4/5 dark:text-white">**

**<label>**

**Выберите организацию:**

**<div className="select mt-2 w-full dark:before:border-rose-600 dark:after:border-rose-600 dark:border-rose-600 ">**

**<select**

**{...register("organizationId")}**

**className="indent-4"**

**required**

**defaultValue=""**

**>**

**<option value="" hidden>**

**Выберите из списка**

**</option>**

**{organizations &&**

**organizations.map((elem) => (**

**<option value={elem.id}>{elem.name}</option>**

**))}**

**</select>**

**</div>**

**</label>**

**</section>**

**<section className="w-4/5 dark:text-white ">**

**<label>**

**Почему именно вы должны стать участником:**

**<div className="mt-2">**

**<textarea**

**placeholder="Напишите почему ..."**

**{...register("whyYouText")}**

**required**

**className="w-full border-2 border-gray-200 h-32 resize-none p-2 dark:border-rose-600"**

**/>**

**</div>**

**</label>**

**</section>**

**<menu className="flex justify-center">**

**<Button variant="contained" className="px-4" type="submit">**

**Отправить**

**</Button>**

**</menu>**

**</form>**

**</div>**

**</div>**

**</ReactPortal>**

**);**

**};**

**import { useQuery } from "@tanstack/react-query";**

**import axios from "axios";**

**import { ILesson } from "../interfaces";**

**import { TOKEN } from "../../../shared/const";**

**export const fetchStudentLessons = async (quater: number) => {**

**return await axios.get<ILesson[]>(**

**"http://prod.dvotch.ru:3001/api/student/lessons/" + quater,**

**{**

**headers: {**

**Authorization: TOKEN(),**

**},**

**}**

**);**

**};**

**export const useLessons = (quater: number) => {**

**return useQuery({**

**queryKey: ["lessons student", quater],**

**queryFn: () => fetchStudentLessons(quater),**

**select: ({ data }) => data,**

**});**

**};**

**import { useQuery } from "@tanstack/react-query";**

**import axios from "axios";**

**import { IMark } from "../interfaces";**

**import { TOKEN } from "../../../shared/const";**

**export const fetchStudentMarks = async (lessonId: string) => {**

**return await axios.get<IMark[]>(**

**"http://prod.dvotch.ru:3001/api/student/marks/" + lessonId,**

**{**

**headers: {**

**Authorization: TOKEN(),**

**},**

**}**

**);**

**};**

**export const useMarks = (lessonId: string) => {**

**return useQuery({**

**queryKey: ["marks student", lessonId],**

**queryFn: () => fetchStudentMarks(lessonId),**

**select: ({ data }) => data,**

**enabled: !!lessonId,**

**});**

**};**

**import { useQuery } from "@tanstack/react-query";**

**import axios from "axios";**

**import { STUDENT\_MARK\_URL, TOKEN } from "../../../shared/const";**

**export const fetchAverageStudentMarks = async (lessonId: string) => {**

**return await axios.get<string>(STUDENT\_MARK\_URL + "/" + lessonId, {**

**headers: {**

**Authorization: TOKEN(),**

**},**

**});**

**};**

**export const useAverageMark = (lessonId: string) => {**

**return useQuery({**

**queryKey: ["average student mark", lessonId],**

**queryFn: () => fetchAverageStudentMarks(lessonId),**

**select: ({ data }) => data,**

**enabled: !!lessonId,**

**});**

**};**

**import { memo } from "react";**

**import { ILesson } from "../interfaces";**

**import { useAverageMark } from "../api/useAverageMark";**

**export const LessonButton = memo(({ onClick, lessons }: props) => {**

**return (**

**<div className="row-start-3 flex flex-col gap-y-8 mt-8" onClick={onClick}>**

**{lessons &&**

**lessons.map((elem) => {**

**const { data } = useAverageMark(elem.id);**

**const mark = data && !isNaN(+data) ? Number(data).toFixed(2) : "";**

**return (**

**<button**

**className="text-left p-[4px] dark:text-white dark:active:text-opacity-35 "**

**key={elem.id}**

**value={elem.id}**

**>**

**{elem.name + " " + mark}**

**</button>**

**);**

**})}**

**</div>**

**);**

**});**

**interface props {**

**onClick: React.MouseEventHandler<HTMLElement>;**

**lessons: ILesson[] | undefined;**

**}**

**import { memo } from "react";**

**export const QuaterButtons = memo(({ onClick }: props) => {**

**return (**

**<nav className="flex ml-20 pt-2 dark:text-white " onClick={onClick}>**

**<button**

**className="active grow dark:text-white dark:active:text-opacity-35"**

**value={1}**

**>**

**1 семестр**

**</button>**

**<button className="grow dark:active:text-opacity-35" value={2}>**

**2 семестр**

**</button>**

**<button className="grow dark:active:text-opacity-35" value={3}>**

**3 семестр**

**</button>**

**<button className="grow dark:active:text-opacity-35" value={4}>**

**4 семестр**

**</button>**

**<button className="grow dark:active:text-opacity-35" value={5}>**

**5 семестр**

**</button>**

**<button className="grow  dark:active:text-opacity-35" value={6}>**

**6 семестр**

**</button>**

**<button className="grow dark:active:text-opacity-35" value={7}>**

**7 семестр**

**</button>**

**<button className="grow dark:active:text-opacity-35" value={8}>**

**8 семестр**

**</button>**

**</nav>**

**);**

**});**

**interface props {**

**onClick: React.MouseEventHandler<HTMLButtonElement>;**

**}**

**export const oddMonths = {**

**"08": "Сентябрь",**

**"09": "Октябрь",**

**"10": "Ноябрь",**

**"11": "Декабрь",**

**"00": "Январь",**

**};**

**export const evenMonths = {**

**"01": "Февраль",**

**"02": "Март",**

**"03": "Апрель",**

**"04": "Май",**

**"05": "Июнь",**

**};**

**import { IMark } from "../interfaces";**

**import { evenMonths, oddMonths } from "./months";**

**export const Quarter = ({ odd, marks }: { odd: boolean; marks: IMark[] }) => {**

**const months = odd ? Object.entries(oddMonths) : Object.entries(evenMonths);**

**const days: number[] = [];**

**for (let i = 1; i <= 31; i++) days.push(i);**

**type TMarks = {**

**[key: string]: number;**

**};**

**const editMarks: TMarks = {};**

**marks.map((data) => {**

**const date = new Date(data.date);**

**const day =**

**date.getDate() < 10 ? "0" + date.getDate() : date.getDate().toString();**

**const month =**

**date.getMonth() < 10 ? "0" + date.getMonth() : date.getMonth().toString();**

**editMarks[month + day] = data.mark;**

**});**

**return (**

**<table className="table-fixed border-2 border-black row-start-3 h-1/2 dark:border-white">**

**<thead>**

**<tr>**

**<th className="w-20"></th>**

**{days.map((value) => {**

**return (**

**<th className="border-[1px] border-red-500  w-6" key={value}>**

**{value}**

**</th>**

**);**

**})}**

**</tr>**

**</thead>**

**<tbody>**

**{months.map((month) => {**

**return (**

**<tr**

**className="border-[1px] border-red-500 text-center"**

**key={month[1]}**

**>**

**<td>{month[1]}</td>**

**{days.map((day) => {**

**const currentDay = day < 10 ? "0" + day : day;**

**return (**

**<td**

**className="border-[1px] border-red-500"**

**key={month[1] + day}**

**>**

**{editMarks.hasOwnProperty(month[0] + currentDay)**

**? editMarks[month[0] + currentDay]**

**: ""}**

**</td>**

**);**

**})}**

**</tr>**

**);**

**})}**

**</tbody>**

**</table>**

**);**

**};**

**import { Quarter } from "./config/quarters";**

**import { useState } from "react";**

**import { useLessons } from "./api/lessons";**

**import { useMarks } from "./api/marks";**

**import { LessonButton, QuaterButtons } from "./components";**

**export const Statement = () => {**

**const [quater, setQuater] = useState(1);**

**const [lesson, setLesson] = useState("");**

**const { data: lessons, isLoading: isLoadingLessons } = useLessons(quater);**

**const { data: marks } = useMarks(lesson);**

**const odd = quater % 2 === 1;**

**const changeQuater = (e: React.MouseEvent<HTMLElement>) => {**

**if (e.target instanceof HTMLButtonElement) {**

**const button = e.target;**

**document.querySelector("nav > .active")?.classList.remove("active");**

**button.classList.add("active");**

**setQuater(+button.value);**

**setLesson("");**

**}**

**};**

**const changeLesson = (e: React.MouseEvent<HTMLElement>) => {**

**if (e.target instanceof HTMLButtonElement) {**

**const button = e.target;**

**document.querySelector("div > .active")?.classList.remove("active");**

**button.classList.add("active");**

**setLesson(button.value);**

**}**

**};**

**return (**

**<div className=" grid grid-cols-statement grid-rows-statement pr-5 dark:text-white">**

**<h1 className=" text-5xl dark:text-white ">Журнал</h1>**

**<QuaterButtons onClick={changeQuater} />**

**{isLoadingLessons ? (**

**<div>Loading</div>**

**) : (**

**<LessonButton lessons={lessons} onClick={changeLesson} />**

**)}**

**{lesson && marks && <Quarter odd={odd} marks={marks} />}**

**</div>**

**);**

**};**

**import React from "react";**

**import ReactDOM from "react-dom/client";**

**import { RouterProvider } from "react-router-dom";**

**import { route } from "./app/routes/routes";**

**import { Context } from "./app/context";**

**import { QueryClient, QueryClientProvider } from "@tanstack/react-query";**

**const queryClient = new QueryClient();**

**ReactDOM.createRoot(document.getElementById("root")!).render(**

**<React.StrictMode>**

**<QueryClientProvider client={queryClient}>**

**<Context children={<RouterProvider router={route}></RouterProvider>} />**

**</QueryClientProvider>**

**</React.StrictMode>**

**);**

**/\*\* @type {import('tailwindcss').Config} \*/**

**export default {**

**content: ["./index.html", "./src/\*\*/\*.{ts,tsx}"],**

**darkMode: "class",**

**mode: "jit",**

**theme: {**

**extend: {**

**gridTemplateColumns: {**

**root: "320px 1fr",**

**statement: "200px 1fr",**

**achievementList: "160px 240px 1fr 320px",**

**organizationCard: "160px 150px 1fr 120px",**

**},**

**gridTemplateRows: {**

**root: "135px 1fr 160px",**

**statement: "50px 60px 1fr",**

**organizationCard: "40px 1fr 60px",**

**},**

**colors: { sub: { 100: "#8392A5" } },**

**backgroundColor: { hr: "#AFAFAF" },**

**},**

**},**

**plugins: [],**

**};**

**ПРИЛОЖЕНИЕ Б**

ПМ студента колледжа

Спецификация

460.ЗМК.00002-01 01

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| |  | | --- | | **Обозначение** | | |  | | --- | | **Наименование** | | |  | | --- | | **Примечание** | |
|  | |  | | --- | | Документация | |  |
| 460.ЗМК.00002-01 12 01 | ПМ студента колледжа |  |
|  | Текст программы |  |
| 460.ЗМК.00002-01 13 01 | ПМ студента колледжа |  |
|  | Описание программы |  |
| 460.ЗМК.00002-01 34 01 | ПМ студента колледжа |  |
|  | Руководство пользователя |  |
| 460.ЗМК.00002-01 81 01 | ПМ студента колледжа |  |
|  | Пояснительная записка |  |

**ПРИЛОЖЕНИЕ B**

**Диск с программным средством**