|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ | | | | | | | | | | | | | | | | | | | | | | |
| **ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ** | | | | | | | | | | | | | | | | | | | | | | |
| **УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ** | | | | | | | | | | | | | | | | | | | | | | |
| **«СЕВАСТОПОЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»** | | | | | | | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | | | | | | | |
| Институт информационных технологий и управления в технических системах | | | | | | | | | | | | | | | | | | | | | | |
| (полное название института) | | | | | | | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | | | | | | | |
| кафедра «Информационные системы» | | | | | | | | | | | | | | | | | | | | | | |
| (полное название кафедры) | | | | | | | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | | | | | | | |
| **Пояснительная записка** | | | | | | | | | | | | | | | | | | | | | | |
| к курсовому проекту  по дисциплине «Веб-технологии» | | | | | | | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | | | | | | | |
| на тему | Система электронных домашних заданий | | | | | | | | | | | | | | | | | | | | | |
| для средней общеобразовательной школы | | | | | | | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | | | | | | | |
| Выполнил: студент | | | | | IV | | | курса, группы: | | | | | | | | ИС/б- 41з | | | | | | |
|  | | | | | | | | | | | | | | | | | | | | | | |
| Направления подготовки (специальности) | | | | | | | | | | | | | | | | 09.03.02 | | | | | | |
|  | | | | | | | | | | | | | | | | | | | | | | |
| Информационные системы и технологии | | | | | | | | | | | | | | | | | | | | | | |
| (код и наименование направления подготовки (специальности)) | | | | | | | | | | | | | | | | | | | | | | |
| профиль (специализация) | | | | | | | | | | Информационные системы | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | | | | | | | |
| **Сербин Александр Александрович** | | | | | | | | | | | | | | | | | | | | | | |
| (фамилия, имя, отчество студента) | | | | | | | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | | | | | | | |
| Руководитель | | | Овчинников А. Л. | | | | | | | | | | | | | | | | | | | |
|  | | | (фамилия, инициалы, степень, звание, должность) | | | | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | | | | | | | |
| Защита « | |  | | » | |  | | | | | | | 20 | |  | | г. | Оценка | | | |  | |
|  | | | | | | |  | | | |  |  | | | | | |  | | |  | | |
| Руководитель | | | | | | | | |  | | | | | | | | | |  |  | | | |
|  | | | | | | | | (подпись) | | | | | | | | | | |  | (инициалы, фамилия) | | | |
|  | | | | | | |  | | | |  |  | | | | | |  | | |  | | |
| Ведущий преподаватель | | | | | | | | |  | | | | | | | | | |  |  | | | |
|  | | | | | | | | (подпись) | | | | | | | | | | |  | (инициалы, фамилия) | | | |
|  | | | | | | | | | | | | | | | | | | | | | | |
| 20 | | | | | | | | | | | | | | 17 | | г. | | | | | | | |

Утверждаю

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

«\_\_\_\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_2017 г.

ТЕХНИЧЕСКОЕ ЗАДАНИЕ

НА РАЗРАБОТКУ WEB-САЙТА ДЛЯ

ГОСУДАРСТВЕННОГО БЮДЖЕТНОГО ОБРАЗОВАТЕЛЬНОГО УЧЕРЕЖДЕНИЯ ГОРОДА СЕВАСТОПОЛЯ

«Средняя общеобразовательная школа»

1. **НАИМЕНОВАНИЕ И ОБЛАСТЬ ПРИМЕНЕНИЯ**

Разработке подлежит WEB-сайт для Государственного бюджетного образовательного учреждения города Севастополя «Средняя общеобразовательная школа № 38 имени Н.В. Челнокова», под рабочим названием: **‘Система электронных домашних заданий для средних общеобразовательных школ’** (конкретная школа была выбрана для примера как школа автора проекта).

Разрабатываемый WEB-сайт является системой электронных домашних заданий для средних общеобразовательных школ.

1. **ОСНОВАНИЕ ДЛЯ РАЗРАБОТКИ**

Основанием для проведения работ по разработке сайта является индивидуальное трудовое соглашение между Государственного бюджетного образовательного учреждения города Севастополя «Средняя общеобразовательная школа № 38 имени Н.В. Челнокова», именуемым в дальнейшем «Заказчик», и Сербиным Александром Александровичем 0914№274221, именуемым в дальнейшем «Исполнитель».

**2.1 Сроки проведения работ:**

Начало работ по созданию WEB-сайта «1» декабря 2017 г.

Окончание работ «1» февраля 2018 г.

1. **ХАРАКТЕРИСТИКА ОБЪЕКТОВ ПРОЕКТИРОВАНИЯ**

Исходными данными для разработки является неформальное описание структуры создаваемого WEB-сайта и набора функций, предоставляемых посетителям (пользовательский интерфейс), и администратору сайта (администраторский интерфейс), а так же графические и текстовые материалы, для размещения в пользовательском интерфейсе сайта. Графические и  
текстовые материалы предоставляются Заказчиком в электронном виде.

**3.1. Структура сайта**

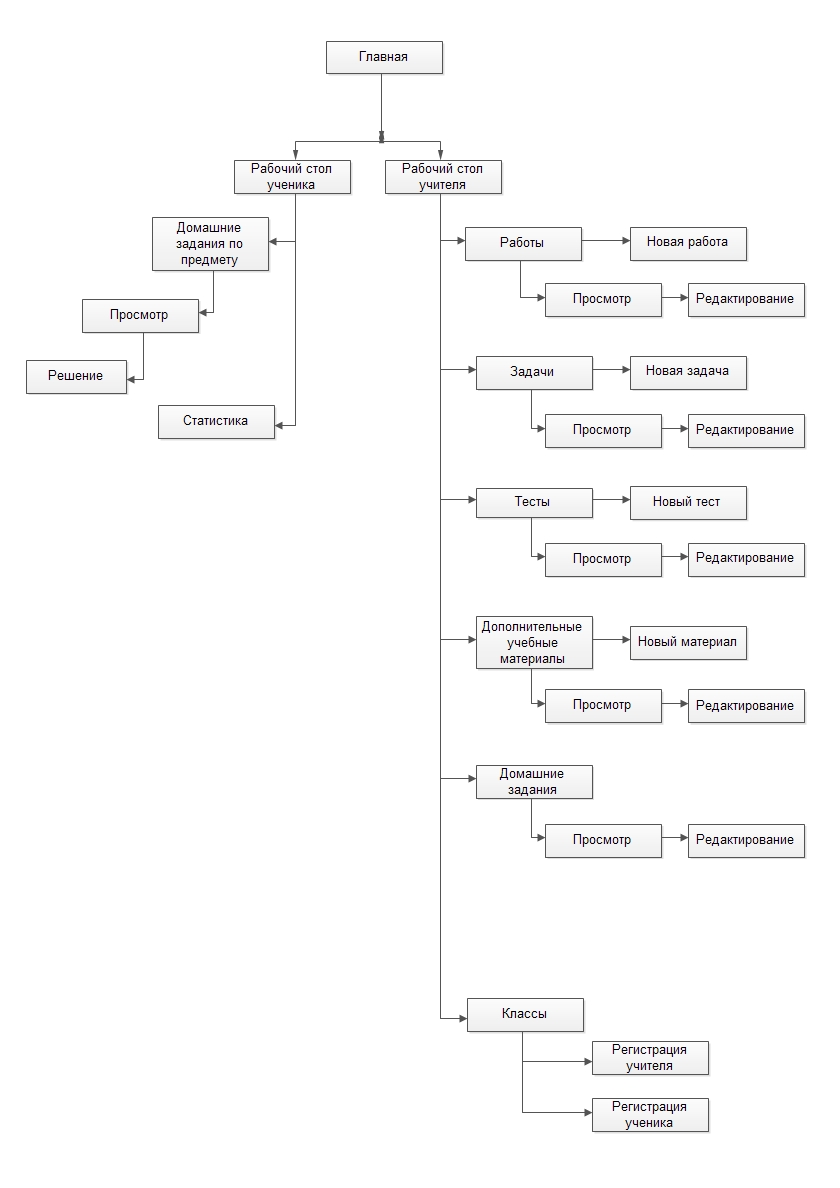
В разрабатываемом WEB-сайте для Государственного бюджетного образовательного учреждения города Севастополя «Средняя общеобразова5тельная школа № 38 имени Н.В. Челнокова» предполагается создание двух независимых интерфейсов пользователей. Интерфейс пользователей - посетителей (учеников) и интерфейс пользователей - администраторов сайта (учителей).

Для пользовательского и административного интерфейса сайта реализуется организация авторизированного доступа. Регистрация пользователей в системе (учеников и учителей) производится в разделе администратора. Пользовательский раздел имеет форму входа, исключая регистрацию.

Информация, добавляемая учителем, сохраняется в базе данных и отображается в соответствующих разделах пользовательского интерфейса WEB-сайта верхней публикацией с указанием даты публикации.

Пользовательский интерфейс содержит статические разделы (разделы, для изменения которых необходимо привлечение разработчика или стороннего WEB-программиста) и динамические разделы (содержимое динамических разделов формируется программно по информации добавленной администратором сайта и хранящейся в базе данных).

Пользовательский интерфейс обеспечивает ученикам возможность просматривать и выполнять свои домашние задания.



**Графическое представление**

**Учитель**

**( Административный раздел )**

**Ученик**

**( Пользовательский раздел )**

**3. 2. Структура веб-страниц**

Ниже будет более подробно описана структура и содержимое пользовательского и администраторского интерфейсов разрабатываемого WEB-сайта. Также описаны определенные аспекты работы сайта.

**3.2.1 Главная страница сайта**

Каждый посетитель сайта, будь то ученик, учитель или гость, при входе на сайт попадает на главную страницу сайта.

Главная страница сайта содержит:

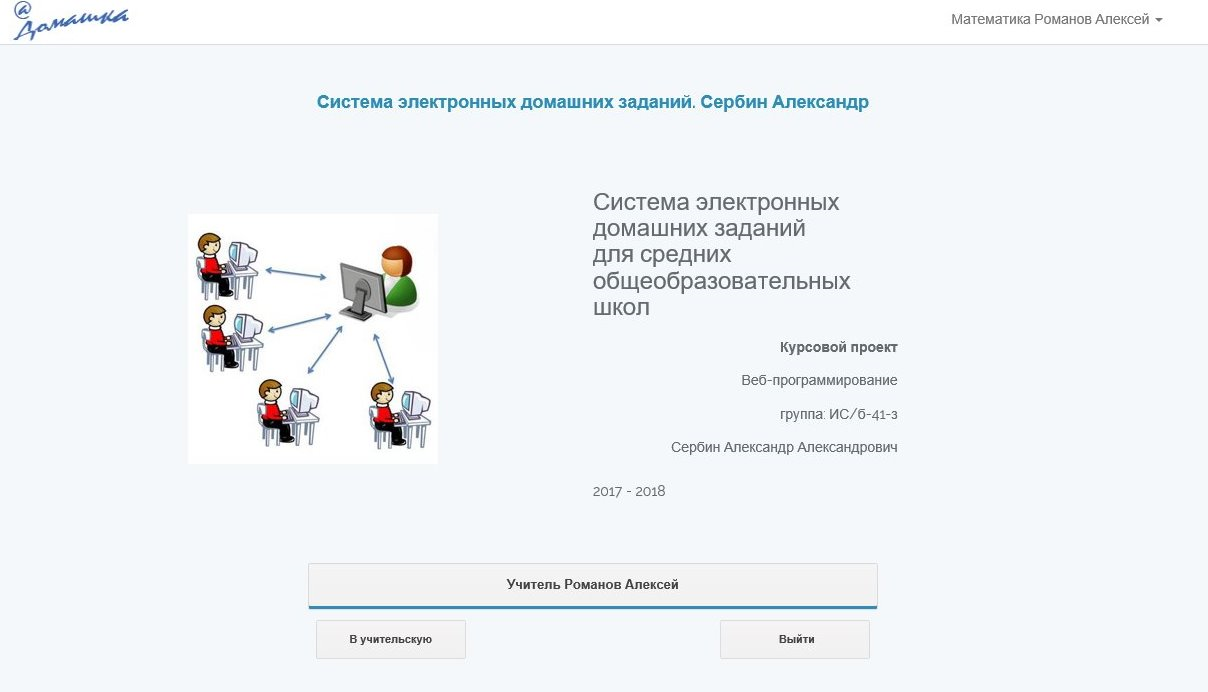
• логотип \*;

• название сайта (ссылка на себя, используется для обновления);

• текстовую информацию о системе и ее деятельности, краткую текстовую информацию, иллюстрацию на тему дистанционное домашнее задание \*;

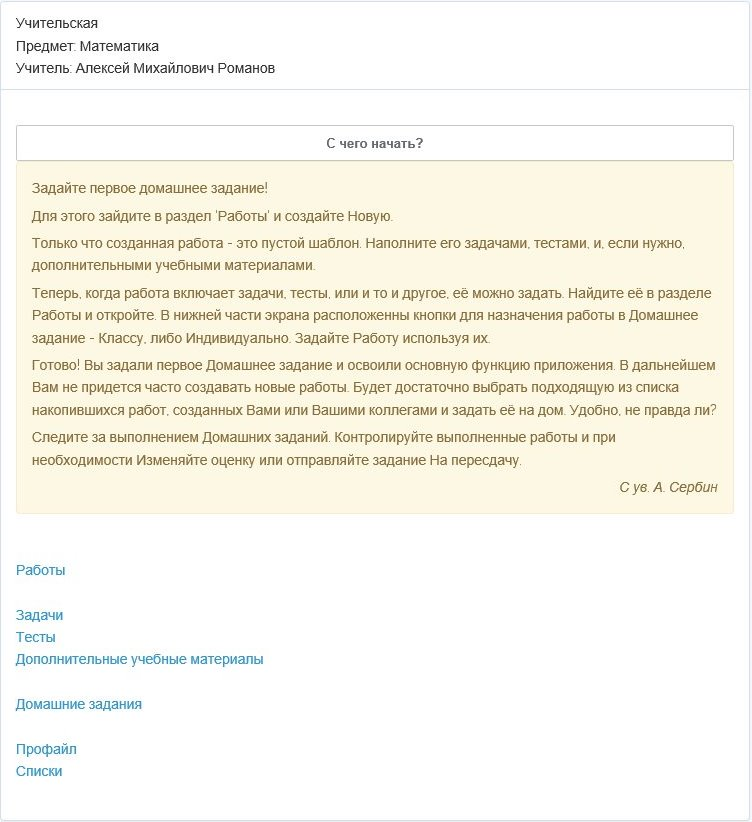
• единая форма входа. Система сама определяет, кто выполнил вход – учитель или ученик. И перенаправляет пользователя в соответствующий раздел.

\* - текстовые, графические и видео материалы предоставляются заказчиком



**3.2.2 Интерфейс администратора**

5.2.2.1 Страница назначения домашнего задания



Страница содержит:

* логотип;
* меню навигации по разделу администратора;
* ссылку на главную страницу (строка навигации)
* список существующих домашних заданий
* ссылка на страницу добавления нового домашнего задания
* форма назначения домашнего задания

Учитель видит список скомпонованных домашних заданий и может выбрать для просмотра и назначить задание.

Учитель может перейти на страницу добавления нового домашнего задания.

5.2.2.2 Страница добавления домашнего задания

Страница содержит:

* логотип;
* меню навигации по разделу администратора;
* ссылку на главную страницу (строка навигации)
* форму компоновки и добавления домашнего задания из существующих в базе данных задач, тестов и дополнительных учебных материалов.
* ссылки на страницы добавления своей задачи, теста и дополнительного учебного материала

Учитель может добавить новое домашнее задание из существующих в базе данных задач, тестов и дополнительных учебных материалов.

Учитель может перейти на страницы добавления своей задачи, теста и дополнительного учебного материала.

3.2.2.3 Страница добавления задачи

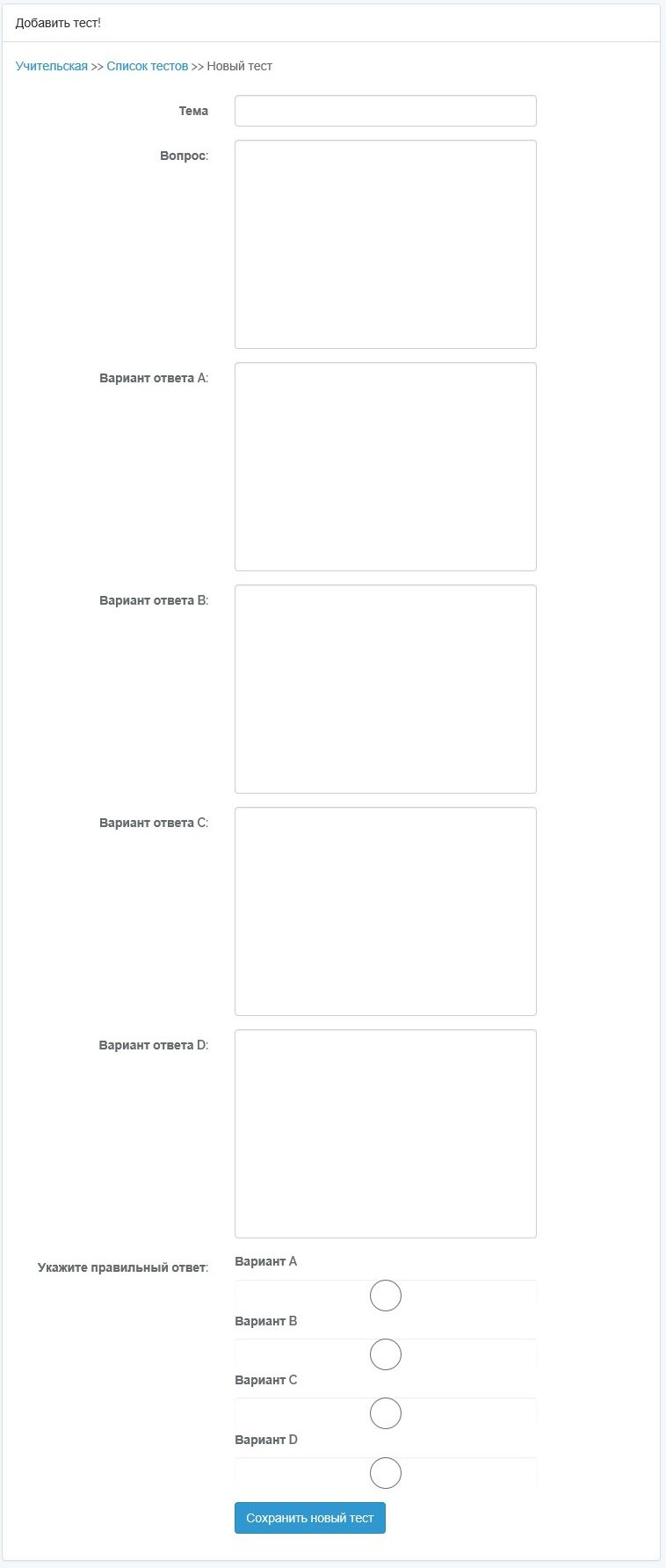
Страница содержит:

* логотип;
* меню навигации по разделу администратора;
* ссылку на главную страницу (строка навигации)
* форму добавления новой задачи в базу данных

3.2.2.4 Страница добавления теста

Страница содержит:

* логотип;
* меню навигации по разделу администратора;
* ссылку на главную страницу (строка навигации)
* форму добавления нового теста в базу данных



3.2.2.5 Страница добавления дополнительного учебного материала

Страница содержит:

* логотип;
* меню навигации по разделу администратора;
* ссылку на главную страницу (строка навигации)
* форму добавления нового дополнительного учебного материала в базу данных

3.2.2.6 Страница просмотра заданных домашних заданий

Страница содержит:

* логотип;
* меню навигации по разделу администратора;
* ссылку на главную страницу (строка навигации)
* календарь с выборкой по числу
* список заданных домашних заданий

3.2.2.7 Страница выполненных домашних работ

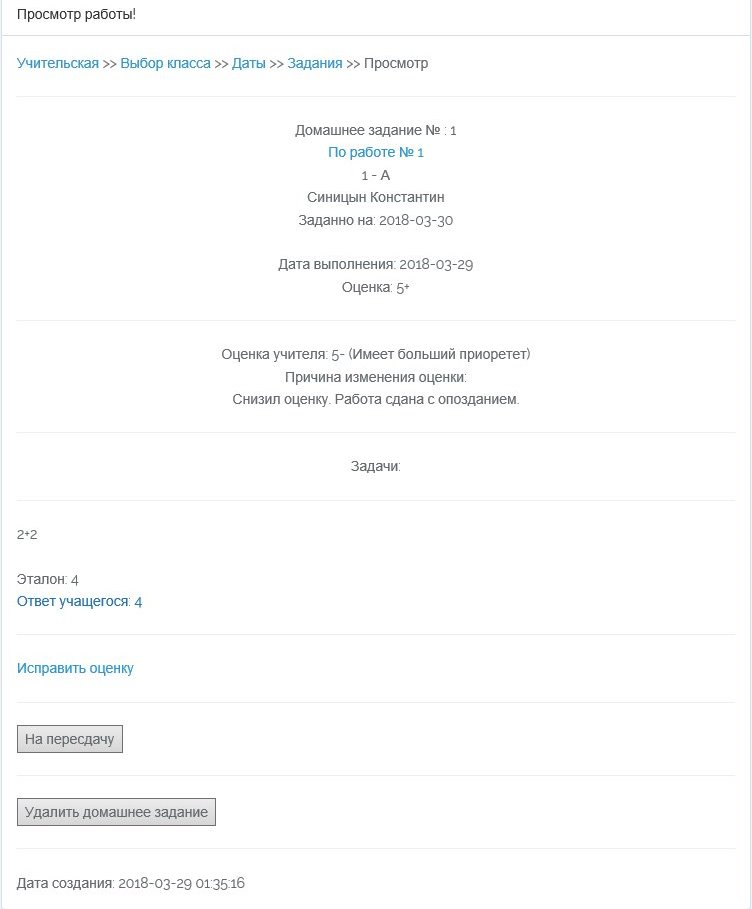
Страница содержит:

* логотип;
* меню навигации по разделу администратора;
* ссылку на главную страницу (строка навигации)
* календарь с выборкой по числу
* список выполненных домашних заданий
* ссылки на каждую выполненную работу

3.2.2.8 Страница контроля выполненной домашней работы

Страница содержит:

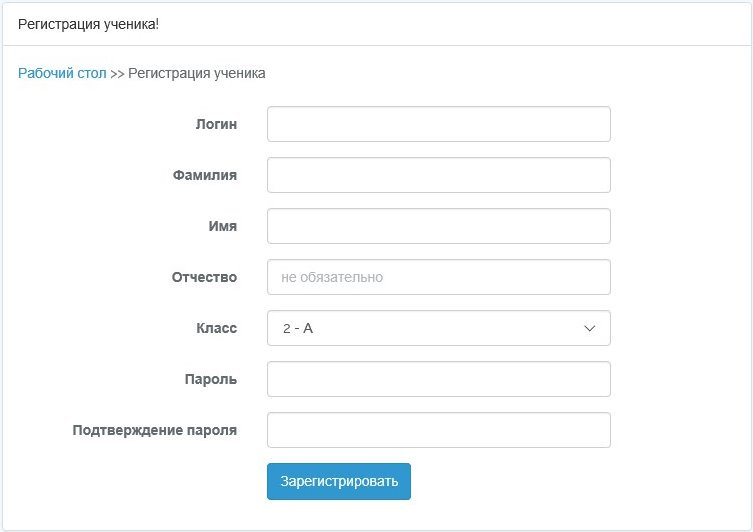
* логотип;
* меню навигации по разделу администратора;
* ссылку на главную страницу (строка навигации)
* задания и данные на них ответы учеником
* контрольные ответы из базы данных
* оценку выставленные программно
* комментарии ученика
* поле для учительской оценки с комментарием
* поле для учительской подписи (отметки checked)



3.2.2.9 Страница для регистрации нового пользователя

Страница содержит:

* логотип;
* меню навигации по разделу администратора;
* ссылку на главную страницу (строка навигации)
* форму для регистрации нового пользователя учителем



**3.2.3 Интерфейс пользователя**

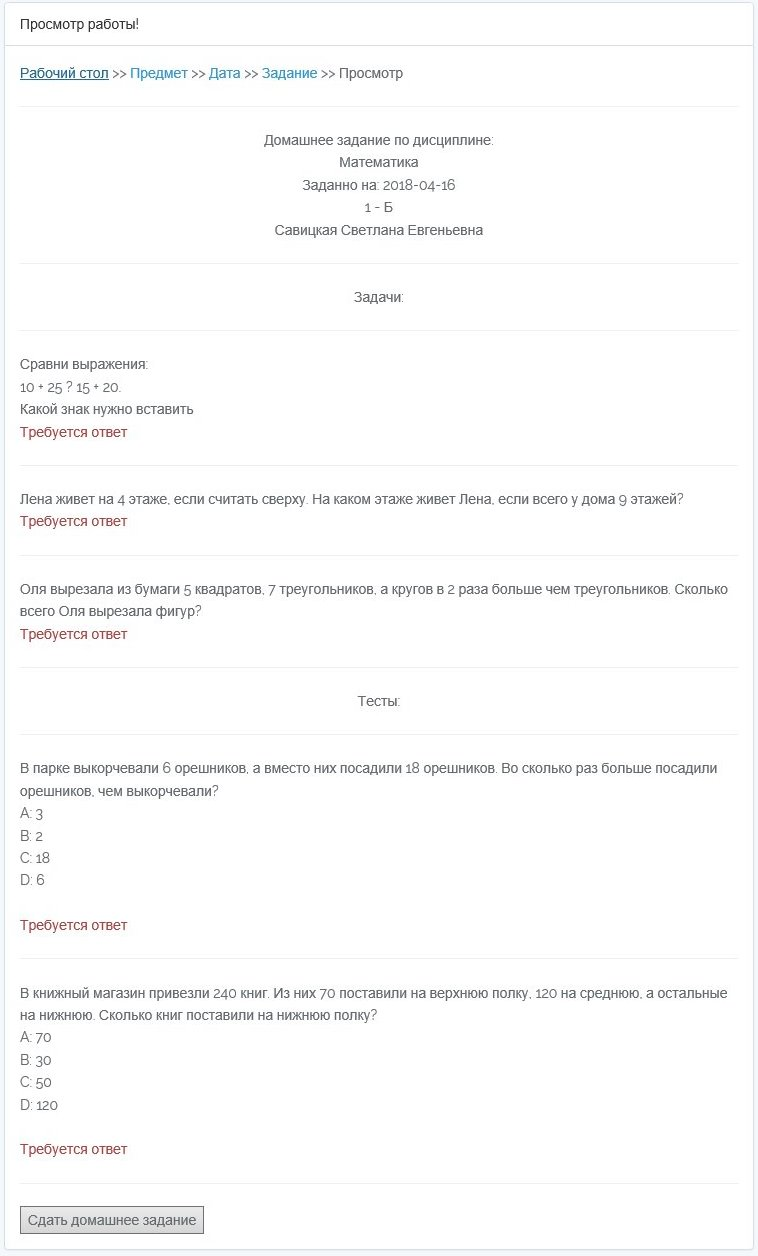
Страница редактирования профиля пользователя

* логотип;
* форму изменения пользовательских данных
* ссылку на страницу изменения пароля

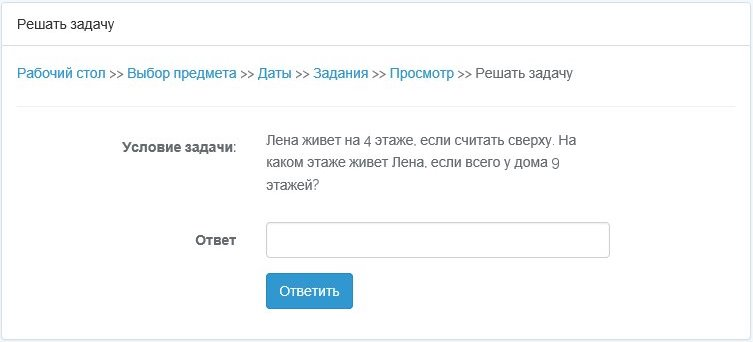
Страница выбора домашнего задания

* логотип;
* меню выбора домешенного задания
* ссылку на домашние задания

Страница решения\просмотра домашнего задания

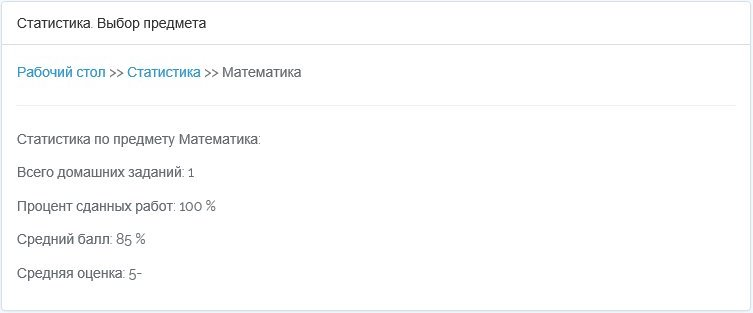


* логотип;
* форму решения домашнего задания (для несданного задания)
* оценку и данные о домашнем задании ( для сданного задания)



Страница информации об успеваемости

* логотип;
* ссылки выбора дисциплины
* суммарная информация об успеваемости по предмету



1. **ОБЩИЕ ТРЕБОВАНИЯ К ОФОРМЛЕНИЮ И ВЕРСТКЕ СТРАНИЦ**

Все страницы разрабатываемого сайта должны быть выдержаны в одном стиле.

Требования к кроссбраузерности: сайт должен отображаться и функционировать идентично в следующих версиях браузеров: Google Chrome 3 и выше, Mozilla Firefox 3.5 и выше, Opera 10 и выше, Explorer 8 и выше.

1. **ТРЕБОВАНИЯ К ПРОГРАММНОМУ ОБЕСПЕЧЕНИЮ СЕРВЕРА**

Для корректного функционирования разрабатываемого сайта необходимо наличие следующих возможностей сервера, на котором будет размещен сайт:

* Функционирование на сервере WEB-сервера Apache-PHP-7 и выше.
* Поддержка MySQL 5.7 и наличие доступа к одной или более базам данных;
* Поддержка PHP 7.1 и выше;
* Возможность загрузки файлов на сервер посредством функций PHP (доступ к временному каталогу сервера).
* Поддержка паролирования директорий средствами Apache;

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/**

**Содержание**

Введение ......................................................................................................................

Анализ предметной области........................................................................................

Постановка задачи ......................................................................................................

Построение диаграммы «сущность-связь» в нотации П. Чена ...............................

Преобразование сетевой структурной схемы в иерархическую..............................

Преобразование иерархической схемы в реляционную...........................................

Построение полной атрибутивной модели в нотации IDEF1X..............................

Разработка физической модели базы данных...........................................................

Реализация базы данных.............................................................................................

Тестирование базы данных.........................................................................................

Разграничение прав доступа.......................................................................................

Заключение...................................................................................................................

Список литературы и информационных ресурсов ..................................................

## ВВЕДЕНИЕ

В наши дни информационные технологии вошли и крепко закрепились в нашей жизни. Информационная грамотность учеников многократно возросла за последние 15 лет.

В современном мире большинство учащихся средних общеобразовательных школ, посещают вебсайты, имеют профайлы в соцсетях, используют поисковые сервисы. В общем, имеют навыки работы с сетью интернет на уровне пользователей. Учеников, не владеющих интернетом, необходимо обучить до уровня пользователь, современный окружающий мир располагает к этому. Желательно интегрировать этот процесс в средние школы.

С другой стороны средние общеобразовательные школы заинтересованы во внедрении информационных технологий в образовательный процесс. Об этом, например, гласит *Программа развития Государственного бюджетного образовательного учреждения города Севастополя «Средняя общеобразовательная школа № 38 имени Н.В. Челнокова» на 2015-2020 годы ()*.

Оба этих вопроса может решить система электронных домашних заданий, внедренная в средние общеобразовательные школы. Кроме того такая система улучшит существующую систему домашних заданий, а именно:

* станет альтернативой старой системы школьных домашних заданий, выполняемых в тетради;
* избавить учителей от работы по проверке домашних заданий, путем программного оценивания, предоставляя время для подготовки образовательных программ;
* предоставит ученикам общеобразовательной школы возможность выполнять домашнюю работу в электронном виде;
* обеспечит возможность добавления информационных и учебных материалов, необходимых для выполнения домашнего задания, учителем;
* обеспечит возможность ученику задать вопрос учителю о выполняемом домашнем задании

Вся информация о домашних заданиях будет храниться в базе данных, что соответствует современным тенденциям развития информационных технологий.

1. **ПОСТАНОВКА ЗАДАЧИ**

Разработать ВЕБ-ПРИЛОЖЕНИЕ «Система электронных домашних заданий для средних общеобразовательных школ» и базу данных к проекту, направленную на поддержание структурированной обработки данных о домашних заданиях и ответов к ним, т.к. главная задача предметной области является организация системы электронных домашних заданий. Так же на основе данных, хранящихся в базе, можно отслеживать нагрузку учителей, успеваемость учеников, а также, регулярность выполнения домашних заданий учениками.

## АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

В проекте рассматривается предметная область – система электронных домашних заданий для средних общеобразовательных школ по одному из предметов.

Средняя школа имеет определенное количество учителей и учащихся по каждому из предметов. Всем им предоставляется доступ к системе электронных домашних заданий, после авторизации в системе. Учителям в административный раздел приложения, а ученикам в пользовательский. Администратор базы данных может зарегистрировать учителя. Авторизированный в приложении учитель может регистрировать учеников.

Домашние задания по предмету проводятся следующим образом:

Учитель добавляет домашнее задание – задачи и \ или тесты и верные ответы на них, путем заполнения формы. Каждое задание сохраняется в базу данных.

Ученик после добавления домашнего задания учителем, может выполнить домашнее задание. Для этого ученик должен зайти в соответствующий раздел и выбрать задание, которое хочет выполнить. После заполнения ответов на задачи в форму и её отправки, будет посчитана оценка за домашнее задание, путем сравнения введенных учеником ответов с эталонными ответами.

Учитель может добавить дополнительные учебные материалы к заданному заданию (дополнительные к школьному учебнику). Дополнительные учебные материалы будут отображаться на странице домашнего задания. Дополнительные учебные материалы сохраняются в базу данных.

Ученик может задать вопрос по выполняемому домашнему заданию в виде комментария к нему. Учитель может ответить на вопрос и \ или добавить дополнительный учебный материал по теме. Ученики видят только свои вопросы учителю. Учитель видит вопросы от всех учеников по заданному им домашнему заданию. Все вопросы и ответы содержатся в базе данных.

1. **ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ**

## Разработка логической модели базы данных

## Построение диаграммы «сущность-связь» в нотации П.Чена

Основные объекты и свойства предметной области:

*Предмет, Учитель, Ученик, Задачи, Тесты, Работа для ДЗ, Домашнее задание, Результаты, Дополнительные учебные материалы, Комментарии;*

Свойства объектов:

***Предмет*** (№ предмета, название, описание);

***Учитель***(№ учителя, ФИО, адрес, номер телефона, квалификация, дата трудоустройства);

***Ученик***(№ ученика, ФИО, класс, адрес, контактная информация родителей);

***Домашнее задание*** (№ домашнего задания, Тема, Дата добавления, № учителя)

***Задача*** (№ задачи, тема, задание, ответ, автор);

***Тест*** (№ теста, тема, задание, вариант ответа A, вариант ответа B, вариант ответа C, вариант ответа D, номер верного варианта ответа, автор);

***Домашняя работа*** (№ домашней работы, № ученика, текст работы, дата);

***Результаты*** (№ результата, № домашней работы, оценка);

***Дополнительные учебные материалы*** (№ дополнительных учебных материалов, иллюстрация, текст, автор);

***Комментарии*** (№ комментария, автор, текст, дата);

Связи между объектами:

М

М

1

1

Имеем связь 1:1 (один-к-одному), можем избавиться от лишнего объекта, объединив сущности: Домашняя работа и Результат, в одну:

***Домашняя работа*** (№ домашней работы, № ученика, текст работы, оценка, дата);

М

М

М

М

М

М

Имеем связь М:М (многие-ко-многим), избавимся введением промежуточной сущности.

1

М

М

1

1

М

1

М

1

М

М

1

1

М

М

1

Новые сущности:

***Блок задач*** (№ домашнего задания, № задачи);

***Блок тестов*** (№ домашнего задания, № теста);

***Назначенное задание*** (№ назначенного задания, № ученика, № домашнего задания, дата);

1

1

Имеем связь 1:1 (один-к-одному), можем избавиться от лишнего объекта, объединив сущности: Назначенное задание и Домашняя работа, в одну:

***Домашняя работа*** (№ домашней работы, № домашнего задания, № ученика, текст работы, оценка, дата назначения, дата выполнения);

1

М

М

1

М

М

1

М

1

М

1

М

1

Связи сущностей 1:М (один-ко-многим) нас устраивают.

Структурная схема предметной области (Сетевая структура)

Ученик

Предмет

Учитель

Работа

Домашнее задание

Дополнительные учебные материалы

Задача

Тест

Комментарии

Рис 1. Структурная модель данных (сетевая)

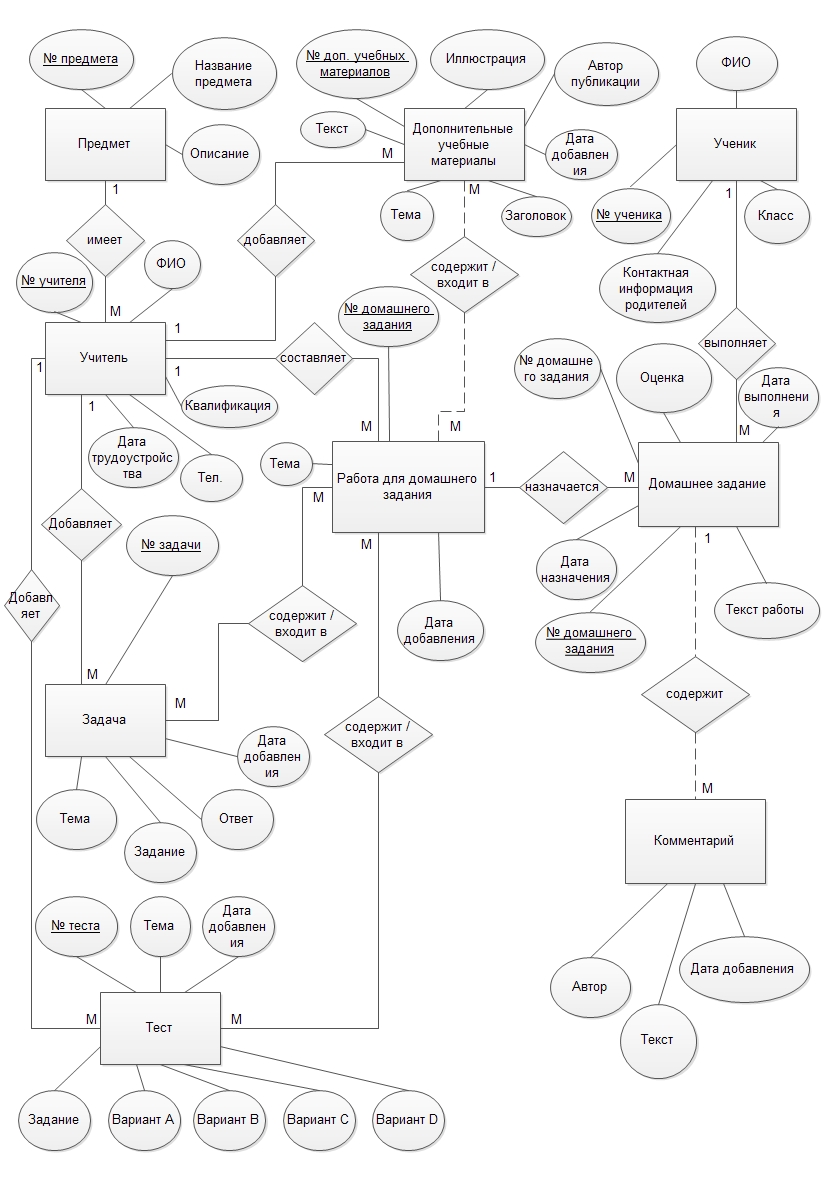


Рис 2 Диаграмма «сущность-связь» в нотации П.Чена

ER-диаграмма

**3.2. Преобразование сетевой структурной схемы в иерархическую**

Ученик

Предмет

Учитель

Работа

Тест

Задача

Дополнительные учебные материалы

Блок дополнительных учебных материалов

Блок тестов

Блок задач

Домашнее задание - Работа

Домашнее задание - Ученик

Комментарии

Рис 3. Структурная модель данных. Иерархическая (Древовидная).

**3.3. Преобразование иерархической схемы в реляционную – НОРМАЛИЗАЦИЯ**

Нормализация – процесс преобразования древовидной структуры в реляционную.

**Первая нормальная форма (1НФ).**

Первичные ключи и атомарность.

**Атомарность:**

В сущностях ***Учитель*** и ***Ученик*** есть атрибут ФИО, он не атормарен. Разобьем атрибут ФИО на 3 атрибута: Имя, Фамилия, Отчество.

**Первичные ключи отношений:**

***Предметы***( № предмета, название, описание );

***Учителя*** ( № учителя, № предмета, имя, фамилия, отчество, номер телефона, квалификация, дата трудоустройства );

***Задачи*** ( № задачи, № учителя, тема, задание, ответ);

***Тесты*** ( № теста, № учителя, тема, задание, вариант ответа A, вариант ответа B, вариант ответа C, вариант ответа D, номер верного варианта ответа);

***Дополнительные учебные материалы*** (№ дополнительного учебного материала, № учителя, тема, заголовок, иллюстрация, текст, автор публикации, дата добавления);

***Работы для домашних заданий*** ( № работы, № учителя, тема, дата добавления);

***Блоки тестов*** (№ работы, № теста);

***Блоки задач*** (№ работы, № задачи);

***Блоки дополнительных учебных материалов*** (№ работы, № дополнительного учебного материала);

***Домашние задания - работы*** (№ домашнего задания, № работы, дата на когда задано);

***Ученики*** ( № ученика, имя, фамилия, отчество, класс, контактная информация родителей );

***Домашние задания - ученики*** (№ домашнего задания, № ученика, текст работы, дата выполнения, оценка, комментарий учителя, оценка учителя);

***Комментарии*** ( № комментария, № домашнего задания, автор, текст, дата добавления);

**Вторая нормальная форма (2НФ):**

Если все возможные ключи отношения содержат по одному атрибуту, то все отношения находится во 2 НФ.

**Третья нормальная форма (3НФ):**

Уберем транзитивные зависимости из сущности *Дополнительные учебные материалы*:

Атрибут *автор публикации* зависит от атрибута *номер учителя*. Мы должны были бы вынести атрибут в сущность *Учитель*, но заметили что это, по сути, тавтология и от атрибута *автор публикации* необходимо избавиться. Удалим его.

Получили следующее отношение:

**Дополнительные учебные материалы** (№ дополнительного учебного материала, № учителя, тема, заголовок, иллюстрация, текст, дата добавления);

**Четвертая нормальная форма (4НФ):**

Многозначные зависимости отсутствуют, значит, отношения в 4НФ.

В итоге нормализации имеем следующие сущности:

***Предметы*** ( № предмета, название, описание )

***Пользователи*** ( № пользователя, логин, email, пароль )

***Учителя*** ( № учителя, № предмета, имя, фамилия, отчество, номер телефона, квалификация, дата трудоустройства )

***Задачи*** ( № задачи, № учителя, тема, задание, ответ, дата добавления, дата редактирования)

***Тесты*** ( № теста, № учителя, тема, задание, вариант ответа A, вариант ответа B, вариант ответа C, вариант ответа D, номер верного варианта ответа, дата добавления, дата редактирования)

***Дополнительные учебные материалы*** (№ дополнительного учебного материала, № учителя, тема, заголовок, иллюстрация, текст, дата добавления, дата редактирования)

***Работы для домашних заданий*** ( № работы, № учителя, тема, дата добавления);

***Блоки тестов*** (№ работы, № теста)

***Блоки задач*** (№ работы, № задачи)

***Блоки дополнительных учебных материалов*** (№ работы, № дополнительного учебного материала)

***Ученики*** ( № ученика, имя, фамилия, отчество, класс, контактная информация родителей )

***Домашние задания*** (№ домашнего задания, № ученика, №работы, дата на когда задано, текст работы, дата выполнения, оценка, комментарий учителя, оценка учителя);

***Комментарии***( № комментария, № домашнего задания, автор, текст, дата добавления)

**3.3 Построение полной атрибутивной модели в нотации IDEF1X (нормализация отношений до третьей или четвертой нормальной формы)**

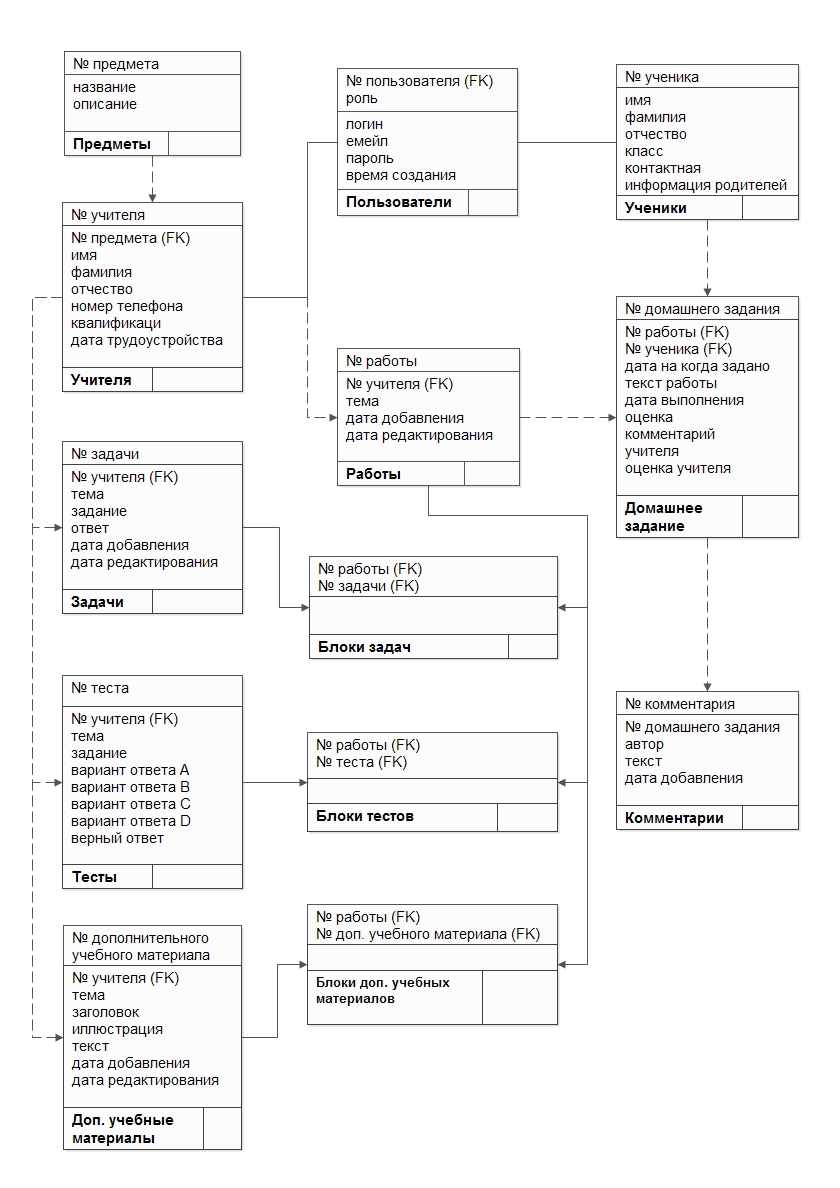
**Разработка физической модели базы данных**

Рис.4. IDEF1X модель

Обозначения:

связь один-ко-многим

связь один-к-одному

неидентифицирующая связь

**3.4 Выбор аппаратной и программной платформы и реализация БД**

Разработанную логическую схему базы данных реализуем с использованием бесплатной СУБД. В качестве продукта для разработки и администрирования выбранной баз данных будем MySQL Workbench, позволяющего максимально систематизировать и автоматизировать все этапы разработки программного обеспечения.

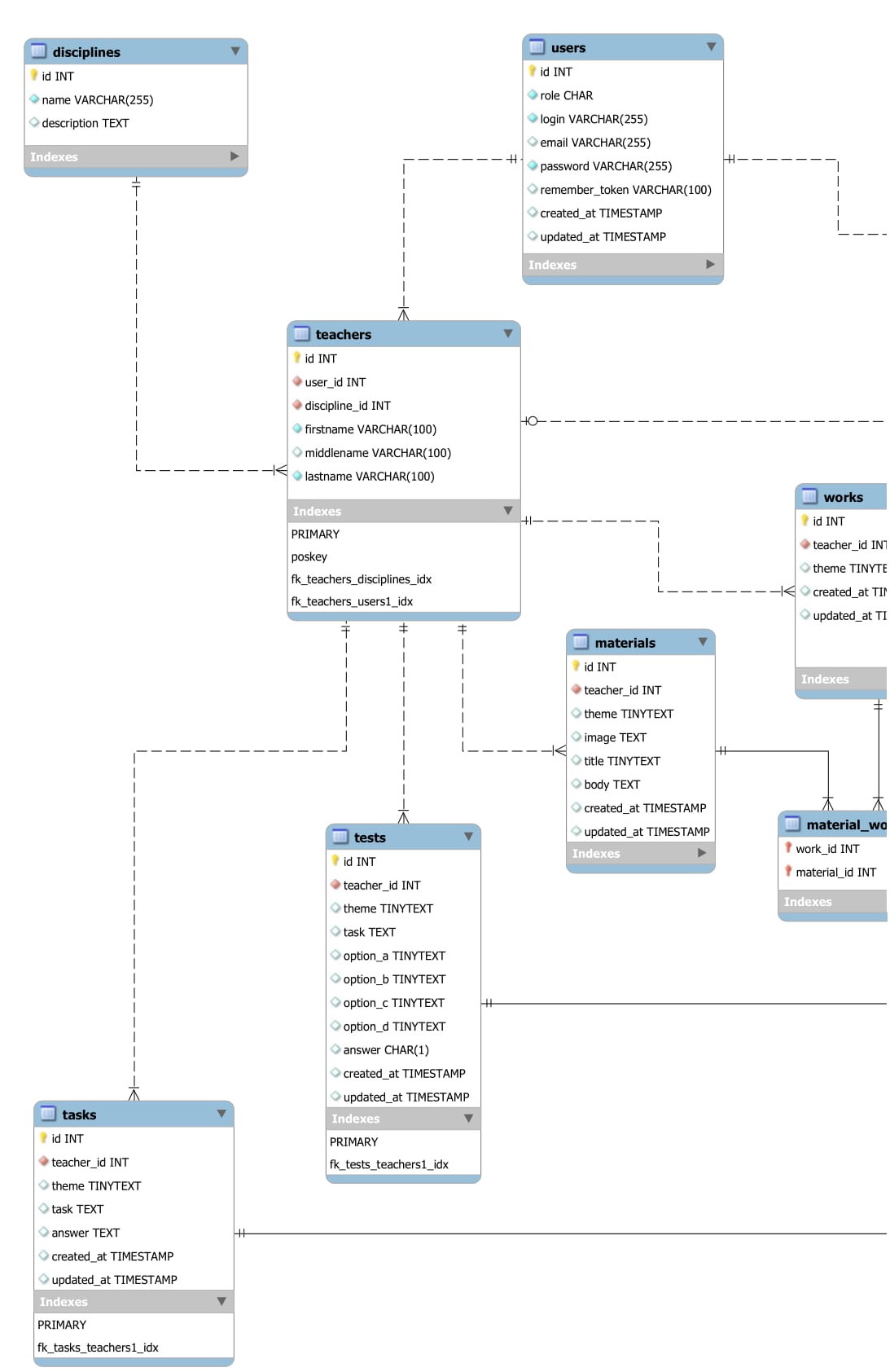


Рис. 5. Физическая модель

базы данных

**3.5 Реализация базы данных**

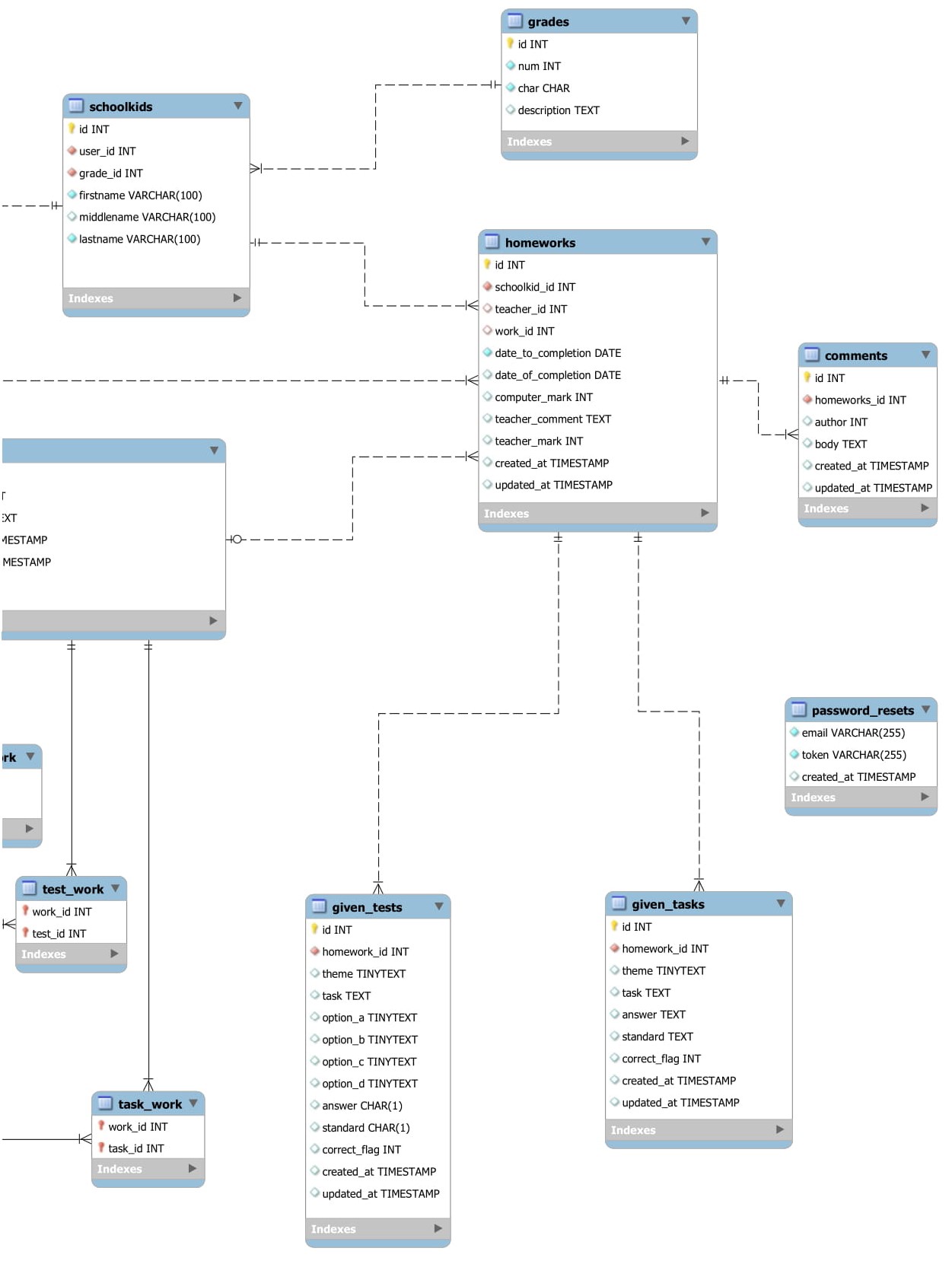


Рис. 5. Физическая модель

базы данных

**3. 6. Итог по разработке базы данных**

Была разработана база данных для проекта ‘**Система электронных домашних заданий для средней общеобразовательной школы**’, направленная на поддержание структурированной обработки данных:

* О задачах и тестах, а так же учебных материалах создаваемых учителями.
* О скомпонованных из задач, тестов и дополнительных учебных материалов работах, которые впоследствии, возможно, назначать как домашние задания ученикам.
* Об успеваемости учеников и загруженности преподавателей.

1. **ВЫБОР ИНСТРУМЕНТАЛЬНЫХ СРЕДСТВ**

Разрабатываемый ВЕБ – сайт, будет написан на языке **PHP 7.1** с использованием фреймворка Laravel 5.4. База данных будет создана в **MySQL 5.7**. Динамический контент веб страниц и валидация на стороне клиента будет выполнена на языке **JavaScript** с использованием фреймворка **JQuery** и **JQuery UI**.

1. **РАЗРАБОТКА ПРОГРАММНЫХ МОДУЛЕЙ ПРИЛОЖЕНИЯ.**

Система электронных домашних задания реализована в трех разделах. Раздел администратора, пользователя и учительский раздел.

Первый программный модуль, с которым мы сталкиваемся при работе с СЭДЗ это *Аутентификация пользователей*.

**Аутентификация пользователей** реализуется с помощью встроенных модулей аутентификации фреймворка Laravel 5.4.

Далее в работе приложение определяет роль пользователя. И каждый пользователь, в зависимости от роли, получает доступ к определенному разделу сайта.

Ролей три:

* Учитель
* Ученик
* Администратор

Для проверки роли в каждом разделе используется посредник базового контроллера. Для администратора он называется *AdminController*, для учителя *TeacherController*, для ученика *UserController*.

Логика контроллера посредника для проверки роли:

**class** AdminController **extends** Controller  
{  
 **protected $user**;  
  
 */\*\*  
 \* Create a new controller instance.  
 \*  
 \** ***@return*** *void  
 \*/* **public function** \_\_construct()  
 {  
 $this->middleware(**function** ($request, $next) {  
 **if** (Auth::**check**()) {  
 $this->**user** = Auth::**user**();  
 **if** (Auth::*user*()->**role** == **'a'**) {  
 **return** $next($request);  
 } **else** {  
 **return** redirect(**'/'**);  
 }  
 } **else** {  
 **return** redirect(**'login'**);  
 }  
 });  
 }  
}

Важно отметить, что вся маршрутизация сайта описана в файле *routes/web.php* следующим образом, пример:

Route::*get*(**'/desktop'**, **'User\DesktopController@index'**);  
  
Route::*get*(**'/homeworks'**, **'User\Homework\UsrChoiceController@index'**);  
  
Route::*get*(**'/homeworks/{discipline\_id}'**, **'User\Homework\UsrChoiceController@show\_dates'**)->name(**'showUsrDates'**);  
  
Route::*get*(**'/homeworks/{discipline\_id}/{date}'**, **'User\Homework\UsrChoiceController@show\_homeworks'**)->name(**'showHomeworks'**);  
  
Route::*resource*(**'/homeworks/{discipline\_id}/{date}/hometask'**, **'User\HometaskController'**);  
  
Route::*resource*(**'/homeworks/{discipline\_id}/{date}/hometask/{hwrk\_id}/given\_task'**, **'User\GivenTaskController'**);

Если пользователь определен как администратор, то он имеет доступ к администраторскому разделу сайта. Там ему доступен следующий функционал:

* Добавить новый предмет
* Добавить новый класс
* Зарегистрировать Учителя
* Зарегистрировать Ученика

Для добавления нового предмета и класса используются формы на странице вида и передаются в соответствующие контроллеры ресурсов методом POST, в функцию *store.* Контроллер извлекает данные из запроса *Request $request* в массив, и передает массив в соответствующую модель. Модель проверяет уникальность предмета или класса в базе данных и выполняет валидацию средствами фреймворка Laravel класса Validator. Пример:

$validator = Validator::*make*($data,  
 [  
 **'num'** => **'required|numeric'**,  
 **'char'** => [  
 **'required'**,  
 **'size:1'**,  
 **'regex:/[А-Я]/'** ],  
  
 ],  
 [  
 **'num.required'** => **'Необходимо указать цифру класса'**,  
 **'num.numeric'** => **'Цифра класса должна быть цифрой'**,  
 **'char.required'** => **'Необходимо указать букву для класса'**,  
 **'char.size'** => **'Необходимо указать 1 букву кириллицей'**,  
 **'char.regex'** => **'Букву необходимо указать Кириллицей'**,  
 ]);

При успехе валидации модель добавляет новую запись в базу данных.

Контроллер обрабатывает ответ из модели – успех или ошибка. И посылает сообщение исходя из результата в представление.

Надо сказать, что данная схема добавления записей в базу данных справедлива для всех сущностей приложения. Поэтому к сохранению данных в БД, валидации и ответах контроллера возвращаться не будем.

Если пользователь авторизован как учитель, то ему доступен следующий функционал:

* Создание, редактирование, изменение, удаление, просмотр - работ, задач, тестов, дополнительных учебных материалов
* Проверка заданных и выполненных домашних заданий.
* Просмотр списков учащихся
* Изменение собственного профиля

Прежде всего, для учителей, в приложении реализован механизм разграничения прав доступа. Например, учитель не может просматривать и задавать работу по другим предметам. Или не может удалять и изменять задачу которую создал не сам. Для этого используется встроенный функционал политик Laravel 5.4. Пример политики Given\_taskPolicy:

*/\*\*  
 \* Determine whether the user can update the givenTask.  
 \*  
 \** ***@param*** *\App\User $user  
 \** ***@param*** *\App\Given\_task $givenTask  
 \** ***@param*** *\App\Homework $homework  
 \** ***@return*** *mixed  
 \*/***public function** update(User $user, Given\_task $givenTask, Homework $homework)  
{  
 **if** ($homework->**computer\_mark**) {  
 $pass = **false**;  
 } **else** {  
 ($homework->**teacher\_mark**) ? $pass = **false** : $pass = **true**;  
 }  
 **return** (  
 ($user->**schoolkid**->**id** === $givenTask->**homework**->**schoolkid\_id**)  
 &&  
 ($pass)  
 );  
}

Создание, редактирование, изменение, удаление и просмотр данных выполняется с помощью контроллеров ресурсов фреймворка Laravel 5.4. Пример контроллера ресурсов MaterialController:

**<?php  
  
namespace** App\Http\Controllers\Teacher;  
  
**use** App\Http\Controllers\TeacherController;  
**use** Illuminate\Http\Request;  
**use** App\Material;  
**use** App\Teacher;  
**use** App\Work;  
  
**class** MaterialController **extends** TeacherController  
{  
 */\*\*  
 \* Display a listing of the resource.  
 \*  
 \** ***@return*** *\Illuminate\Http\Response  
 \*/* **public function** index(Material $material, Teacher $author)  
 {  
 Подготовка данных;

Вернуть представление с данными;  
 }  
  
 */\*\*  
 \* Show the form for creating a new resource.  
 \*  
 \** ***@return*** *\Illuminate\Http\Response  
 \*/* **public function** create()  
 {  
 Вернуть представление с формой добавления записи в БД;  
 }  
  
 */\*\*  
 \* Store a newly created resource in storage.  
 \*  
 \** ***@param*** *\Illuminate\Http\Request $request  
 \** ***@return*** *\Illuminate\Http\Response  
 \*/* **public function** store(Request $request, Material $material)  
 {  
 Подготовка данных;

Передача данных в соответствующую модель

и получение ответа;

Вернуть представление с ответом;  
 }  
  
 */\*\*  
 \* Display the specified resource.  
 \*  
 \** ***@param*** *int $id  
 \** ***@return*** *\Illuminate\Http\Response  
 \*/* **public function** show($id)  
 {  
 Подготовка данных;

Проверка прав доступа;

Передача данных в соответствующую модель

и получение ответа;

Вернуть представление с ответом;  
  
 }  
  
 */\*\*  
 \* Show the form for editing the specified resource.  
 \*  
 \** ***@param*** *int $id  
 \** ***@return*** *\Illuminate\Http\Response  
 \*/* **public function** edit($id)  
 {  
 Подготовка данных;

Проверка прав доступа;

Вернуть представление с формой редактирование или

сообщением об ошибке;  
  
 }  
  
 */\*\*  
 \* Update the specified resource in storage.  
 \*  
 \** ***@param*** *\Illuminate\Http\Request $request  
 \** ***@param*** *int $id  
 \** ***@return*** *\Illuminate\Http\Response  
 \*/* **public function** update(Request $request, $id)  
 {  
 Подготовка данных;

Проверка прав доступа;

Передача данных в соответствующую модель

и получение ответа;

Вернуть представление с ответом;  
 }  
  
 */\*\*  
 \* Remove the specified resource from storage.  
 \*  
 \** ***@param*** *int $id  
 \** ***@return*** *\Illuminate\Http\Response  
 \*/* **public function** destroy($id)  
 {  
 Подготовка данных;

Проверка прав доступа;

Передача данных в соответствующую модель

и получение ответа;

Вернуть представление с ответом;  
 }  
}

Пример модели Material:

**<?php  
  
namespace** App;  
**use** Illuminate\Database\Eloquent\Model;  
**use** Validator;  
  
**class** Material **extends** Model  
{  
 **protected $fillable** = [  
 Перечисление полей открытых для редактирования;];  
  
 **public function** getAllPaginated()  
 {  
 Получить и вернуть все записи из соответствующей таблицы

базы данных упакованные в постраничную навигацию;  
 }  
  
 **public function** getDisciplineMaterialsPaginated($discipline\_id)  
 {  
 Получить и вернуть записи по указанному

предмету из соответствующей таблицы

базы данных упакованные в постраничную навигацию;  
 }  
  
 **public function** getOne($id)  
 {  
 Получить и вернуть одну запись из таблицы базы данных.  
 }  
  
 **public function** store($data, $teacher\_id)  
 {  
 Выполнить валидацию входных данных;

Сохранить новую запись в базу данных;

Вернуть ответ;  
 }  
  
 **public function** edit($id, $teacher\_id, $data)  
 {  
 Выполнить валидацию входных данных;

Сохранить новую запись в базу данных;

Вернуть ответ;  
 }  
  
  
 **public function** kill($id)  
 {  
 Найти и удалить запись из Базы данных;  
 }

На этих принципах работы с данными построено все приложение «Система электронных домашних заданий для средних общеобразовательных школ»

Все описанное выше справедливо и для раздела ученика.

**ЗАКЛЮЧЕНИЕ**

В ходе выполнения курсового проекта было разработано ВЕБ-ПРИЛОЖЕНИЕ «Система электронных домашних заданий для средних общеобразовательных школ» и база данных к проекту, направленная на поддержание структурированной обработки данных о домашних заданиях и ответов к ним, т.к. главная задача предметной области является организация системы электронных домашних заданий. Так же на основе данных, хранящихся в базе, можно отслеживать нагрузку учителей, успеваемость учеников, а также, регулярность выполнения домашних заданий учениками.

**Список литературы и информационных ресурсов**

1. Barker R. CASE-Method. Entity-Relationship Modelling. Copyright Oracle

Corporation UK Limited, Addison-Wesley Publishing Co., 1990.

1. IDEF1 Information Modeling Method. Описание. Режим доступа –

http://www.idef.com/IDEF1.htm.

1. IDEF1X Data Modeling Method. Описание. Режим доступа –

http://www.idef.com/IDEF1x.htm.

1. Integration Definition For Information Modeling (IDEF1X). Режим доступа

– http://www.idef.ru/documents/Idef1x.pdf.

1. Laravel 5.4, Official Documentation - https://laravel.com/docs/5.4
2. «Программа развития на 2015-2020 гг.» средней школы №38 им. Челнокова.

**Приложение А. Исходный код программы. Основные модули.**

Файл routes/web.php

**<?php***/\*  
|--------------------------------------------------------------------------  
| Web Routes  
|--------------------------------------------------------------------------  
|  
| Here is where you can register web routes for your application. These  
| routes are loaded by the RouteServiceProvider within a group which  
| contains the "web" middleware group. Now create something great!  
|  
\*/*Route::*get*(**'/'**, **'IndexController@index'**);  
  
Route::*get*(**'/test'**, **'IndexController@index'**)->middleware(**'auth'**);  
  
Auth::*routes*();  
  
  
*//USER*Route::*get*(**'/desktop'**, **'User\DesktopController@index'**);  
  
Route::*get*(**'/homeworks'**, **'User\Homework\UsrChoiceController@index'**);  
  
Route::*get*(**'/homeworks/{discipline\_id}'**, **'User\Homework\UsrChoiceController@show\_dates'**)->name(**'showUsrDates'**);  
  
Route::*get*(**'/homeworks/{discipline\_id}/{date}'**, **'User\Homework\UsrChoiceController@show\_homeworks'**)->name(**'showHomeworks'**);  
  
Route::*resource*(**'/homeworks/{discipline\_id}/{date}/hometask'**, **'User\HometaskController'**);  
  
Route::*resource*(**'/homeworks/{discipline\_id}/{date}/hometask/{hwrk\_id}/given\_task'**, **'User\GivenTaskController'**);  
  
Route::*resource*(**'/homeworks/{discipline\_id}/{date}/hometask/{hwrk\_id}/given\_test'**, **'User\GivenTestController'**);  
  
Route::*post*(**'/homeworks/pass\_homework'**, **'User\Homework\PassController@pass'**)->name(**'passHomework'**);  
  
  
Route::*get*(**'/statistics'**, **'User\StatisticsController@index'**);  
  
Route::*get*(**'/statistics/{discipline\_id}'**, **'User\StatisticsController@show'**)->name(**'showDisciplineStat'**);  
  
*//User Account*Route::*get*(**'/kid-account/change\_password'**, **'User\ChangePasswordController@index'**);  
  
Route::*post*(**'/kid-account/change\_password'**, **'User\ChangePasswordController@change'**)->name(**'userPasswordChange'**);  
  
Route::*resource*(**'/kid-account'**, **'User\AccountController'**);  
  
  
*//TEACHER*Route::*get*(**'/teacher'**, **'Teacher\DesktopController@index'**);  
  
*//Set-Unset*Route::*get*(**'/teacher/works/{id}/set'**, **'Teacher\Set\SetWorkController@index'**)->name(**'indexSet'**);  
  
Route::*post*(**'/teacher/works/set'**, **'Teacher\Set\SetWorkController@set'**)->name(**'setWork'**);  
  
Route::*get*(**'/teacher/works/{id}/individually'**, **'Teacher\Set\SetIndividuallyController@index'**)->name(**'indexIndividually'**);  
  
Route::*post*(**'/teacher/works/individually'**, **'Teacher\Set\SetIndividuallyController@set'**)->name(**'setIndividually'**);  
  
Route::*post*(**'/teacher/set/task'**, **'Teacher\Set\SetTaskController'**)->name(**'setTask'**);  
  
Route::*post*(**'/teacher/unset/task'**, **'Teacher\Set\UnsetTaskController'**)->name(**'unsetTask'**);  
  
Route::*post*(**'/teacher/set/test'**, **'Teacher\Set\SetTestController'**)->name(**'setTest'**);  
  
Route::*post*(**'/teacher/unset/test'**, **'Teacher\Set\UnsetTestController'**)->name(**'unsetTest'**);  
  
Route::*post*(**'/teacher/set/material'**, **'Teacher\Set\SetMaterialController'**)->name(**'setMaterial'**);  
  
Route::*post*(**'/teacher/unset/material/'**, **'Teacher\Set\UnsetMaterialController'**)->name(**'unsetMaterial'**);  
  
*// For works*Route::*resource*(**'/teacher/tasks'**, **'Teacher\TaskController'**);  
  
Route::*resource*(**'/teacher/tests'**, **'Teacher\TestController'**);  
  
Route::*resource*(**'/teacher/materials'**, **'Teacher\MaterialController'**);  
  
Route::*resource*(**'/teacher/works'**, **'Teacher\WorksController'**);  
  
  
*// Teachers Homework*Route::*get*(**'/teacher/homeworks'**, **'Teacher\Homework\TchrChoiceController@index'**);  
  
Route::*post*(**'/teacher/repass'**, **'Teacher\Homework\RePassController@repass'**)->name(**'toRePass'**);  
  
Route::*get*(**'/teacher/homeworks/{grade\_id}'**, **'Teacher\Homework\TchrChoiceController@show\_dates'**)->name(**'showTcrDates'**);  
  
Route::*get*(**'/teacher/homeworks/{grade\_id}/{date}'**, **'Teacher\Homework\TchrChoiceController@show\_homeworks'**)->name(**'showHwKids'**);  
  
Route::*resource*(**'/teacher/homeworks/{grade\_id}/{date}/homework'**, **'Teacher\HomeworkController'**);  
  
*// Teachers Lists*Route::*get*(**'/teacher/lists'**, **'Teacher\ListsController@index'**);  
  
Route::*get*(**'/teacher/lists/grades'**, **'Teacher\ListsController@grade\_list'**);  
  
Route::*get*(**'/teacher/lists/grades/view'**, **'Teacher\ListsController@grade\_show\_by\_get'**)->name(**'gradeShowGet'**);  
  
Route::*get*(**'/teacher/lists/grades/{id}'**, **'Teacher\ListsController@grade\_show'**)->name(**'gradeShow'**);  
  
Route::*get*(**'/teacher/lists/disciplines'**, **'Teacher\ListsController@discipline\_list'**);  
  
Route::*get*(**'/teacher/lists/disciplines/view'**, **'Teacher\ListsController@discipline\_show\_by\_get'**)->name(**'disciplineShowGet'**);  
  
Route::*get*(**'/teacher/lists/disciplines/{id}'**, **'Teacher\ListsController@discipline\_show'**)->name(**'disciplineShow'**);  
  
Route::*get*(**'/teacher/lists/schoolkids'**, **'Teacher\ListsController@schoolkid\_list'**);  
  
Route::*get*(**'/teacher/lists/teachers'**, **'Teacher\ListsController@teacher\_list'**);  
  
  
*//Teacher Account*Route::*get*(**'/teacher/account/change\_password'**, **'Teacher\ChangePasswordController@index'**);  
  
Route::*post*(**'/teacher/account/change\_password'**, **'Teacher\ChangePasswordController@change'**)->name(**'teacherPasswordChange'**);  
  
Route::*resource*(**'/teacher/account'**, **'Teacher\AccountController'**);  
  
  
*//ADMIN*Route::*get*(**'/admin'**, **'Admin\DesktopController@index'**);  
  
*// Register*Route::*resource*(**'/admin/teach'**, **'Admin\PedagogueController'**);  
  
Route::*resource*(**'/admin/kid'**, **'Admin\SchoolkidController'**);  
  
Route::*get*(**'/admin/register/redirect-to-main-page-after-registration'**, **'Admin\InfoPagesController@afterReg'**);  
  
*// Create*Route::*resource*(**'/admin/grade'**, **'Admin\GradeController'**);  
  
Route::*resource*(**'/admin/discipline'**, **'Admin\DisciplineController'**);  
  
*// Admin Lists*Route::*get*(**'/admin/lists'**, **'Admin\ListsController@index'**);  
  
Route::*get*(**'/admin/lists/grades'**, **'Admin\ListsController@grade\_list'**);  
  
Route::*get*(**'/admin/lists/disciplines'**, **'Admin\ListsController@discipline\_list'**);  
  
Route::*get*(**'/admin/lists/schoolkids'**, **'Admin\ListsController@schoolkid\_list'**);  
  
Route::*get*(**'/admin/lists/teachers'**, **'Admin\ListsController@teacher\_list'**);

Представление index.php

**@extends('layouts.app')  
  
@section('content')** <**div class="container"**>  
  
 <**section**>  
 **@if (**session(**'status'**)**)** <**div class="alert alert-success"**>  
 {{ session(**'status'**) }}  
 </**div**>  
 **@endif  
 @include('common.errors')** *{{--****todo обернуть в bootstrap шаблон как на остальных страничках****--}}* <**div class="main\_lable"**>  
 <**a href="/"**>  
 Система электронных домашних заданий. Сербин Александр  
 </**a**>  
 </**div**>  
 <**table class="hello\_table"**>  
 <**tbody**>  
 <**tr**>  
 <**td**>  
 <**img src="/public/img/dhw.jpg" width="250" alt=""**/>  
 </**td**>  
 <**td class="lr\_column"**>  
 <**h3**>  
 Система электронных домашних заданий<**br**>  
 для средних общеобразовательных школ<**br**>  
 </**h3**>  
 <**div class="to\_right"**>  
 <**p**>  
 <**strong**>  
 Курсовой проект<**br**>  
 </**strong**>  
 </**p**>  
 <**p**>  
 Веб-программирование  
 </**p**>  
 <**p**>  
 группа: ИС/б-41-з  
 </**p**>  
 <**p**>  
 Сербин Александр Александрович  
 </**p**>  
 </**div**>  
 <**p**>  
 2017 - 2018  
 </**p**>  
 </**td**>  
 </**tr**>  
 </**tbody**>  
 </**table**>  
  
 **@if(**$user**)** <**div class="enter\_area"**>  
 <**div class="hello\_label"**>  
 **@if (**$user->**role** == **'t')** Учитель  
 {{ $user->**teacher**->**lastname** }}  
 {{ $user->**teacher**->**firstname** }}  
 **@elseif (**$user->**role** == **'s')** Учащийся  
 {{ $user->**schoolkid**->**lastname** }}  
 {{ $user->**schoolkid**->**firstname** }}  
 **@elseif (**$user->**role** == **'a')** Администратор  
 {{$user->**login**}}  
 **@endif** </**div**>  
  
 <**table class="hello\_table"**>  
 <**tbody**>  
 <**tr**>  
 <**td**>  
 **@if (**$user->**role** == **'t')** <**a href="/teacher" style="text-decoration**: **none" class="enter\_button"**>В  
 учительскую</**a**>  
 **@elseif (**$user->**role** == **'s')** <**a href="/desktop" style="text-decoration**: **none" class="enter\_button"**>За уроки</**a**>  
 **@elseif (**$user->**role** == **'a')** <**a href="/admin" style="text-decoration**: **none" class="enter\_button"**>Войти</**a**>  
 **@endif** </**td**>  
 <**td**>  
 {!! Form::*open*([**'url'**=>route(**'logout'**)]) !!}  
 {!! Form::*submit*(**' Выйти '**,[**'class'**=>**'enter\_button'**]) !!}  
 {!! Form::*close*() !!}  
 </**td**>  
 </**tr**>  
 </**tbody**>  
 </**table**>  
 </**div**>  
 **@else** <**div class="enter\_area"**>  
 <**div class="enter\_block"**>  
 <**a href="login" style="text-decoration**: **none"**>  
 <**div id="mybutton"**> Войти в программу</**div**>  
 </**a**>  
 </**div**>  
 </**div**>  
 **@endif** </**section**>  
 </**div**>  
**@endsection**

Admin/skhoolkids.blade.php

**@extends('layouts.app')  
  
@section('content')** <**div class="container"**>  
 <**div class="row"**>  
 <**div class="col-md-8 col-md-offset-2"**>  
 <**div class="panel panel-default"**>  
 <**div class="panel-heading"**>Учащиеся</**div**>  
  
 <**div class="panel-body"**>  
 **@if (**session(**'status'**)**)** <**div class="alert alert-success"**>  
 {{ session(**'status'**) }}  
 </**div**>  
 **@endif  
 @include('common.errors')** <**a href="/admin"**>Рабочий стол</**a**> >>  
 <**a href="/admin/lists"**>Списки</**a**> >>  
 Учащиеся  
 <**hr**>  
 <**br**>  
  
 **@foreach(**$schoolkids **as** $schoolkid**)** <**a href="**{{route(**'kid.show'**, $schoolkid->**id**)}}**"**>  
 {{$schoolkid->**lastname** }}  
 {{$schoolkid->**firstname**}}  
 {{$schoolkid->**middlename**}}  
 </**a**>  
 ,  
 <**a href="**{{route(**'grade.show'**, $schoolkid->**grade**->**id**)}}**"**>  
 {{$schoolkid->**grade**->**num**}}  
 -  
 {{$schoolkid->**grade**->**char**}}  
 </**a**>  
 <**hr**>  
  
 **@endforeach** {{$schoolkids->links()}}  
  
 </**div**>  
 </**div**>  
 </**div**>  
 </**div**>  
 </**div**>  
**@endsection**

Admin/skhoolkids/create.blade.php

**@extends('layouts.app')  
  
@section('content')** <**div class="container"**>  
 <**div class="row"**>  
 <**div class="col-md-8 col-md-offset-2"**>  
 <**div class="panel panel-default"**>  
 <**div class="panel-heading"**>Регистрация ученика!</**div**>  
  
 <**div class="panel-body"**>  
 **@if (**session(**'status'**)**)** <**div class="alert alert-success"**>  
 {{ session(**'status'**) }}  
 </**div**>  
 **@endif** <**a href="/admin"**>Рабочий стол</**a**> >>  
 Регистрация ученика  
 <**br**><**br**>  
  
 <**form class="form-horizontal" method="POST" action="**{{ route(**'register'**) }}**"**>  
 {{ csrf\_field() }}  
  
 *{{-- indicate the role 's' - schoolkid, 't' - teacher, 'a' - admin --}}* {!! Form::*hidden*(**'role'**, **'s'**) !!}  
  
 <**div class="form-group**{{ $errors->has(**'login'**) ? **' has-error'** : **''** }}**"**>  
 <**label for="login" class="col-md-4 control-label"**>Логин</**label**>  
  
 <**div class="col-md-6"**>  
 <**input id="login" type="text" class="form-control" name="login"  
 value="**{{ old(**'login'**) }}**" required autofocus**>  
  
 **@if (**$errors->has(**'login'**)**)** <**span class="help-block"**>  
 <**strong**>{{ $errors->first(**'login'**) }}</**strong**>  
 </**span**>  
 **@endif** </**div**>  
 </**div**>  
  
 <**div class="form-group**{{ $errors->has(**'lastname'**) ? **' has-error'** : **''** }}**"**>  
 <**label for="lastname" class="col-md-4 control-label"**>Фамилия</**label**>  
  
 <**div class="col-md-6"**>  
 <**input id="lastname" type="text" class="form-control" name="lastname"  
 value="**{{ old(**'lastname'**) }}**" required**>  
  
 **@if (**$errors->has(**'lastname'**)**)** <**span class="help-block"**>  
 <**strong**>{{ $errors->first(**'lastname'**) }}</**strong**>  
 </**span**>  
 **@endif** </**div**>  
 </**div**>  
  
 <**div class="form-group**{{ $errors->has(**'firstname'**) ? **' has-error'** : **''** }}**"**>  
 <**label for="firstname" class="col-md-4 control-label"**>Имя</**label**>  
  
 <**div class="col-md-6"**>  
 <**input id="firstname" type="text" class="form-control" name="firstname"  
 value="**{{ old(**'firstname'**) }}**" required**>  
  
 **@if (**$errors->has(**'firstname'**)**)** <**span class="help-block"**>  
 <**strong**>{{ $errors->first(**'firstname'**) }}</**strong**>  
 </**span**>  
 **@endif** </**div**>  
 </**div**>  
  
 <**div class="form-group**{{ $errors->has(**'middlename'**) ? **' has-error'** : **''** }}**"**>  
 <**label for="middlename" class="col-md-4 control-label"**>Отчество</**label**>  
  
 <**div class="col-md-6"**>  
 <**input id="middlename" type="text" class="form-control" name="middlename"  
 value="**{{ old(**'middlename'**) }}**" placeholder="не обязательно"**>  
  
 **@if (**$errors->has(**'middlename'**)**)** <**span class="help-block"**>  
 <**strong**>{{ $errors->first(**'middlename'**) }}</**strong**>  
 </**span**>  
 **@endif** </**div**>  
 </**div**>  
  
 <**div class="form-group**{{ $errors->has(**'grade'**) ? **' has-error'** : **''** }}**"**>  
 <**label for="grade" class="col-md-4 control-label"**>Класс</**label**>  
  
 <**div class="col-md-6"**>  
 <**select name="grade" class="form-control" id="grade" required**>  
 <**option selected="selected" value=""**>Выберите класс...</**option**>  
 **@foreach(**$grades **as** $grade**)** <**option value=**{{$grade->**id**}} >  
 {{$grade->**num**}}  
 -  
 {{$grade->**char**}}  
 </**option**>  
 **@endforeach** </**select**>  
  
 **@if (**$errors->has(**'grade'**)**)** <**span class="help-block"**>  
 <**strong**>{{ $errors->first(**'grade'**) }}</**strong**>  
 </**span**>  
 **@endif** </**div**>  
 </**div**>  
  
 <**div class="form-group**{{ $errors->has(**'password'**) ? **' has-error'** : **''** }}**"**>  
 <**label for="password" class="col-md-4 control-label"**>Пароль</**label**>  
  
 <**div class="col-md-6"**>  
 <**input id="password" type="password" class="form-control" name="password" required**>  
  
 **@if (**$errors->has(**'password'**)**)** <**span class="help-block"**>  
 <**strong**>{{ $errors->first(**'password'**) }}</**strong**>  
 </**span**>  
 **@endif** </**div**>  
 </**div**>  
  
 <**div class="form-group"**>  
 <**label for="password-confirm" class="col-md-4 control-label"**>Подтверждение  
 пароля</**label**>  
  
 <**div class="col-md-6"**>  
 <**input id="password-confirm" type="password" class="form-control"  
 name="password\_confirmation" required**>  
 </**div**>  
 </**div**>  
  
 <**div class="form-group"**>  
 <**div class="col-md-6 col-md-offset-4"**>  
 <**button type="submit" class="btn btn-primary"**>  
 Зарегистрировать  
 </**button**>  
 </**div**>  
 </**div**>  
 </**form**>  
 </**div**>  
 </**div**>  
 </**div**>  
 </**div**>  
 </**div**>  
**@endsection**

Admin/disciplines/cerate.php

**@extends('layouts.app')  
  
@section('content')** <**div class="container"**>  
 <**div class="row"**>  
 <**div class="col-md-8 col-md-offset-2"**>  
 <**div class="panel panel-default"**>  
 <**div class="panel-heading"**>Редактирование предмета</**div**>  
  
 <**div class="panel-body"**>  
 **@if (**session(**'status'**)**)** <**div class="alert alert-success"**>  
 {{ session(**'status'**) }}  
 </**div**>  
 **@endif** <**a href="/admin"**>Рабочий стол</**a**> >>  
 <**a href="/admin/lists"**>Списки</**a**> >>  
 <**a href="/admin/lists/disciplines"**>Передметы</**a**> >>  
 <**a href="**{{route(**'discipline.show'**, $discipline->**id**)}}**"**>  
 {{$discipline->**name**}}  
 </**a**> >>  
 Редактирование  
 <**br**><**br**>  
  
 <**form class="form-horizontal" method="POST"  
 action="**{{ route(**'discipline.update'**, $discipline->**id**) }}**"**>  
 {{ csrf\_field() }}  
 {{ method\_field(**'PUT'**) }}  
  
 <**input type="hidden" name="id" value="**{{$discipline->**id**}}**"**>  
  
 <**div class="form-group**{{ $errors->has(**'name'**) ? **' has-error'** : **''** }}**"**>  
 <**label for="name" class="col-md-4 control-label"**>Название предмета</**label**>  
  
 <**div class="col-md-6"**>  
 <**input id="name" type="text" class="form-control" name="name"  
 value="**{{$discipline->**name**}}**" required autofocus**>  
  
 **@if (**$errors->has(**'name'**)**)** <**span class="help-block"**>  
 <**strong**>{{ $errors->first(**'name'**) }}</**strong**>  
 </**span**>  
 **@endif** </**div**>  
 </**div**>  
  
 <**div class="form-group**{{ $errors->has(**'description'**) ? **' has-error'** : **''** }}**"**>  
 <**label for="description" class="col-md-4 control-label"**>Описание</**label**>  
  
 <**div class="col-md-6"**>  
 <**textarea id="description" class="form-control" name="description"  
 placeholder="Не обязательно"**>{{$discipline->**description**}}  
 </**textarea**>  
  
 **@if (**$errors->has(**'description'**)**)** <**span class="help-block"**>  
 <**strong**>{{ $errors->first(**'description'**) }}</**strong**>  
 </**span**>  
 **@endif** </**div**>  
 </**div**>  
  
 <**div class="form-group"**>  
 <**div class="col-md-6 col-md-offset-4"**>  
 <**button type="submit" class="btn btn-primary"**>  
 Редактировать  
 </**button**>  
 </**div**>  
 </**div**>  
 </**form**>  
 </**div**>  
 </**div**>  
 </**div**>  
 </**div**>  
 </**div**>  
**@endsection**

Teacher/account/change\_password.blade.php

**@extends('layouts.app')  
  
@section('content')** <**div class="container"**>  
 <**div class="row"**>  
 <**div class="col-md-8 col-md-offset-2"**>  
 <**div class="panel panel-default"**>  
 <**div class="panel-heading"**>Домашние задания</**div**>  
  
 <**div class="panel-body"**>  
 **@if (**session(**'status'**)**)** <**div class="alert alert-success"**>  
 {{ session(**'status'**) }}  
 </**div**>  
 **@endif  
 @include('common.errors')** <**a href="/teacher"**>Учительская</**a**> >>  
 <**a href="/teacher/homeworks"**> Выбор класса </**a**> >>  
 Даты  
 <**br**>  
 <**hr**>  
  
  
 **@foreach(**$dates **as** $date**)** <**a href="**{{route(**'showHwKids'**, [$grade\_id, $date->**date\_to\_completion**])}}**"**>  
 {{ $date->**date\_to\_completion** }}  
 </**a**>  
 <**hr**>  
  
 **@endforeach** {{$dates->links()}}  
  
  
 </**div**>  
 </**div**>  
 </**div**>  
 </**div**>  
 </**div**>  
**@endsection**

Teacher/account/homework/calendar.blade.php

**@extends('layouts.app')  
  
@section('content')** <**div class="container"**>  
 <**div class="row"**>  
 <**div class="col-md-8 col-md-offset-2"**>  
 <**div class="panel panel-default"**>  
 <**div class="panel-heading"**>Домашние задания</**div**>  
  
 <**div class="panel-body"**>  
 **@if (**session(**'status'**)**)** <**div class="alert alert-success"**>  
 {{ session(**'status'**) }}  
 </**div**>  
 **@endif  
 @include('common.errors')** <**a href="/teacher"**>Учительская</**a**> >>  
 <**a href="/teacher/homeworks"**> Выбор класса </**a**> >>  
 <**a href="**{{route(**'showTcrDates'**, $grade\_id)}}**"**> Даты </**a**> >>  
 Задания  
 <**br**>  
 <**hr**>  
  
  
 **@foreach(**$homeworks **as** $homework**)** <**a href="**{{route(**'homework.show'**, [  
 **'grade\_id'** => $grade\_id,  
 **'date'** => $date,  
 **'id'** => $homework->**id** ])}}**"**>  
 {{$homework->**schoolkid**->**lastname** }}  
 {{$homework->**schoolkid**->**firstname**}}  
 {{$homework->**schoolkid**->**middlename**}}  
 </**a**>  
 (  
 <**a href="**{{route(**'works.show'**, [**'id'**=>$homework->**work\_id**])}}**"**>  
 Работа N:  
 {{$homework->**work\_id**}}  
 </**a**>  
 )  
 **@if (**$homework->**computer\_mark)  
 &#10004;  
 @endif** <**hr**>  
 **@endforeach** </**div**>  
 </**div**>  
 </**div**>  
 </**div**>  
 </**div**>  
**@endsection**

Teacher/account/homework/show.blade.php

**@extends('layouts.app')  
  
@section('content')** <**div class="container"**>  
 <**div class="row"**>  
 <**div class="col-md-8 col-md-offset-2"**>  
 <**div class="panel panel-default"**>  
 <**div class="panel-heading"**>  
 Просмотр работы!  
 </**div**>  
 <**div class="panel-body"**>  
 **@if (**session(**'status'**)**)** <**div class="alert alert-success"**>  
 {{ session(**'status'**) }}  
 </**div**>  
 **@endif** <**a href="/teacher"**>Учительская</**a**> >>  
 <**a href="/teacher/homeworks"**> Выбор класса </**a**> >>  
 <**a href="**{{route(**'showTcrDates'**, $grade\_id)}}**"**> Даты </**a**> >>  
 <**a href="**{{route(**'showHwKids'**, [$grade\_id, $date])}}**"**> Задания </**a**> >>  
 Просмотр  
 <**br**>  
 <**hr**>  
  
 **@if (**!$homework->**computer\_mark)** <**p class="works\_show\_block\_title"**>  
 Домашнее задание № : {{ $homework->**id** }}<**br**>  
 <**a href="**{{route(**'works.show'**, [**'id'**=>$homework->**work**->**id**])}}**"**>  
 По работе № {{$homework->**work**->**id**}}  
 </**a**><**br**>  
 {{$homework->**schoolkid**->**grade**->**num**}}  
 -  
 {{$homework->**schoolkid**->**grade**->**char**}}<**br**>  
 {{$homework->**schoolkid**->**lastname**}}  
 {{$homework->**schoolkid**->**firstname**}}  
 {{$homework->**schoolkid**->**middlename**}}<**br**>  
 Заданно на: {{$homework->**date\_to\_completion**}}<**br**><**br**>  
  
 Домашнее задание еще не сдано  
 <**br**>  
 <**hr**>  
  
 **@if (**!**empty**($homework\_content[**'given\_tasks'**])**)** <**p class="works\_show\_block\_title"**>Задачи:</**p**>  
 <**hr**>  
 **@foreach(**$homework\_content[**'given\_tasks'**] **as** $task**)** {!! *nl2br*($task->**task**)!!}<**br**><**br**>  
  
 Эталон: {{$task->**standard**}}  
 <**hr**>  
 **@endforeach  
 @endif  
  
 @if (**!**empty**($homework\_content[**'given\_tests'**])**)** <**p class="works\_show\_block\_title"**>Тесты:</**p**>  
 <**hr**>  
 **@foreach(**$homework\_content[**'given\_tests'**] **as** $test**)** {!! *nl2br*($test->**task**) !!}<**br**>  
 A: {!! *nl2br*($test->**option\_a**) !!}<**br**>  
 B: {!! *nl2br*($test->**option\_b**) !!}<**br**>  
 C: {!! *nl2br*($test->**option\_c**) !!}<**br**>  
 D: {!! *nl2br*($test->**option\_d**) !!}<**br**><**br**>  
  
 Эталон: {{$test->**standard**}}  
 <**hr**>  
 **@endforeach  
 @endif  
  
 @if (**!**empty**($homework\_content[**'materials'**])**)** <**p class="works\_show\_block\_title"**>Дополнительные учебные материалы:</**p**>  
 <**hr**>  
 **@foreach(**$homework\_content[**'materials'**] **as** $material**)** <**a href="**{{route(**'materials.show'**, [**'id'**=>$material->**id**])}}**"**>  
 Дополнительный учебный материал № : {{ $material->**id** }}  
 </**a**><**br**>  
 Тема: {{$material->**theme**}}<**br**>  
 Заголовок: {{ $material->**title** }}<**br**>  
 Изображение: {{$material->**image**}}<**br**>  
 {!! *nl2br*($material->**body**) !!}<**br**>  
 <**hr**>  
 **@endforeach  
 @endif  
  
 @else** <**p class="works\_show\_block\_title"**>Домашнее задание № : {{ $homework->**id** }}<**br**>  
 <**a href="**{{route(**'works.show'**, [**'id'**=>$homework->**work**->**id**])}}**"**>  
 По работе № {{$homework->**work**->**id**}}  
 </**a**><**br**>  
 {{$homework->**schoolkid**->**grade**->**num**}}  
 -  
 {{$homework->**schoolkid**->**grade**->**char**}}<**br**>  
 {{$homework->**schoolkid**->**lastname**}}  
 {{$homework->**schoolkid**->**firstname**}}  
 {{$homework->**schoolkid**->**middlename**}}<**br**>  
 Заданно на: {{$homework->**date\_to\_completion**}}<**br**><**br**>  
  
 Дата выполнения: {{$homework->**date\_of\_completion**}}<**br**>  
 Оценка: {{$homework->**computer\_mark**}}<**br**>  
 <**hr**>  
 **@if (**!**empty**($homework->**teacher\_mark**)**)** <**p class="works\_show\_block\_title"**>  
 Оценка учителя: {{$homework->**teacher\_mark**}}  
 (Имеет больший приоретет)<**br**>  
 Причина изменения оценки:<**br**>  
 {!! *nl2br*($homework->**teacher\_comment**) !!}  
 </**p**>  
 <**hr**>  
 **@endif  
  
  
 @if (**!**empty**($homework\_content[**'given\_tasks'**])**)** <**p class="works\_show\_block\_title"**>Задачи:</**p**>  
 <**hr**>  
 **@foreach(**$homework\_content[**'given\_tasks'**] **as** $task**)** {!! *nl2br*($task->**task**)!!}<**br**><**br**>  
  
 Эталон: {{$task->**standard**}}<**br**>  
 **@if(**!**empty**($task->**answer**)**)** <**p class="answer\_exist"**>  
 Ответ учащегося: {{$task->**answer**}}  
 </**p**>  
 **@else** <**p class="answer\_absent"**>  
 Сдано без ответа  
 </**p**>  
 **@endif** <**hr**>  
 **@endforeach  
 @endif  
  
 @if (**!**empty**($homework\_content[**'given\_tests'**])**)** <**p class="works\_show\_block\_title"**>Тесты:</**p**>  
 <**hr**>  
 **@foreach(**$homework\_content[**'given\_tests'**] **as** $test**)** {!! *nl2br*($test->**task**) !!}<**br**>  
 A: {!! *nl2br*($test->**option\_a**) !!}<**br**>  
 B: {!! *nl2br*($test->**option\_b**) !!}<**br**>  
 C: {!! *nl2br*($test->**option\_c**) !!}<**br**>  
 D: {!! *nl2br*($test->**option\_d**) !!}<**br**><**br**>  
  
 Эталон: {{$test->**standard**}}<**br**>  
 **@if(**!**empty**($test->**answer**)**)** <**p class="answer\_exist"**>  
 Ответ учащегося: {!! *nl2br*($test->**answer**) !!}  
 </**p**>  
 **@else** <**p class="answer\_absent"**>  
 Сдано без ответа  
 </**p**>  
 **@endif** <**hr**>  
 **@endforeach  
 @endif  
  
 @if (**!**empty**($homework\_content[**'materials'**])**)** <**p class="works\_show\_block\_title"**>Дополнительные учебные материалы:</**p**>  
 <**hr**>  
 **@foreach(**$homework\_content[**'materials'**] **as** $material**)** <**a href="**{{route(**'materials.show'**, [**'id'**=>$material->**id**])}}**"**>  
 Дополнительный учебный материал № : {{ $material->**id** }}  
 </**a**><**br**>  
 Тема: {{$material->**theme**}}<**br**>  
 Заголовок: {{ $material->**title** }}<**br**>  
 Изображение: {{$material->**image**}}<**br**>  
 {!! *nl2br*($material->**body**) !!}<**br**>  
 <**hr**>  
 **@endforeach  
 @endif** <**a href="**{{route(**'homework.edit'**, [  
 **'grade\_id'** => $grade\_id,  
 **'date'** => $date,  
 **'id'** => $homework->**id** ])}}**"**>  
 Исправить оценку  
 </**a**>  
 <**hr**>  
  
 {!! Form::*open*([**'url'**=>route(**'toRePass'**)]) !!}  
 {!! Form::*hidden*(**'grade\_id'**, $grade\_id) !!}  
 {!! Form::*hidden*(**'date'**, $date) !!}  
 {!! Form::*hidden*(**'homework\_id'**, $homework->**id**) !!}  
 {!! Form::*submit*(**'На пересдачу'**, [**'class'**=>**''**]) !!}  
 {!! Form::*close*() !!}  
 <**hr**>  
 **@endif** {!! Form::*open*([**'url'**=>route(**'homework.destroy'**, [  
 **'grade\_id'** => $grade\_id,  
 **'date'** => $date,  
 **'id'** => $homework->**id** ])]) !!}  
 {!! Form::*submit*(**'Удалить домашнее задание'**, [**'class'**=>**''**]) !!}  
 {{method\_field(**'DELETE'**)}}  
 {!! Form::*close*() !!}  
 <**hr**>  
  
 Дата создания: {{ $homework->**created\_at** }}  
  
  
 </**div**>  
 </**div**>  
 </**div**>  
 </**div**>  
 </**div**>  
**@endsection**

Teacher/materials/create.blade.php

**@extends('layouts.app')  
  
@section('content')** <**div class="container"**>  
 <**div class="row"**>  
 <**div class="col-md-8 col-md-offset-2"**>  
 <**div class="panel panel-default"**>  
 <**div class="panel-heading"**>Добавить дополнительный учебный материал!</**div**>  
  
 <**div class="panel-body"**>  
 **@if (**session(**'status'**)**)** <**div class="alert alert-success"**>  
 {{ session(**'status'**) }}  
 </**div**>  
 **@endif** <**a href="/teacher"**>Учительская</**a**> >>  
 <**a href="/teacher/materials"**>Список материалов</**a**> >>  
 Новый учебный материал  
 <**br**><**br**>  
  
 {!! Form::*open*([**'url'**=>route(**'materials.store'**),**'method'**=>**'post'**, **'class'**=>**'form-horizontal'**]) !!}  
  
 <**div class="form-group**{{ $errors->has(**'theme'**) ? **' has-error'** : **''** }}**"**>  
 {!! Form::*label*(**'theme'**, **'Тема'**, [**'class'**=>**'col-md-4 control-label'**]) !!}  
  
 <**div class="col-md-6"**>  
 {!! Form::*text*(**'theme'**, old(**'theme'**), [**'required'**,**'class'**=>**'form-control'**]) !!}  
  
 **@if (**$errors->has(**'theme'**)**)** <**span class="help-block"**>  
 <**strong**>{{ $errors->first(**'theme'**) }}</**strong**>  
 </**span**>  
 **@endif** </**div**>  
 </**div**>  
  
 <**div class="form-group**{{ $errors->has(**'image'**) ? **' has-error'** : **''** }}**"**>  
 {!! Form::*label*(**'image'**, **'Ссылка на зображение'**, [**'class'**=>**'col-md-4 control-label'**]) !!}  
  
 <**div class="col-md-6"**>  
 {!! Form::*text*(**'image'**, **null**, [**'placeholder'**=>**'Не обязательно'**, **'class'**=>**'form-control'**]) !!}  
  
 **@if (**$errors->has(**'image'**)**)** <**span class="help-block"**>  
 <**strong**>{{ $errors->first(**'image'**) }}</**strong**>  
 </**span**>  
 **@endif** </**div**>  
 </**div**>  
  
 <**div class="form-group**{{ $errors->has(**'title'**) ? **' has-error'** : **''** }}**"**>  
 {!! Form::*label*(**'title'**, **'Заголовок'**, [**'class'**=>**'col-md-4 control-label'**]) !!}  
  
 <**div class="col-md-6"**>  
 {!! Form::*text*(**'title'**, **null**, [**'required'**,**'class'**=>**'form-control'**]) !!}  
  
 **@if (**$errors->has(**'title'**)**)** <**span class="help-block"**>  
 <**strong**>{{ $errors->first(**'title'**) }}</**strong**>  
 </**span**>  
 **@endif** </**div**>  
 </**div**>  
  
 <**div class="form-group**{{ $errors->has(**'body'**) ? **' has-error'** : **''** }}**"**>  
 {!! Form::*label*(**'body'**, **'Основной текст: '**, [**'class'**=>**'col-md-4 control-label'**]) !!}  
  
 <**div class="col-md-6"**>  
 {!! Form::*textarea*(**'body'**, **null**, [**'required'**,**'class'**=>**'form-control'**]) !!}  
  
 **@if (**$errors->has(**'body'**)**)** <**span class="help-block"**>  
 <**strong**>{{ $errors->first(**'body'**) }}</**strong**>  
 </**span**>  
 **@endif** </**div**>  
 </**div**>  
  
 <**div class="form-group"**>  
 <**div class="col-md-6 col-md-offset-4"**>  
 {!! Form::*submit*(**'Сохранить новый материал'**, [**'class'** => **'btn btn-primary'**]) !!}  
 </**div**>  
 </**div**>  
  
 {!! Form::*close*() !!}  
 </**div**>  
 </**div**>  
 </**div**>  
 </**div**>  
 </**div**>  
**@endsection**

Teacher/materials.blade.php

**@extends('layouts.app')  
  
@section('content')** <**div class="container"**>  
 <**div class="row"**>  
 <**div class="col-md-8 col-md-offset-2"**>  
 <**div class="panel panel-default"**>  
 <**div class="panel-heading"**>Список дополнительных учебных материалов!</**div**>  
  
 <**div class="panel-body"**>  
 **@if (**session(**'status'**)**)** <**div class="alert alert-success"**>  
 {{ session(**'status'**) }}  
 </**div**>  
 **@endif  
  
 @include('common.errors')** <**a href="/teacher"**>Учительская</**a**> >>  
 Список материалов  
 <**br**><**br**>  
  
 <**a href="**{{route(**'materials.create'**)}}**"**>Добавить новый материал</**a**><**br**><**br**>  
  
 <**hr**>  
 **@foreach(**$materials **as** $material**)** Материал №: {{ $material->**id** }}<**br**>  
 Тема: {{$material->**theme**}}<**br**>  
 Заголовок: {{$material->**title**}}<**br**><**br**>  
  
 Добавил: {{$author->getFIO($material->**teacher\_id**)}}<**br**>  
 Дата добавления: {{ $material->**created\_at** }}<**br**><**br**>  
  
 <**a href="**{{route(**'materials.show'**, [**'id'**=>$material->**id**])}}**"**>  
 Просмотр материала  
 </**a**><**br**>  
 <**hr**>  
  
 **@endforeach** {{$materials->links()}}  
 </**div**>  
 </**div**>  
 </**div**>  
 </**div**>  
 </**div**>  
**@endsection**

User/homework/show.blade.php

**@extends('layouts.app')  
  
@section('content')** <**div class="container"**>  
 <**div class="row"**>  
 <**div class="col-md-8 col-md-offset-2"**>  
 <**div class="panel panel-default"**>  
 <**div class="panel-heading"**>  
 Просмотр работы!  
 </**div**>  
 <**div class="panel-body"**>  
 **@if (**session(**'status'**)**)** <**div class="alert alert-success"**>  
 {{ session(**'status'**) }}  
 </**div**>  
 **@endif** <**a href="/desktop"**>Рабочий стол</**a**> >>  
 <**a href="/homeworks"**> Предмет </**a**> >>  
 <**a href="**{{route(**'showUsrDates'**, $discipline\_id)}}**"**> Дата </**a**> >>  
 <**a href="**{{route(**'showHomeworks'**, [$discipline\_id, $date])}}**"**> Задание </**a**> >>  
 Просмотр  
 <**hr**>  
  
 <**p class="works\_show\_block\_title"**>Домашнее задание по дисциплине:<**br**>  
 {{ $homework->**work**->**teacher**->**discipline**->**name** }}<**br**>  
 Заданно на: {{$homework->**date\_to\_completion**}}<**br**>  
 {{$homework->**schoolkid**->**grade**->**num**}}  
 -  
 {{$homework->**schoolkid**->**grade**->**char**}}<**br**>  
 {{$homework->**schoolkid**->**lastname**}}  
 {{$homework->**schoolkid**->**firstname**}}  
 {{$homework->**schoolkid**->**middlename**}}<**br**>  
 <**hr**>  
  
 **@if (**!$homework->**computer\_mark)  
 @if (**!**empty**($homework\_content[**'given\_tasks'**])**)** <**p class="works\_show\_block\_title"**>Задачи:</**p**>  
 <**hr**>  
 **@foreach(**$homework\_content[**'given\_tasks'**] **as** $task**)** {!! *nl2br*($task->**task**)!!}<**br**>  
 **@if(**!**empty**($task->**answer**)**)** Ваш ответ:  
 <**a href="**{{route(**'given\_task.edit'**, [  
 $discipline\_id, $date, $homework->**id**, $task->**id** ])}}**"**>  
 {!! *nl2br*($task->**answer**) !!}  
 </**a**>  
 **@else** <**a href="**{{route(**'given\_task.edit'**, [  
 $discipline\_id, $date, $homework->**id**,  
 $task->**id** ])}}**" class="answer\_absent"**>  
 Требуется ответ  
 </**a**>  
 **@endif** <**hr**>  
 **@endforeach  
 @endif  
  
 @if (**!**empty**($homework\_content[**'given\_tests'**])**)** <**p class="works\_show\_block\_title"**>Тесты:</**p**>  
 <**hr**>  
 **@foreach(**$homework\_content[**'given\_tests'**] **as** $test**)** {!! *nl2br*($test->**task**) !!}<**br**>  
 A: {!! *nl2br*($test->**option\_a**) !!}<**br**>  
 B: {!! *nl2br*($test->**option\_b**) !!}<**br**>  
 C: {!! *nl2br*($test->**option\_c**) !!}<**br**>  
 D: {!! *nl2br*($test->**option\_d**) !!}<**br**><**br**>  
  
 **@if(**!**empty**($test->**answer**)**)** Ваш ответ:  
 <**a href="**{{route(**'given\_test.edit'**, [  
 $discipline\_id, $date,  
 $homework->**id**, $test->**id**])}}**"**>  
 {!! *nl2br*($test->**answer**) !!}  
 </**a**>  
 **@else** <**a href="**{{route(**'given\_test.edit'**, [  
 $discipline\_id, $date, $homework->**id**, $test->**id** ])}}**" class="answer\_absent"**>  
 Требуется ответ  
 </**a**>  
 **@endif** <**hr**>  
 **@endforeach  
 @endif** {!! Form::*open*([**'url'**=>route(**'passHomework'**)]) !!}  
 {!! Form::*hidden*(**'discipline\_id'**, $discipline\_id) !!}  
 {!! Form::*hidden*(**'date'**, $date) !!}  
 {!! Form::*hidden*(**'homework\_id'**, $homework->**id**) !!}  
 {!! Form::*submit*(**'Сдать домашнее задание'**, [**'class'**=>**''**]) !!}  
 {!! Form::*close*() !!}  
  
 **@else** <**p class="works\_show\_block\_title"**>  
 Оценка: {{$homework->**computer\_mark**}} <**br**>  
 Дата выполнения: {{$homework->**date\_of\_completion**}}  
 </**p**>  
 <**hr**>  
 **@if (**!**empty**($homework->**teacher\_mark**)**)** <**p class="works\_show\_block\_title"**>  
 Оценка учителя: {{$homework->**teacher\_mark**}}  
 (Имеет больший приоретет)<**br**>  
 Причина изменения оценки:<**br**>  
 {!! *nl2br*($homework->**teacher\_comment**) !!}  
 </**p**>  
 <**hr**>  
 **@endif  
  
 @if (**!**empty**($homework\_content[**'given\_tasks'**])**)** <**p class="works\_show\_block\_title"**>Задачи:</**p**>  
 <**hr**>  
 **@foreach(**$homework\_content[**'given\_tasks'**] **as** $task**)** {!! *nl2br*($task->**task**)!!}<**br**>  
 **@if(**!**empty**($task->**answer**)**)** Ваш ответ: {!! *nl2br*($task->**answer**) !!}  
 **@else** Сдано без ответа  
 **@endif** <**hr**>  
 **@endforeach  
 @endif  
  
 @if (**!**empty**($homework\_content[**'given\_tests'**])**)** <**p class="works\_show\_block\_title"**>Тесты:</**p**>  
 <**hr**>  
 **@foreach(**$homework\_content[**'given\_tests'**] **as** $test**)** {!! *nl2br*($test->**task**) !!}<**br**>  
 A: {!! *nl2br*($test->**option\_a**) !!}<**br**>  
 B: {!! *nl2br*($test->**option\_b**) !!}<**br**>  
 C: {!! *nl2br*($test->**option\_c**) !!}<**br**>  
 D: {!! *nl2br*($test->**option\_d**) !!}<**br**><**br**>  
  
 **@if(**!**empty**($test->**answer**)**)** Ваш ответ: {!! *nl2br*($test->**answer**) !!}  
 **@else** Сдано без ответа  
 **@endif** <**hr**>  
 **@endforeach  
 @endif  
  
 @endif  
 @if (**!**empty**($homework\_content[**'materials'**])**)** <**p class="works\_show\_block\_title"**>Дополнительные учебные материалы:</**p**>  
 <**hr**>  
 **@foreach(**$homework\_content[**'materials'**] **as** $material**)** Тема: {{$material->**theme**}}<**br**>  
 Заголовок: {{ $material->**title** }}<**br**>  
 Изображение: {{$material->**image**}}<**br**>  
 {!! *nl2br*($material->**body**) !!}<**br**>  
 <**hr**>  
 **@endforeach  
 @endif** </**div**>  
 </**div**>  
 </**div**>  
 </**div**>  
 </**div**>  
**@endsection**

User/homework/solve/task.blade.php

**@extends('layouts.app')  
  
@section('content')** <**div class="container"**>  
 <**div class="row"**>  
 <**div class="col-md-8 col-md-offset-2"**>  
 <**div class="panel panel-default"**>  
 <**div class="panel-heading"**>Решать задачу</**div**>  
  
 <**div class="panel-body"**>  
 **@if (**session(**'status'**)**)** <**div class="alert alert-success"**>  
 {{ session(**'status'**) }}  
 </**div**>  
 **@endif  
 @include('common.errors')** <**a href="/desktop"**>Рабочий стол</**a**> >>  
 <**a href="/homeworks"**> Выбор предмета </**a**> >>  
 <**a href="**{{route(**'showUsrDates'**, $discipline\_id)}}**"**> Даты </**a**> >>  
 <**a href="**{{route(**'showHomeworks'**, [$discipline\_id, $date])}}**"**> Задания </**a**> >>  
 <**a href="**{{route(**'hometask.show'**, [  
 **'discipline\_id'** => $discipline\_id,  
 **'date'** => $date,  
 **'id'** => $homework\_id  
 ])}}**"**>Просмотр</**a**> >>  
 Решать задачу  
 <**br**>  
 <**hr**>  
  
  
 {!! Form::*model*($task, [**'method'**=>**'put'**, **'route'** => [**'given\_task.update'**,  
 **'discipline\_id'** => $discipline\_id, **'date'** => $date, **'hwrk\_id'** => $homework\_id,  
 **'id'**=>$task->**id**], **'class'**=>**'form-horizontal'**]) !!}  
  
 <**div class="form-group**{{ $errors->has(**'task'**) ? **' has-error'** : **''** }}**"**>  
 {!! Form::*label*(**'task'**, **'Условие задачи: '**, [**'class'**=>**'col-md-4 control-label'**]) !!}  
  
 <**div class="col-md-6"**>  
 <**p class=""**>{!! *nl2br*($task->**task**)!!}</**p**>  
  
 **@if (**$errors->has(**'task'**)**)** <**span class="help-block"**>  
 <**strong**>{{ $errors->first(**'task'**) }}</**strong**>  
 </**span**>  
 **@endif** </**div**>  
 </**div**>  
  
 <**div class="form-group**{{ $errors->has(**'answer'**) ? **' has-error'** : **''** }}**"**>  
 {!! Form::*label*(**'answer'**, **'Ответ'**, [**'class'**=>**'col-md-4 control-label'**]) !!}  
  
 <**div class="col-md-6"**>  
 {!! Form::*text*(**'answer'**, **null**, [**'class'**=>**'form-control'**]) !!}  
  
 **@if (**$errors->has(**'answer'**)**)** <**span class="help-block"**>  
 <**strong**>{{ $errors->first(**'answer'**) }}</**strong**>  
 </**span**>  
 **@endif** </**div**>  
 </**div**>  
  
 <**div class="form-group"**>  
 <**div class="col-md-6 col-md-offset-4"**>  
 {!! Form::*submit*(**'Ответить'**, [**'class'** => **'btn btn-primary'**]) !!}  
 </**div**>  
 </**div**>  
  
 {!! Form::*close*() !!}  
  
  
 </**div**>  
 </**div**>  
 </**div**>  
 </**div**>  
 </**div**>  
**@endsection**

User/homework/solve/test.blade.php

**@extends('layouts.app')  
  
@section('content')** <**div class="container"**>  
 <**div class="row"**>  
 <**div class="col-md-8 col-md-offset-2"**>  
 <**div class="panel panel-default"**>  
 <**div class="panel-heading"**>Решить тест</**div**>  
  
 <**div class="panel-body"**>  
 **@if (**session(**'status'**)**)** <**div class="alert alert-success"**>  
 {{ session(**'status'**) }}  
 </**div**>  
 **@endif** <**a href="/desktop"**>Рабочий стол</**a**> >>  
 <**a href="/homeworks"**> Выбор предмета </**a**> >>  
 <**a href="**{{route(**'showUsrDates'**, $discipline\_id)}}**"**> Даты </**a**> >>  
 <**a href="**{{route(**'showHomeworks'**, [$discipline\_id, $date])}}**"**> Задания </**a**> >>  
 <**a href="**{{route(**'hometask.show'**, [  
 **'discipline\_id'** => $discipline\_id,  
 **'date'** => $date,  
 **'id'** => $homework\_id  
 ])}}**"**>Просмотр</**a**> >>  
 Решить тест  
 <**br**>  
 <**hr**>  
  
 {!! Form::*model*($test, [**'method'**=>**'put'**, **'route'** => [**'given\_test.update'**,  
 **'discipline\_id'** => $discipline\_id, **'date'** => $date, **'hwrk\_id'** => $homework\_id,  
 **'id'**=>$test->**id**], **'class'**=>**'form-horizontal'**]) !!}  
  
 <**div class="form-group"**>  
 {!! Form::*label*(**'task'**, **'Вопрос: '**, [**'class'**=>**'col-md-4 control-label'**]) !!}  
  
 <**div class="col-md-6"**>  
 <**p class="given\_test\_opt"**>{!! *nl2br*($test->**task**) !!}</**p**>  
 </**div**>  
 </**div**>  
  
 <**div class="form-group"**>  
 {!! Form::*label*(**'option\_a'**, **'Вариант ответа A: '**, [**'class'**=>**'col-md-4 control-label'**]) !!}  
  
 <**div class="col-md-6"**>  
 <**p class="given\_test\_opt"**>{!! *nl2br*($test->**option\_a**) !!}</**p**>  
 </**div**>  
 </**div**>  
  
 <**div class="form-group"**>  
 {!! Form::*label*(**'option\_b'**, **'Вариант ответа B: '**, [**'class'**=>**'col-md-4 control-label'**]) !!}  
  
 <**div class="col-md-6"**>  
 <**p class="given\_test\_opt"**>{!! *nl2br*($test->**option\_b**) !!}</**p**>  
 </**div**>  
 </**div**>  
  
 <**div class="form-group"**>  
 {!! Form::*label*(**'option\_c'**, **'Вариант ответа C: '**, [**'class'**=>**'col-md-4 control-label'**]) !!}  
  
 <**div class="col-md-6"**>  
 <**p class="given\_test\_opt"**>{!! *nl2br*($test->**option\_c**) !!}</**p**>  
 </**div**>  
 </**div**>  
  
 <**div class="form-group"**>  
 {!! Form::*label*(**'option\_d'**, **'Вариант ответа D: '**, [**'class'**=>**'col-md-4 control-label'**]) !!}  
  
 <**div class="col-md-6"**>  
 <**p class="given\_test\_opt"**>{!! *nl2br*($test->**option\_d**) !!}</**p**>  
 </**div**>  
 </**div**>  
  
 <**div class="form-group**{{ $errors->has(**'answer'**) ? **' has-error'** : **''** }}**"**>  
 {!! Form::*label*(**'ans\_label'**, **'Ответ:'** , [**'class'**=>**'col-md-4 control-label'**]) !!}  
  
 <**div class="col-md-6"**>  
 <**p class="given\_test\_opt"**>  
 {!! Form::*label*(**'ans\_opt\_a'**, **'Вариант A'**) !!}  
 {!! Form::*radio*(**'answer'**, **'A'**, **false**, [**'id'** => **'ans\_opt\_a'**, **'class'**=>**''**, **'required'**]) !!}  
 <**br**>  
 {!! Form::*label*(**'ans\_opt\_b'**, **'Вариант B'**) !!}  
 {!! Form::*radio*(**'answer'**,**'B'**, **false**, [**'id'** => **'ans\_opt\_b'**, **'class'**=>**''**]) !!}  
 <**br**>  
 {!! Form::*label*(**'ans\_opt\_c'**, **'Вариант C'**) !!}  
 {!! Form::*radio*(**'answer'**,**'C'**, **false**, [**'id'** => **'ans\_opt\_c'**, **'class'**=>**''**]) !!}  
 <**br**>  
 {!! Form::*label*(**'ans\_opt\_d'**, **'Вариант D'**) !!}  
 {!! Form::*radio*(**'answer'**,**'D'**, **false**, [**'id'** => **'ans\_opt\_d'**, **'class'**=>**''**]) !!}  
 </**p**>  
 **@if (**$errors->has(**'answer'**)**)** <**span class="help-block"**>  
 <**strong**>{{ $errors->first(**'answer'**) }}</**strong**>  
 </**span**>  
 **@endif** </**div**>  
 </**div**>  
  
 <**div class="form-group"**>  
 <**div class="col-md-6 col-md-offset-4"**>  
 {!! Form::*submit*(**'Ответить'**, [**'class'** => **'btn btn-primary'**]) !!}  
 </**div**>  
 </**div**>  
  
 {!! Form::*close*() !!}  
  
 </**div**>  
 </**div**>  
 </**div**>  
 </**div**>  
 </**div**>  
**@endsection**

User/statistics/show.blade.php

**@extends('layouts.app')  
  
@section('content')** <**div class="container"**>  
 <**div class="row"**>  
 <**div class="col-md-8 col-md-offset-2"**>  
 <**div class="panel panel-default"**>  
 <**div class="panel-heading"**>Статистика. Выбор предмета</**div**>  
  
 <**div class="panel-body"**>  
 **@if (**session(**'status'**)**)** <**div class="alert alert-success"**>  
 {{ session(**'status'**) }}  
 </**div**>  
 **@endif  
 @include('common.errors')** <**a href="/desktop"**>Рабочий стол</**a**> >>  
 <**a href="/statistics"**>Статистика</**a**> >>  
 {{ $discipline->**name** }}  
 <**hr**>  
  
 <**p**>  
 Статистика по предмету {{$discipline->**name**}}:  
 </**p**>  
 <**p**>  
 Всего домашних заданий:  
 {{$homeworks\_count}}  
 </**p**>  
 <**p**>  
 Процент сданных работ:  
 {{$done\_percent}} %  
 </**p**>  
 <**p**>  
 Средний балл:  
 {{$average\_percent}} %  
 </**p**>  
 <**p**>  
 Средняя оценка:  
 {{$average\_mark}}  
 </**p**>  
  
 </**div**>  
 </**div**>  
 </**div**>  
 </**div**>  
 </**div**>  
**@endsection**

Контроллеры

IndexController.php

**<?php  
  
namespace** App\Http\Controllers;  
  
**use** Illuminate\Http\Request;  
**use** Illuminate\Support\Facades\Auth;  
  
**class** IndexController **extends** Controller  
{  
 **function** index()  
 {  
 **return** view(**'index'**, [**'title'** => **'ЭДЗ. Главная'**, **'user'** => Auth::*user*()]);  
 }  
}

AdminController.php

**<?php  
  
namespace** App\Http\Controllers;  
  
**use** Illuminate\Support\Facades\Auth;  
  
**class** AdminController **extends** Controller  
{  
 **protected $user**;  
  
 */\*\*  
 \* Create a new controller instance.  
 \*  
 \** ***@return*** *void  
 \*/* **public function** \_\_construct()  
 {  
 $this->middleware(**function** ($request, $next) {  
 **if** (Auth::**check**()) {  
 $this->**user** = Auth::**user**();  
 **if** (Auth::*user*()->**role** == **'a'**) {  
 **return** $next($request);  
 } **else** {  
 **return** redirect(**'/'**);  
 }  
 } **else** {  
 **return** redirect(**'login'**);  
 }  
 });  
 }  
}

TeacherController.php

**<?php  
  
namespace** App\Http\Controllers;  
  
**use** Illuminate\Support\Facades\Auth;  
  
**class** TeacherController **extends** Controller  
{  
 **protected $user**;  
  
 */\*\*  
 \* Create a new controller instance.  
 \*  
 \** ***@return*** *void  
 \*/* **public function** \_\_construct()  
 {  
 $this->middleware(**function** ($request, $next) {  
 **if** (Auth::**check**()) {  
 $this->**user** = Auth::**user**();  
 **if** (Auth::*user*()->**role** == **'t'**) {  
 **return** $next($request);  
 } **else** {  
 **return** redirect(**'/'**);  
 }  
 } **else** {  
 **return** redirect(**'login'**);  
 }  
  
 });  
 }  
}

UserController.php

**<?php  
  
namespace** App\Http\Controllers;  
  
**use** Illuminate\Support\Facades\Auth;  
  
**class** UserController **extends** Controller  
{  
 **protected $user**;  
  
 */\*\*  
 \* Create a new controller instance.  
 \*  
 \** ***@return*** *void  
 \*/* **public function** \_\_construct()  
 {  
 $this->middleware(**function** ($request, $next) {  
 **if** (Auth::**check**()) {  
 $this->**user** = Auth::**user**();  
 **if** (Auth::*user*()->**role** == **'s'**) {  
 **return** $next($request);  
 } **else** {  
 **return** redirect(**'/'**);  
 }  
 } **else** {  
 **return** redirect(**'login'**);  
 }  
  
 });  
 }  
}

Admin\GradeController.php

**<?php  
  
namespace** App\Http\Controllers\Admin;  
  
**use** App\Http\Controllers\AdminController;  
**use** Illuminate\Http\Request;  
**use** App\Grade;  
  
**class** GradeController **extends** AdminController  
{  
 */\*\*  
 \* Display a listing of the resource.  
 \*  
 \** ***@return*** *\Illuminate\Http\Response  
 \*/* **public function** index(Grade $grade)  
 {  
 **echo *\_\_METHOD\_\_***;  
 }  
  
 */\*\*  
 \* Show the form for creating a new resource.  
 \*  
 \** ***@return*** *\Illuminate\Http\Response  
 \*/* **public function** create()  
 {  
 **return** view(**'admin.grades.create'**, [**'title'** => **'ЭДЗ. Новый класс'**]);  
 }  
  
 */\*\*  
 \* Store a newly created resource in storage.  
 \*  
 \** ***@param*** *\App\Grade $grade  
 \** ***@param*** *\Illuminate\Http\Request $request  
 \** ***@return*** *\Illuminate\Http\Response  
 \*/* **public function** store(Request $request, Grade $grade)  
 {  
 $data = $request->except(**'\_token'**);  
 $response = $grade->store($data);  
  
 **if** (*is\_array*($response) && *array\_key\_exists*(**'errors'**, $response)) {  
 **return** back()->withInput()->withErrors($response[**'errors'**]);  
 }  
  
 $num = $response->**num**;  
 $char = $response->**char**;  
 $message = $num . **' - '** . $char . **' класс - добавлен'**;  
 **return** back()->with(**'status'**, $message);  
 }  
  
 */\*\*  
 \* Display the specified resource.  
 \*  
 \** ***@param*** *int $id  
 \** ***@return*** *\Illuminate\Http\Response  
 \*/* **public function** show($id)  
 {  
 $grade = **new** Grade();  
 $grade\_to\_show = $grade->getOne($id);  
 $name = $grade\_to\_show->**num** . **' - '** . $grade\_to\_show->**char**;  
  
 $schoolkids = $grade\_to\_show->schoolkids()->orderBy(**'lastname'**)->get();  
 **return** view(**'admin.grades.show'**, [  
 **'title'** => $name,  
 **'grade'** => $grade\_to\_show,  
 **'schoolkids'** => $schoolkids  
 ]);  
 }  
  
 */\*\*  
 \* Show the form for editing the specified resource.  
 \*  
 \** ***@param*** *int $id  
 \** ***@return*** *\Illuminate\Http\Response  
 \*/* **public function** edit($id)  
 {  
 $grade = **new** Grade();  
 $grade\_to\_update = $grade->getOne($id);  
 **return** view(**'admin.grades.edit'**, [  
 **'title'** => **'Редактирование класса'**,  
 **'grade'** => $grade\_to\_update  
 ]);  
 }  
  
 */\*\*  
 \* Update the specified resource in storage.  
 \*  
 \** ***@param*** *\Illuminate\Http\Request $request  
 \** ***@param*** *int $id  
 \** ***@return*** *\Illuminate\Http\Response  
 \*/* **public function** update(Request $request, $id)  
 {  
 $data = $request->except(**'\_token'**, **'\_method'**);  
 $grade = **new** Grade();  
  
 $response = $grade->edit($id, $data);  
  
 **if** (*is\_array*($response) && *array\_key\_exists*(**'errors'**, $response)) {  
 **return** back()->withInput()->withErrors($response[**'errors'**]);  
 }  
  
 $num = $response->**num**;  
 $char = $response->**char**;  
 $message = $num . **' - '** . $char . **' класс - данные обновлены'**;  
 **return** redirect(route(**'grade.show'**, $id))->with(**'status'**, $message);  
 }  
  
 */\*\*  
 \* Remove the specified resource from storage.  
 \*  
 \** ***@param*** *int $id  
 \** ***@return*** *\Illuminate\Http\Response  
 \*/* **public function** destroy($id)  
 {  
 $grade = **new** Grade();  
 $grade\_to\_delete = $grade->getOne($id);  
 $allKids = $grade\_to\_delete->**schoolkids**->count();  
 **if** ($allKids === 0) {  
 $num = $grade\_to\_delete->**num**;  
 $char = $grade\_to\_delete->**char**;  
 $grade->kill($id);  
 $message = **'Класс '** . $num . **' - '** . $char . **' удален'**;  
 **return** redirect(**'/admin/lists/grades/'**)->withErrors($message);  
 }  
 $message = **'Вы не можете удалить из системы класс, в котором есть ученики.   
 Обратитесь к системному администратору'**;  
 **return** back()->withErrors($message);  
 }  
}

Teacher/WorksController.php

**<?php  
  
namespace** App\Http\Controllers\Teacher;  
  
**use** App\Http\Controllers\TeacherController;  
**use** App\Teacher;  
**use** App\Work;  
**use** Illuminate\Http\Request;  
  
  
**class** WorksController **extends** TeacherController  
{  
 */\*\*  
 \* Display a listing of the resource.  
 \*  
 \** ***@return*** *\Illuminate\Http\Response  
 \*/* **public function** index(Work $work, Teacher $author)  
 {  
 *//$all\_works = $work->getAllPaginated();* $work = **new** Work();  
 $discipline\_id = $this->**user**->**teacher**->**discipline\_id**;  
 $discipline\_works = $work->getDisciplineWorksPaginated($discipline\_id);  
  
 **return** view(**'teacher.works'**, [  
 **'title'** => **'ЭДЗ. Работы'**,  
 **'works'** => $discipline\_works,  
 **'author'** => $author  
 ]);  
 }  
  
 */\*\*  
 \* Show the form for creating a new resource.  
 \*  
 \** ***@return*** *\Illuminate\Http\Response  
 \*/* **public function** create()  
 {  
 **return** view(**'teacher.works.create'**, [**'title'** => **'ЭДЗ. Создать задание'**]);  
 }  
  
 */\*\*  
 \* Store a newly created resource in storage.  
 \*  
 \** ***@param*** *\Illuminate\Http\Request $request  
 \** ***@return*** *\Illuminate\Http\Response  
 \*/* **public function** store(Request $request, Work $work)  
 {  
 $data = $request->except(**'\_token'**);  
 $response = $work->store($data, $this->**user**->**teacher**->**id**);  
 **if** (*is\_array*($response) && *array\_key\_exists*(**'errors'**, $response)) {  
 **return** back()->withInput()->withErrors($response[**'errors'**]);  
 }  
 $new\_work\_id = $response->**id**;  
 $message = **'Работа добавлена под номером '** . $new\_work\_id;  
 **return** redirect(**'/teacher/works'**)->with(**'status'**, $message);  
 }  
  
 */\*\*  
 \* Display the specified resource.  
 \*  
 \** ***@param*** *int $id  
 \** ***@return*** *\Illuminate\Http\Response  
 \*/* **public function** show($id)  
 {  
 $work = **new** Work();  
 $work\_to\_show = $work->getOne($id);  
  
 **if** ($this->**user**->can(**'view'**, $work\_to\_show)) {  
  
 $work\_content = [  
 **'tasks'** => $work\_to\_show->**tasks**->all(),  
 **'tests'** => $work\_to\_show->**tests**->all(),  
 **'materials'** => $work\_to\_show->**materials**->all()  
 ];  
  
 $author = **new** Teacher();  
 $author\_fio = $author->getFIO($work\_to\_show->**teacher\_id**);  
  
 **return** view(**'teacher.works.show'**, [  
 **'title'** => **'ЭДЗ. Просмотр работы'**,  
 **'work'** => $work\_to\_show,  
 **'teacher'** => $this->**user**->**teacher**,  
 **'author\_fio'** => $author\_fio,  
 **'work\_content'** => $work\_content  
 ]);  
 }  
 $message = **'ОШИБКА. Нет права просмотра !!!'**;  
 **return** redirect(**'/teacher/works/'**)->withErrors($message);  
 }  
  
 */\*\*  
 \* Show the form for editing the specified resource.  
 \*  
 \** ***@param*** *int $id  
 \** ***@return*** *\Illuminate\Http\Response  
 \*/* **public function** edit($id)  
 {  
 $work = **new** Work();  
 $work\_to\_update = $work->getOne($id);  
 $this->authorize(**'update'**, $work\_to\_update);  
  
 **return** view(**'teacher.works.edit'**, [  
 **'title'** => **'ЭДЗ. Новый тест'**,  
 **'work\_to\_update'** => $work\_to\_update  
 ]);  
 }  
  
 */\*\*  
 \* Update the specified resource in storage.  
 \*  
 \** ***@param*** *\Illuminate\Http\Request $request  
 \** ***@param*** *int $id  
 \** ***@return*** *\Illuminate\Http\Response  
 \*/* **public function** update(Request $request, $id)  
 {  
 $data = $request->except(**'\_token'**);  
 $work = **new** Work();  
 $work\_for\_update = $work->getOne($id);  
  
 **if** ($this->**user**->can(**'update'**, $work\_for\_update)) {  
  
 $response = $work->edit($id, $this->**user**->**teacher**->**id**, $data);  
  
 **if** (*is\_array*($response) && *array\_key\_exists*(**'errors'**, $response)) {  
 **return** back()->withInput()->withErrors($response[**'errors'**]);  
 }  
 $message = **'Работа под номером '** . $id . **' обновлена'**;  
 **return** redirect(**'/teacher/works/'** . $id)->with(**'status'**, $message);  
 }  
 $message = **'ОШИБКА. Нет права редактирования !!!'**;  
 **return** redirect(**'/teacher/works/'** . $id)->withErrors($message);  
 }  
  
 */\*\*  
 \* Remove the specified resource from storage.  
 \*  
 \** ***@param*** *int $id  
 \** ***@return*** *\Illuminate\Http\Response  
 \*/* **public function** destroy($id)  
 {  
 $work = **new** Work();  
 $work\_for\_delete = $work->getOne($id);  
 **if** ($this->**user**->can(**'update'**, $work\_for\_delete)) {  
 $work->kill($id);  
 $message = **'Работа '** . $id . **' удалена!'**;  
 **return** redirect(**'/teacher/works'**)->with(**'status'**, $message);  
 }  
 $message = **'ОШИБКА. Нет права удаления !!!'**;  
 **return** redirect(**'/teacher/works/'** . $id)->withErrors($message);  
 }  
}

Teacher/Set/SetWorkController.php

**<?php  
  
namespace** App\Http\Controllers\Teacher\Set;  
  
**use** App\Given\_task;  
**use** App\Given\_test;  
**use** App\Http\Controllers\TeacherController;  
**use** App\Grade;  
**use** App\Homework;  
**use** App\Work;  
**use** Illuminate\Http\Request;  
  
**class** SetWorkController **extends** TeacherController  
{  
 **public function** index($id, Work $work, Grade $grade)  
 {  
 $work\_to\_set = $work->getOne($id);  
 $grades = $grade->getAll();  
 **if** ($this->**user**->can(**'view'**, $work\_to\_set)) {  
 **return** view(**'teacher.works.set'**, [  
 **'title'** => **'ЭДЗ. Задать работу'**,  
 **'work\_id'** => $id,  
 **'grades'** => $grades  
 ]);  
 }  
 $message = **'ОШИБКА. Нет прав'**;  
 **return** redirect(**'/teacher/works/'**)->withErrors($message);  
 }  
  
 **public function** set(Request $request, Work $work)  
 {  
 $data = $request->except(**'\_token'**);  
 $work\_to\_set = $work->getOne($data[**'work\_id'**]);  
  
 **if** ($this->**user**->can(**'view'**, $work\_to\_set)) {  
  
 $grade\_id = $data[**'grade'**];  
 $grade = **new** Grade();  
 $grade = $grade->getOne($grade\_id);  
 $allKids = $grade->**schoolkids**->count();  
 $hasHomeworkKids = 0;  
  
 **if** ($allKids === 0) {  
 $message = **'В классе нет учеников'**;  
 **return** back()->withErrors($message);  
 }  
  
 $homework = **new** Homework();  
  
 **foreach** ($grade->**schoolkids as** $kid) {  
  
 $hasHomework = $kid->hasHomework($work\_to\_set);  
  
 **if** (!$hasHomework) {  
 $teacher\_id = $this->**user**->**teacher**->**id**;  
 $homework\_id = $kid->setHomework($work\_to\_set, $teacher\_id, $data[**'date'**]);  
  
 $homework\_to\_fill = $homework->getOne($homework\_id);  
 $tests = $homework\_to\_fill->**work**->**tests**;  
  
 **foreach** ($tests **as** $test) {  
 $given\_test = **new** Given\_test();  
 $given\_test->store($test, $homework\_id);  
 }  
  
 $tasks = $homework\_to\_fill->**work**->**tasks**;  
  
 **foreach** ($tasks **as** $task) {  
 $given\_task = **new** Given\_task();  
 $given\_task->store($task, $homework\_id);  
 }  
  
 } **else** {  
 $hasHomeworkKids = $hasHomeworkKids + 1;  
 }  
 }  
 **if** ($allKids === $hasHomeworkKids) {  
 $message = **'Работа '** . $data[**'work\_id'**] . **' уже была заданна '** . $grade->**name** . **' классу'**;  
 **return** back()->withErrors($message);  
 }  
 $message = **'Работа '** . $data[**'work\_id'**] . **' заданна '** . $grade->**name** . **' классу'**;  
 **return** back()->with(**'status'**, $message);  
 }  
 $message = **'ОШИБКА. Нет прав'**;  
 **return** redirect(**'/teacher/works/'**)->withErrors($message);  
 }  
}

Teacher/HomeworkController.php

**<?php  
  
namespace** App\Http\Controllers\Teacher;  
  
**use** App\Http\Controllers\TeacherController;  
**use** Illuminate\Http\Request;  
**use** App\Homework;  
  
**class** HomeworkController **extends** TeacherController  
{  
 */\*\*  
 \* Display a listing of the resource.  
 \*  
 \** ***@return*** *\Illuminate\Http\Response  
 \*/* **public function** index()  
 {  
 **return** redirect(**'\teacher'**);  
 }  
  
 */\*\*  
 \* Show the form for creating a new resource.  
 \*  
 \** ***@return*** *\Illuminate\Http\Response  
 \*/* **public function** create()  
 {  
 **echo *\_\_METHOD\_\_***;  
 }  
  
 */\*\*  
 \* Store a newly created resource in storage.  
 \*  
 \** ***@param*** *\Illuminate\Http\Request $request  
 \** ***@return*** *\Illuminate\Http\Response  
 \*/* **public function** store(Request $request)  
 {  
 **echo *\_\_METHOD\_\_***;  
 }  
  
 */\*\*  
 \* Display the specified resource.  
 \*  
 \** ***@param*** *int $grade\_id  
 \** ***@param*** *string $date  
 \** ***@param*** *int $homework\_id  
 \** ***@return*** *\Illuminate\Http\Response  
 \*/* **public function** show($grade\_id, $date, $homework\_id)  
 {  
 $homework = **new** Homework();  
 $homework\_to\_show = $homework->getOne($homework\_id);  
  
 **if** ($this->**user**->can(**'view'**, $homework\_to\_show)  
 && ($grade\_id == $homework\_to\_show->**schoolkid**->**grade\_id**)  
 && ($date == $homework\_to\_show->**date\_to\_completion**)  
 ) {  
  
 **if** (**isset**($homework\_to\_show->**computer\_mark**)) {  
 $mark\_to\_show = $homework->percent\_per\_character($homework\_to\_show->**computer\_mark**);  
 $homework\_to\_show->**computer\_mark** = $mark\_to\_show;  
 }  
 **if** (**isset**($homework\_to\_show->**teacher\_mark**)) {  
 $mark\_to\_show = $homework->percent\_per\_character($homework\_to\_show->**teacher\_mark**);  
 $homework\_to\_show->**teacher\_mark** = $mark\_to\_show;  
 }  
  
 $homework\_content = [  
 **'given\_tasks'** => $homework\_to\_show->**given\_tasks**->all(),  
 **'given\_tests'** => $homework\_to\_show->**given\_tests**->all(),  
 **'materials'** => $homework\_to\_show->**work**->**materials**->all()  
 ];  
  
 **return** view(**'teacher.homework.show'**, [  
 **'title'** => **'ЭДЗ. Просмотр домашнего задания'**,  
 **'grade\_id'** => $grade\_id,  
 **'date'** => $date,  
 **'homework'** => $homework\_to\_show,  
 **'homework\_content'** => $homework\_content  
 ]);  
 }  
 $message = **'ОШИБКА. Нет прав!!!'**;  
 **return** redirect(**'/teacher'**)->withErrors($message);  
 }  
  
 */\*\*  
 \* Show the form for editing the specified resource.  
 \*  
 \** ***@param*** *int $grade\_id  
 \** ***@param*** *string $date  
 \** ***@param*** *int $homework\_id  
 \** ***@return*** *\Illuminate\Http\Response  
 \*/* **public function** edit($grade\_id, $date, $homework\_id)  
 {  
 $homework = **new** Homework();  
 $homework\_to\_show = $homework->getOne($homework\_id);  
  
 **if** ($this->**user**->can(**'edit\_mark'**, $homework\_to\_show)  
 && ($grade\_id == $homework\_to\_show->**schoolkid**->**grade\_id**)  
 && ($date == $homework\_to\_show->**date\_to\_completion**)  
 ) {  
 $homework\_content = [  
 **'given\_tasks'** => $homework\_to\_show->**given\_tasks**->all(),  
 **'given\_tests'** => $homework\_to\_show->**given\_tests**->all(),  
 **'materials'** => $homework\_to\_show->**work**->**materials**->all()  
 ];  
  
 **return** view(**'teacher.homework.edit'**, [  
 **'title'** => **'ЭДЗ. Исправить оценку'**,  
 **'grade\_id'** => $grade\_id,  
 **'date'** => $date,  
 **'homework'** => $homework\_to\_show,  
 **'homework\_content'** => $homework\_content  
 ]);  
 }  
 $message = **'ОШИБКА. Нет прав!!!'**;  
 **return** redirect(**'/teacher'**)->withErrors($message);  
 }  
  
 */\*\*  
 \* Update the specified resource in storage.  
 \*  
 \** ***@param*** *\Illuminate\Http\Request $request  
 \** ***@param*** *int $grade\_id  
 \** ***@param*** *string $date  
 \** ***@param*** *int $homework\_id  
 \** ***@return*** *\Illuminate\Http\Response  
 \*/* **public function** update(Request $request, $grade\_id, $date, $homework\_id)  
 {  
 $data = $request->except(**'\_token'**, **'\_method'**);  
 $homework = **new** Homework();  
 $homework\_to\_update = $homework->getOne($homework\_id);  
 **if** ($this->**user**->can(**'edit\_mark'**, $homework\_to\_update)  
 && ($grade\_id == $homework\_to\_update->**schoolkid**->**grade\_id**)  
 && ($date == $homework\_to\_update->**date\_to\_completion**)  
 ) {  
 $response = $homework->edit\_mark($homework\_id, $data);  
  
 **if** (*is\_array*($response) && *array\_key\_exists*(**'errors'**, $response)) {  
 **return** back()->withInput()->withErrors($response[**'errors'**]);  
 }  
  
 $message = **'Изменения внесены'**;  
 **return** redirect( route(**'homework.show'**,[  
 $grade\_id, $date, $homework\_id  
 ]))->with(**'status'**, $message);  
 }  
 $message = **'ОШИБКА. Нет прав!!!'**;  
 **return** redirect(**'/teacher'**)->withErrors($message);  
 }  
  
 */\*\*  
 \* Remove the specified resource from storage.  
 \*  
 \** ***@param*** *int $grade\_id  
 \** ***@param*** *string $date  
 \** ***@param*** *int $homework\_id  
 \** ***@return*** *\Illuminate\Http\Response  
 \*/* **public function** destroy($grade\_id, $date, $homework\_id)  
 {  
 **echo *\_\_METHOD\_\_***;  
 dump($homework\_id);  
 }  
}

Teacher/Homework/TchrChoiceController.php

**<?php  
  
namespace** App\Http\Controllers\Teacher\Homework;  
  
**use** App\Http\Controllers\TeacherController;  
**use** App\Grade;  
**use** App\Homework;  
  
**class** TchrChoiceController **extends** TeacherController  
{  
 **public function** index(Grade $grade)  
 {  
 $grades\_to\_show = $grade->getTeacherGrades($this->**user**->**teacher**->**id**)->all();  
  
 **return** view(**'teacher.homeworks'**, [  
 **'title'** => **'Выбор класса'**,  
 **'grades'** => $grades\_to\_show  
 ]);  
 }  
  
 **public function** show\_dates($grade\_id)  
 {  
 $homework = **new** Homework();  
 $homework\_dates = $homework->getTeacherGradeDates($this->**user**->**teacher**->**id**, $grade\_id);  
  
 **return** view(**'teacher.homework.calendar'**, [  
 **'title'** => **'Выбор даты'**,  
 **'grade\_id'** => $grade\_id,  
 **'dates'** => $homework\_dates  
 ]);  
 }  
  
 **public function** show\_homeworks($grade\_id, $date)  
 {  
 $homework = **new** Homework();  
 $homeworks\_to\_show = $homework->getByDateForTeacher($this->**user**->**teacher**->**id**, $grade\_id, $date);  
  
 **return** view(**'teacher.homework.homework\_kids'**, [  
 **'title'** => **'Домашние задания'**,  
 **'grade\_id'** => $grade\_id,  
 **'date'** => $date,  
 **'homeworks'** => $homeworks\_to\_show  
 ]);  
 }  
}

Teacher/AccountController.php

**<?php***/\*  
 \* Испльзуется имя класса AccountController для модели Teacher,  
 \* а не более подходящее TeacherController потому что класс TeacherController  
 \* уже есть и используется для проверки роли учителя  
 \*/***namespace** App\Http\Controllers\Teacher;  
  
**use** App\Http\Controllers\TeacherController;  
**use** Illuminate\Http\Request;  
**use** App\Teacher;  
  
**class** AccountController **extends** TeacherController  
{  
 */\*\*  
 \* Display a listing of the resource.  
 \*  
 \** ***@return*** *\Illuminate\Http\Response  
 \*/* **public function** index()  
 {  
 **echo *\_\_METHOD\_\_***;  
 }  
  
 */\*\*  
 \* Show the form for creating a new resource.  
 \*  
 \** ***@return*** *\Illuminate\Http\Response  
 \*/* **public function** create()  
 {  
 **echo *\_\_METHOD\_\_***;  
 }  
  
 */\*\*  
 \* Store a newly created resource in storage.  
 \*  
 \** ***@param*** *\Illuminate\Http\Request $request  
 \** ***@return*** *\Illuminate\Http\Response  
 \*/* **public function** store(Request $request)  
 {  
 **echo *\_\_METHOD\_\_***;  
 }  
  
 */\*\*  
 \* Display the specified resource.  
 \*  
 \** ***@param*** *int $id  
 \** ***@return*** *\Illuminate\Http\Response  
 \*/* **public function** show($id)  
 {  
 $teacher = **new** Teacher();  
 $teacher\_to\_show = $teacher->getOne($id);  
  
 **if** ($this->**user**->can(**'view'**, $teacher\_to\_show)) {  
 **return** view(**'teacher.account'**, [  
 **'title'** => **'ЭДЗ. Просмотр профиля'**,  
 **'teacher'** => $teacher\_to\_show,  
 ]);  
 }  
 $message = **'ОШИБКА. Нет прав'**;  
 **return** redirect(**'/teacher'**)->withErrors($message);  
 }  
  
 */\*\*  
 \* Show the form for editing the specified resource.  
 \*  
 \** ***@param*** *int $id  
 \** ***@return*** *\Illuminate\Http\Response  
 \*/* **public function** edit($id)  
 {  
 $teacher = **new** Teacher();  
 $teacher\_to\_update = $teacher->getOne($id);  
 **if** ($this->**user**->can(**'update'**, $teacher\_to\_update)) {  
  
 **return** view(**'teacher.account.edit'**, [  
 **'title'** => **'ЭДЗ. Редактирование профиля'**,  
 **'teacher'** => $teacher\_to\_update,  
 ]);  
 }  
  
 $message = **'ОШИБКА. Нет прав'**;  
 **return** redirect(**'/teacher'**)->withErrors($message);  
 }  
  
 */\*\*  
 \* Update the specified resource in storage.  
 \*  
 \** ***@param*** *\Illuminate\Http\Request $request  
 \** ***@param*** *int $id  
 \** ***@return*** *\Illuminate\Http\Response  
 \*/* **public function** update(Request $request, $id)  
 {  
 $data = $request->except(**'\_token'**);  
 $teacher = **new** Teacher();  
 $teacher\_to\_update = $teacher->getOne($id);  
 **if** ($this->**user**->can(**'update'**, $teacher\_to\_update)) {  
  
 $response = $teacher->edit($id, $data);  
 **if** (*is\_array*($response) && *array\_key\_exists*(**'errors'**, $response)) {  
 **return** back()->withInput()->withErrors($response[**'errors'**]);  
 }  
 $message = **'Учетная запись обновлена'**;  
 **return** redirect(**'/teacher/account/'** . $id)->with(**'status'**, $message);  
  
 }  
 $message = **'ОШИБКА. Нет прав'**;  
 **return** redirect(**'/teacher'**)->withErrors($message);  
 }  
  
 */\*\*  
 \* Remove the specified resource from storage.  
 \*  
 \** ***@param*** *int $id  
 \** ***@return*** *\Illuminate\Http\Response  
 \*/* **public function** destroy($id)  
 {  
 $teacher = **new** Teacher();  
 $teacher\_to\_delete = $teacher->getOne($id);  
 **if** ($this->**user**->can(**'delete'**, $teacher\_to\_delete)) {  
 $teacher->kill($id);  
 $message = **'Учетная запись учителя '** . $id . **' удалена!'**;  
 **return** redirect(**'/'**)->with(**'status'**, $message);  
 }  
 $message = **'ОШИБКА. Нет права удаления !!!'**;  
 **return** redirect(**'/teacher'**)->withErrors($message);  
 }  
}

User/HometaskController.php

**<?php  
  
namespace** App\Http\Controllers\User;  
  
**use** App\Http\Controllers\UserController;  
**use** Illuminate\Http\Request;  
**use** App\Homework;  
  
**class** HometaskController **extends** UserController  
{  
 */\*\*  
 \* Display a listing of the resource.  
 \*  
 \** ***@return*** *\Illuminate\Http\Response  
 \*/* **public function** index()  
 {  
 **echo *\_\_METHOD\_\_***;  
 }  
  
 */\*\*  
 \* Show the form for creating a new resource.  
 \*  
 \** ***@return*** *\Illuminate\Http\Response  
 \*/* **public function** create()  
 {  
 **echo *\_\_METHOD\_\_***;  
 }  
  
 */\*\*  
 \* Store a newly created resource in storage.  
 \*  
 \** ***@param*** *\Illuminate\Http\Request $request  
 \** ***@return*** *\Illuminate\Http\Response  
 \*/* **public function** store(Request $request)  
 {  
 **echo *\_\_METHOD\_\_***;  
 }  
  
 */\*\*  
 \* Display the specified resource.  
 \*  
 \** ***@param*** *int $id  
 \** ***@return*** *\Illuminate\Http\Response  
 \*/* **public function** show($discipline\_id, $date, $id)  
 {  
 $homework = **new** Homework();  
 $homework\_to\_show = $homework->getOne($id);  
  
  
 **if** ($this->**user**->can(**'view'**, $homework\_to\_show)  
 && ($discipline\_id == $homework\_to\_show->**work**->**teacher**->**discipline\_id**)  
 && ($date == $homework\_to\_show->**date\_to\_completion**)  
 ) {  
  
 **if** (**isset**($homework\_to\_show->**computer\_mark**)) {  
 $mark\_to\_show = $homework->percent\_per\_character($homework\_to\_show->**computer\_mark**);  
 $homework\_to\_show->**computer\_mark** = $mark\_to\_show;  
 }  
 **if** (**isset**($homework\_to\_show->**teacher\_mark**)) {  
 $mark\_to\_show = $homework->percent\_per\_character($homework\_to\_show->**teacher\_mark**);  
 $homework\_to\_show->**teacher\_mark** = $mark\_to\_show;  
 }  
  
 $homework\_content = [  
 **'given\_tasks'** => $homework\_to\_show->**given\_tasks**->all(),  
 **'given\_tests'** => $homework\_to\_show->**given\_tests**->all(),  
 **'materials'** => $homework\_to\_show->**work**->**materials**->all()  
 ];  
  
 **return** view(**'user.homework.show'**, [  
 **'title'** => **'ЭДЗ. Просмотр домашнего задания'**,  
 **'discipline\_id'** => $discipline\_id,  
 **'date'** => $date,  
 **'homework'** => $homework\_to\_show,  
 **'homework\_content'** => $homework\_content,  
 ]);  
  
 }  
 $message = **'ОШИБКА. Нет прав!!!'**;  
 **return** redirect(**'/desktop'**)->withErrors($message);  
 }  
  
 */\*\*  
 \* Show the form for editing the specified resource.  
 \*  
 \** ***@param*** *int $id  
 \** ***@return*** *\Illuminate\Http\Response  
 \*/* **public function** edit($id)  
 {  
 **echo *\_\_METHOD\_\_***;  
 }  
  
 */\*\*  
 \* Update the specified resource in storage.  
 \*  
 \** ***@param*** *\Illuminate\Http\Request $request  
 \** ***@param*** *int $id  
 \** ***@return*** *\Illuminate\Http\Response  
 \*/* **public function** update(Request $request, $id)  
 {  
 **echo *\_\_METHOD\_\_***;  
 }  
  
 */\*\*  
 \* Remove the specified resource from storage.  
 \*  
 \** ***@param*** *int $id  
 \** ***@return*** *\Illuminate\Http\Response  
 \*/* **public function** destroy($discipline\_id, $date, $id)  
 {  
 **echo *\_\_METHOD\_\_***;  
 dump($id);  
 }  
}

User/GivenTaskController.php

**<?php  
  
namespace** App\Http\Controllers\User;  
  
**use** App\Given\_task;  
**use** App\Homework;  
**use** App\Http\Controllers\UserController;  
**use** Illuminate\Http\Request;  
  
**class** GivenTaskController **extends** UserController  
{  
 */\*\*  
 \* Display a listing of the resource.  
 \*  
 \** ***@return*** *\Illuminate\Http\Response  
 \*/* **public function** index()  
 {  
 **echo *\_\_METHOD\_\_***;  
 }  
  
 */\*\*  
 \* Show the form for creating a new resource.  
 \*  
 \** ***@return*** *\Illuminate\Http\Response  
 \*/* **public function** create()  
 {  
 **echo *\_\_METHOD\_\_***;  
 }  
  
 */\*\*  
 \* Store a newly created resource in storage.  
 \*  
 \** ***@param*** *\Illuminate\Http\Request $request  
 \** ***@return*** *\Illuminate\Http\Response  
 \*/* **public function** store(Request $request)  
 {  
 **echo *\_\_METHOD\_\_***;  
 }  
  
 */\*\*  
 \* Display the specified resource.  
 \*  
 \** ***@param*** *int $task\_id  
 \** ***@return*** *\Illuminate\Http\Response  
 \*/* **public function** show($task\_id)  
 {  
 **echo *\_\_METHOD\_\_***;  
 }  
  
 */\*\*  
 \* Show the form for editing the specified resource.  
 \*  
 \** ***@param*** *int $discipline\_id  
 \** ***@param*** *string $date  
 \** ***@param*** *int $homework\_id  
 \** ***@param*** *int $task\_id  
 \** ***@return*** *\Illuminate\Http\Response  
 \*/* **public function** edit($discipline\_id, $date, $homework\_id, $task\_id)  
 {  
 $task = **new** Given\_task();  
 $task\_to\_update = $task->getOne($task\_id);  
  
 $homework = **new** Homework();  
 $homework\_to\_solve = $homework->getOne($homework\_id);  
  
 **if** (**empty**($homework\_to\_solve)) {  
 $message = **'ОШИБКА. Нет прав'**;  
 **return** redirect(**'/desktop'**)->withErrors($message);  
 }  
  
 **if** ($this->**user**->can(**'update'**, [$task\_to\_update, $homework\_to\_solve])  
 && ($discipline\_id == $homework\_to\_solve->**work**->**teacher**->**discipline\_id**)  
 && ($date == $homework\_to\_solve->**date\_to\_completion**)  
 && ($homework\_id == $homework\_to\_solve->**id**)  
 ) {  
  
 **return** view(**'user.homework.solve.task'**, [  
 **'title'** => **'ЭДЗ. Решать задачу'**,  
 **'discipline\_id'** => $discipline\_id,  
 **'date'** => $date,  
 **'homework\_id'** => $homework\_id,  
 **'task'** => $task\_to\_update  
 ]);  
 }  
  
 $message = **'ОШИБКА. Нет прав'**;  
 **return** redirect(**'/desktop'**)->withErrors($message);  
 }  
  
 */\*\*  
 \* Update the specified resource in storage.  
 \*  
 \** ***@param*** *\Illuminate\Http\Request $request  
 \** ***@param*** *int $discipline\_id  
 \** ***@param*** *string $date  
 \** ***@param*** *int $homework\_id  
 \** ***@param*** *int $task\_id  
 \** ***@return*** *\Illuminate\Http\Response  
 \*/* **public function** update(Request $request, $discipline\_id, $date, $homework\_id, $task\_id)  
 {  
  
 $data = $request->except(**'\_token'**, **'\_method'**);  
 $task = **new** Given\_task();  
 $task\_to\_update = $task->getOne($task\_id);  
  
 $homework = **new** Homework();  
 $homework\_to\_solve = $homework->getOne($homework\_id);  
  
 **if** ($this->**user**->can(**'update'**, [$task\_to\_update, $homework\_to\_solve])  
 && ($discipline\_id == $homework\_to\_solve->**work**->**teacher**->**discipline\_id**)  
 && ($date == $homework\_to\_solve->**date\_to\_completion**)  
 && ($homework\_id == $homework\_to\_solve->**id**)  
 ) {  
 $response = $task->edit($task\_id, $data);  
  
 **if** (*is\_array*($response) && *array\_key\_exists*(**'errors'**, $response)) {  
 **return** back()->withInput()->withErrors($response[**'errors'**]);  
 }  
  
 $message = **'Ответ записан'**;  
 **return** redirect( route(**'hometask.show'**,[  
 $discipline\_id, $date, $homework\_id  
 ]))->with(**'status'**, $message);  
 }  
 $message = **'ОШИБКА. Нет прав'**;  
 **return** redirect(**'/desktop'**)->withErrors($message);  
 }  
  
 */\*\*  
 \* Remove the specified resource from storage.  
 \*  
 \** ***@param*** *int $task\_id  
 \** ***@return*** *\Illuminate\Http\Response  
 \*/* **public function** destroy($task\_id)  
 {  
 **echo *\_\_METHOD\_\_***;  
 }  
}

User/Homework/UsrChoiceController.php

**<?php  
  
namespace** App\Http\Controllers\User\Homework;  
  
**use** App\Http\Controllers\UserController;  
**use** App\Schoolkid;  
**use** App\Homework;  
  
**class** UsrChoiceController **extends** UserController  
{  
  
 **public function** index(Schoolkid $schoolkid)  
 {  
 $disciplines\_to\_show = $schoolkid->getKidDisciplines($this->**user**->**schoolkid**->**id**);  
  
 **return** view(**'user.homeworks'**, [  
 **'title'** => **'Выбор предмета'**,  
 **'disciplines'** => $disciplines\_to\_show  
 ]);  
 }  
  
 **public function** show\_dates($discipline\_id)  
 {  
 $homework = **new** Homework();  
 $homework\_dates = $homework->getKidDisciplineDates($this->**user**->**schoolkid**->**id**, $discipline\_id);  
  
 **return** view(**'user.homework.calendar'**, [  
 **'title'** => **'Выбор даты'**,  
 **'discipline\_id'** => $discipline\_id,  
 **'dates'** => $homework\_dates  
 ]);  
 }  
  
 **public function** show\_homeworks($discipline\_id, $date)  
 {  
 $homework = **new** Homework();  
 $homeworks\_to\_show = $homework->getByDateForKid($this->**user**->**schoolkid**->**id**, $discipline\_id, $date);  
  
 **if** ($homeworks\_to\_show->count() === 1) {  
 $homework\_to\_show = $homeworks\_to\_show->first();  
 $date = $homework\_to\_show->**date\_to\_completion**;  
 **return** redirect(route(**'hometask.show'**, [  
 **'discipline\_id'** => $discipline\_id,  
 **'date'** => $date,  
 **'id'** => $homework\_to\_show->**id**,  
 ]));  
  
 };  
  
 **return** view(**'user.homework.homeworks\_for\_kid'**, [  
 **'title'** => **'Домашние задания'**,  
 **'discipline\_id'** => $discipline\_id,  
 **'date'** => $date,  
 **'homeworks'** => $homeworks\_to\_show  
 ]);  
 }  
}

User/Homework/PassController.php

**<?php  
  
namespace** App\Http\Controllers\User\Homework;  
  
**use** App\Homework;  
**use** App\Http\Controllers\UserController;  
**use** Illuminate\Http\Request;  
  
**class** PassController **extends** UserController  
{  
 **public function** pass(Request $request)  
 {  
 $data = $request->except(**'\_token'**);  
  
 $homework = **new** Homework();  
 $homework\_to\_pass = $homework->getOne($data[**'homework\_id'**]);  
  
 **if** ($this->**user**->can(**'pass'**, $homework\_to\_pass)) {  
  
 $tasks = $homework\_to\_pass->**given\_tasks**;  
 $tasks\_num = $tasks->count();  
 $tasks\_correct\_num = $tasks->where(**'correct\_flag'**, 1)->count();  
  
 $tests = $homework\_to\_pass->**given\_tests**;  
 $tests\_num = $tests->count();  
 $tests\_correct\_num = $tests->where(**'correct\_flag'**, 1)->count();  
  
 *//Если задана пустая работа, ученик сдавая ее получает 100%* **if**(($tasks\_num + $tests\_num) !== 0)  
 {  
 $mark = ($tasks\_correct\_num + $tests\_correct\_num) / ($tasks\_num + $tests\_num) \* 100;  
 $mark = *round*($mark, 2);  
 } **else** {  
 $mark = 100;  
 }  
  
 $homework->evaluate($data[**'homework\_id'**], $mark);  
  
 $message = **'Оценка посчитана'**;  
 **return** redirect(route(**'hometask.show'**, [  
 **'discipline\_id'** => $data[**'discipline\_id'**],  
 **'date'** => $data[**'date'**],  
 **'id'** => $data[**'homework\_id'**],  
 ]))->with(**'status'**, $message);  
 }  
 }  
}

Политики

HomeworkPolicy.php

**<?php  
  
namespace** App\Policies;  
  
**use** App\User;  
**use** App\Homework;  
**use** Illuminate\Auth\Access\HandlesAuthorization;  
  
**class** HomeworkPolicy  
{  
 **use** HandlesAuthorization;  
  
 */\*\*  
 \* Determine whether the user can view the homework.  
 \*  
 \** ***@param*** *\App\User $user  
 \** ***@param*** *\App\Homework $homework  
 \** ***@return*** *mixed  
 \*/* **public function** view(User $user, Homework $homework)  
 {  
 **if** (**isset**($user->**teacher**))  
 **return** $user->**teacher**->**id** === $homework->**teacher\_id**;  
 **if** (**isset**($user->**schoolkid**))  
 **return** $user->**schoolkid**->**id** === $homework->**schoolkid\_id**;  
 }  
  
 */\*\*  
 \* Determine whether the user can create homeworks.  
 \*  
 \** ***@param*** *\App\User $user  
 \** ***@return*** *mixed  
 \*/* **public function** create(User $user)  
 {  
 *//* }  
  
 */\*\*  
 \* Determine whether the user can update the homework.  
 \*  
 \** ***@param*** *\App\User $user  
 \** ***@param*** *\App\Homework $homework  
 \** ***@return*** *mixed  
 \*/* **public function** update(User $user, Homework $homework)  
 {  
 *//* }  
  
 */\*\*  
 \* Determine whether the user can delete the homework.  
 \*  
 \** ***@param*** *\App\User $user  
 \** ***@param*** *\App\Homework $homework  
 \** ***@return*** *mixed  
 \*/* **public function** delete(User $user, Homework $homework)  
 {  
 *//* }  
  
 */\*\*  
 \* Determine whether the user can update the homework.  
 \*  
 \** ***@param*** *\App\User $user  
 \** ***@param*** *\App\Homework $homework  
 \** ***@return*** *mixed  
 \*/* **public function** pass(User $user, Homework $homework)  
 {  
 **return** $user->**schoolkid**->**id** === $homework->**schoolkid\_id**;  
 }  
  
 */\*\*  
 \* Determine whether the user can update the homework.  
 \*  
 \** ***@param*** *\App\User $user  
 \** ***@param*** *\App\Homework $homework  
 \** ***@return*** *mixed  
 \*/* **public function** edit\_mark(User $user, Homework $homework)  
 {  
 **return** (($user->**teacher**->**id** === $homework->**teacher\_id**)  
 &&  
 ($homework->**computer\_mark**));  
 }  
}

Модели

Discipline.php

**<?php  
  
namespace** App;  
  
**use** Illuminate\Database\Eloquent\Model;  
**use** Illuminate\Validation\Rule;  
**use** Validator;  
  
**class** Discipline **extends** Model  
{  
 **protected $fillable** = [  
 **'name'**, **'description'** ];  
  
 **public $timestamps** = **false**;  
  
 **public function** getAll()  
 {  
 **return** $this->orderBy(**'name'**)->get();  
 }  
  
 **public function** getOne($id)  
 {  
 **return** $this->find($id);  
 }  
  
 **public function** getAllPaginated($num)  
 {  
 $entities = $this->orderBy(**'name'**)->paginate($num);  
 **return** $entities;  
 }  
  
 **public function** store($data)  
 {  
 **if** ($err = $this->validate($data)) {  
 **return** $err;  
 }  
  
 $newDiscipline = [  
 **'name'** => $data[**'name'**],  
 **'description'** => $data[**'description'**],  
 ];  
 $this->fill($newDiscipline);  
 $this->save();  
 **return** $this;  
 }  
  
 **public function** edit($id, $data)  
 {  
 **if** ($err = $this->validate($data)) {  
 **return** $err;  
 }  
  
 $entity = $this->find($id);  
  
 $entity->**name** = $data[**'name'**];  
 $entity->**description** = $data[**'description'**];  
  
 $entity->save();  
 **return** $entity;  
 }  
  
 **public function** validate($data)  
 {  
 **if** (*array\_key\_exists*(**'id'**, $data))  
 $unique\_rule = Rule::*unique*(**'disciplines'**)->ignore($data[**'id'**]);  
 **else** $unique\_rule = **'unique:disciplines'**;  
  
 $validator = Validator::*make*($data,  
 [  
 **'name'** => [  
 **'required'**,  
 $unique\_rule,  
 ],  
 ],  
 [  
 **'name.required'** => **'Необходимо указать название предмета'**,  
 **'name.unique'** => **'Предмет '** . $data[**'name'**] . **' уже есть в списке'**,  
 ]);  
  
 **if** ($validator->fails()) {  
 **return** [**'errors'** => $validator->errors()];  
 }  
 }  
  
 **public function** kill($id)  
 {  
 $entity = $this->find($id);  
 **return** $entity->delete();  
 }  
  
 *//Eloquent: Relationships* **public function** teachers()  
 {  
 **return** $this->hasMany(**'App\Teacher'**, **'discipline\_id'**);  
 }  
}

Given\_test.php

**<?php  
  
namespace** App;  
  
**use** Illuminate\Database\Eloquent\Model;  
**use** Validator;  
  
**class** Given\_test **extends** Model  
{  
 **protected $fillable** = [  
 **'homework\_id'**, **'theme'**, **'task'**, **'option\_a'**, **'option\_b'**, **'option\_c'**, **'option\_d'**, **'answer'**, **'standard'**, **'correct\_flag'** ];  
  
 **public function** store($standard\_test, $homework\_id)  
 {  
 $newGivenTest = [  
 **'homework\_id'** => $homework\_id,  
 **'theme'** => $standard\_test[**'theme'**],  
 **'task'** => $standard\_test[**'task'**],  
 **'option\_a'** => $standard\_test[**'option\_a'**],  
 **'option\_b'** => $standard\_test[**'option\_b'**],  
 **'option\_c'** => $standard\_test[**'option\_c'**],  
 **'option\_d'** => $standard\_test[**'option\_d'**],  
 **'standard'** => $standard\_test[**'answer'**]  
 ];  
 $this->fill($newGivenTest);  
 $this->save();  
 **return** $this;  
 }  
  
 **public function** getOne($id)  
 {  
 $entity = $this->find($id);  
 **return** $entity;  
 }  
  
 **public function** edit($id, $data)  
 {  
 **if** ($err = $this->validate($data)) {  
 **return** $err;  
 }  
 $entity = $this->find($id);  
 $entity->**answer** = $data[**'answer'**];  
  
 **if** ($entity->**answer** == $entity->**standard**) {  
 $entity->**correct\_flag** = 1;  
 } **else** {  
 $entity->**correct\_flag** = **NULL**;  
 }  
  
 $entity->save();  
 **return** $entity;  
 }  
  
 **public function** validate($data)  
 {  
 $validator = Validator::*make*($data,  
 [  
 **'answer'** => **'required'** ],  
 [  
 **'answer.required'** => **'Необходимо указать ответ'** ]);  
  
 **if** ($validator->fails()) {  
 **return** [**'errors'** => $validator->errors()];  
 }  
 }  
  
 *//Eloquent: Relationships* **public function** homework()  
 {  
 **return** $this->belongsTo(**'App\Homework'**);  
 }  
}

Homework.php

**<?php  
  
namespace** App;  
  
**use** Illuminate\Database\Eloquent\Model;  
**use** Validator;  
  
**class** Homework **extends** Model  
{  
 **protected $fillable** = [  
 **'computer\_mark'**, **'teacher\_comment'**, **'teacher\_mark'** ];  
  
 **public function** getOne($id)  
 {  
 $entity = $this->find($id);  
 **return** $entity;  
 }  
  
 **public function** getTeacherGradeDates($teacher\_id, $grade\_id)  
 {  
 $entities = $this  
 ->join(**'teachers'**, **'homeworks.teacher\_id'**, **'='**, **'teachers.id'**)  
 ->join(**'schoolkids'**, **'homeworks.schoolkid\_id'**, **'='**, **'schoolkids.id'**)  
 ->join(**'grades'**, **'schoolkids.grade\_id'**, **'='**, **'grades.id'**)  
 ->where(**'homeworks.teacher\_id'**, $teacher\_id)  
 ->where(**'grades.id'**, $grade\_id)  
 ->select(**'homeworks.date\_to\_completion'**)  
 ->orderBy(**'date\_to\_completion'**, **'desc'**)  
 ->distinct()  
 ->paginate(15);  
  
 **return** $entities;  
 }  
  
 **public function** getByDateForTeacher($teacher\_id, $grade\_id, $date)  
 {  
 $entities = $this  
 ->join(**'works'**, **'homeworks.work\_id'**, **'='**, **'works.id'**)  
 ->join(**'schoolkids'**, **'homeworks.schoolkid\_id'**, **'='**, **'schoolkids.id'**)  
 ->join(**'grades'**, **'schoolkids.grade\_id'**, **'='**, **'grades.id'**)  
 ->where(**'homeworks.teacher\_id'**, $teacher\_id)  
 ->where(**'date\_to\_completion'**, $date)  
 ->where(**'grades.id'**, $grade\_id)  
 ->select(**'homeworks.\*'**)  
 ->get();  
 **return** $entities;  
 }  
  
 **public function** getKidDisciplineDates($schoolkid\_id, $discipline\_id)  
 {  
 $entities = $this  
 ->join(**'works'**, **'homeworks.work\_id'**, **'='**, **'works.id'**)  
 ->join(**'teachers'**, **'works.teacher\_id'**, **'='**, **'teachers.id'**)  
 ->join(**'disciplines'**, **'teachers.discipline\_id'**, **'='**, **'disciplines.id'**)  
 ->where(**'homeworks.schoolkid\_id'**, $schoolkid\_id)  
 ->where(**'disciplines.id'**, $discipline\_id)  
 ->select(**'homeworks.date\_to\_completion'**)  
 ->orderBy(**'date\_to\_completion'**, **'desc'**)  
 ->distinct()  
 ->paginate(15);  
  
 **return** $entities;  
 }  
  
 **public function** getByDateForKid($schoolkid\_id, $discipline\_id, $date)  
 {  
 $entities = $this  
 ->join(**'works'**, **'homeworks.work\_id'**, **'='**, **'works.id'**)  
 ->join(**'teachers'**, **'works.teacher\_id'**, **'='**, **'teachers.id'**)  
 ->join(**'disciplines'**, **'teachers.discipline\_id'**, **'='**, **'disciplines.id'**)  
 ->where(**'homeworks.schoolkid\_id'**, $schoolkid\_id)  
 ->where(**'date\_to\_completion'**, $date)  
 ->where(**'disciplines.id'**, $discipline\_id)  
 ->select(**'homeworks.\*'**)  
 ->get();  
 **return** $entities;  
 }  
  
 **public function** getKidDisciplineHomeworks($schoolkid\_id, $discipline\_id)  
 {  
 $entities = $this  
 ->join(**'works'**, **'homeworks.work\_id'**, **'='**, **'works.id'**)  
 ->join(**'teachers'**, **'works.teacher\_id'**, **'='**, **'teachers.id'**)  
 ->join(**'disciplines'**, **'teachers.discipline\_id'**, **'='**, **'disciplines.id'**)  
 ->where(**'homeworks.schoolkid\_id'**, $schoolkid\_id)  
 ->where(**'disciplines.id'**, $discipline\_id)  
 ->select(**'homeworks.\*'**)  
*// ->orderBy('date\_to\_completion', 'desc')* ->get();  
  
 **return** $entities;  
 }  
  
 **public function** evaluate($id, $mark)  
 {  
 $entity = $this->find($id);  
  
 **if** ($mark != 0) {  
 $entity->**computer\_mark** = $mark;  
 } **else** {  
 $entity->**computer\_mark** = 1;  
 }  
 $entity->**date\_of\_completion** = *date*(**'Y-m-d'**);  
 $entity->save();  
 **return** $entity;  
 }  
  
 **public function** percent\_per\_character($percent)  
 {  
 **if** ($percent > 100 || $percent < 0)  
 **return null**;  
 **if** ($percent == 100)  
 $char = **'5+'**;  
 **else if** ($percent >= 90 && $percent <= 99)  
 $char = **'5'**;  
 **else if** ($percent >= 85 && $percent <= 89)  
 $char = **'5-'**;  
 **else if** ($percent >= 70 && $percent <= 84)  
 $char = **'4'**;  
 **else if** ($percent >= 65 && $percent <= 69)  
 $char = **'4-'**;  
 **else if** ($percent >= 50 && $percent <= 64)  
 $char = **'3'**;  
 **else if** ($percent >= 45 && $percent <= 49)  
 $char = **'3-'**;  
 **else if** ($percent >= 30 && $percent <= 44)  
 $char = **'2'**;  
 **else if** ($percent >= 1 && $percent <= 29)  
 $char = **'2-'**;  
 **else** $char = **'1'**;  
 **return** $char;  
 }  
  
 **public function** edit\_mark($id, $data)  
 {  
 **if** ($err = $this->validate\_mark($data)) {  
 **return** $err;  
 }  
 $entity = $this->find($id);  
  
 $entity->**teacher\_comment** = $data[**'teacher\_comment'**];  
  
 **if** (!**empty**($data[**'teacher\_mark'**])) {  
 $entity->**teacher\_mark** = $data[**'teacher\_mark'**];  
 }  
  
 $entity->save();  
 **return** $entity;  
 }  
  
 **public function** validate\_mark($data)  
 {  
 $validator = Validator::*make*($data,  
 [  
 **'teacher\_comment'** => **'required'**,  
 **'teacher\_mark'** => **'integer|min:10|max:100'**,  
 ],  
 [  
 **'teacher\_comment.required'** => **'Напишите пару слов'**,  
 **'teacher\_mark.integer'** => **'Оценка должна быть целым числом'**,  
 **'teacher\_mark.min'** => **'Оценка не должна быть меньше 1 процента'**,  
 **'teacher\_mark.max'** => **'Оценка не должна быть больше 100 процентов'**,  
 ]);  
  
 **if** ($validator->fails()) {  
 **return** [**'errors'** => $validator->errors()];  
 }  
 }  
  
 **public function** clear\_marks($id)  
 {  
 $entity = $this->find($id);  
  
 $entity->**computer\_mark** = **NULL**;  
 $entity->**teacher\_comment** = **NULL**;  
 $entity->**teacher\_mark** = **NULL**;  
  
 $entity->save();  
 **return** $entity;  
 }  
  
  
 *//Eloquent: Relationships* **public function** schoolkid()  
 {  
 **return** $this->belongsTo(**'App\Schoolkid'**);  
 }  
  
 **public function** work()  
 {  
 **return** $this->belongsTo(**'App\Work'**);  
 }  
  
 **public function** given\_tasks()  
 {  
 **return** $this->HasMany(**'App\Given\_task'**);  
 }  
  
 **public function** given\_tests()  
 {  
 **return** $this->HasMany(**'App\Given\_test'**);  
 }  
}

Material.php

**<?php  
  
namespace** App;  
  
**use** Illuminate\Database\Eloquent\Model;  
**use** Validator;  
  
**class** Material **extends** Model  
{  
 **protected $fillable** = [  
 **'teacher\_id'**, **'theme'**, **'image'**, **'title'**, **'body'** ];  
  
 **public function** getAllPaginated()  
 {  
 $entities = $this->orderBy(**'id'**, **'desc'**)->paginate(10);  
 **return** $entities;  
 }  
  
 **public function** getDisciplineMaterialsPaginated($discipline\_id)  
 {  
 $entities = $this->join(**'teachers'**, **'materials.teacher\_id'**, **'='**, **'teachers.id'**)  
 ->join(**'disciplines'**, **'teachers.discipline\_id'**, **'='**, **'disciplines.id'**)  
 ->where(**'disciplines.id'**, $discipline\_id)  
 ->select(**'materials.\*'**)  
 ->orderBy(**'materials.id'**, **'desc'**)  
 ->paginate(10);  
 **return** $entities;  
 }  
  
 **public function** getOne($id)  
 {  
 $entity = $this->find($id);  
 **return** $entity;  
 }  
  
 **public function** store($data, $teacher\_id)  
 {  
 **if** ($err = $this->validate($data)) {  
 **return** $err;  
 }  
  
 **if** ($data[**'image'**] == **''**) {  
 $data[**'image'**] = **'no image'**;  
 }  
  
 $newMaterial = [  
 **'teacher\_id'** => $teacher\_id,  
 **'theme'** => $data[**'theme'**],  
 **'image'** => $data[**'image'**],  
 **'title'** => $data[**'title'**],  
 **'body'** => $data[**'body'**]  
 ];  
 $this->fill($newMaterial);  
 $this->save();  
 **return** $this;  
 }  
  
 **public function** edit($id, $teacher\_id, $data)  
 {  
 **if** ($err = $this->validate($data)) {  
 **return** $err;  
 }  
  
 $entity = $this->find($id);  
  
 $entity->**teacher\_id** = $teacher\_id;  
 $entity->**theme** = $data[**'theme'**];  
 $entity->**image** = $data[**'image'**];  
 $entity->**title** = $data[**'title'**];  
 $entity->**body** = $data[**'body'**];  
  
 **if** ($entity->**image** == **''**) {  
 $entity->**image** = **'no image'**;  
 }  
  
 $entity->save();  
 **return** $entity;  
 }  
  
 **public function** validate($data)  
 {  
 $validator = Validator::*make*($data,  
 [  
 **'theme'** => **'required'**,  
 **'title'** => **'required'**,  
 **'body'** => **'required'** ],  
 [  
 **'theme.required'** => **'Необходимо указать тему'**,  
 **'title.required'** => **'Необходимо указать заголовок'**,  
 **'body.required'** => **'Необходимо указать основной текст'** ]);  
  
 **if** ($validator->fails()) {  
 **return** [**'errors'** => $validator->errors()];  
 }  
 }  
  
 **public function** kill($id)  
 {  
 $entity = $this->find($id);  
 **return** $entity->delete();  
 }  
  
 *//Eloquent: Relationships* **public function** teacher()  
 {  
 **return** $this->belongsTo(**'App\Teacher'**);  
 }  
  
 **public function** works()  
 {  
 **return** $this->belongsToMany(**'App\Work'**);  
 }  
}

Schoolkid.php

**<?php  
  
namespace** App;  
  
**use** Illuminate\Database\Eloquent\Model;  
**use** Illuminate\Support\Facades\DB;  
**use** Validator;  
**use** Illuminate\Validation\Rule;  
**use** Illuminate\Support\Facades\Hash;  
  
**class** Schoolkid **extends** Model  
{  
  
 **public $timestamps** = **false**;  
  
 */\*\*  
 \* The attributes that are mass assignable.  
 \*  
 \** ***@var*** *array  
 \*/* **protected $fillable** = [  
 **'firstname'**, **'middlename'**, **'lastname'**, **'grade\_id'** ];  
  
 **public function** getOne($id)  
 {  
 $entity = $this->find($id);  
 **return** $entity;  
 }  
  
 **public function** getAllPaginated($num)  
 {  
 $entities = $this->orderBy(**'lastname'**)->paginate($num);  
 **return** $entities;  
 }  
  
 **public function** hasHomework($work)  
 {  
 **return** $this->works()->where(**'work\_id'**, $work->**id**)->exists();  
 }  
  
 **public function** setHomework($work, $teacher\_id, $date)  
 {  
 $this->works()->attach($work, [**'teacher\_id'** => $teacher\_id, **'date\_to\_completion'** => $date]);  
 *// get inserted Identifier of pivot when attach()  
 // Получаем Id записи в промежуточной таблице ( pivot ) после выполнения attach()* $homework\_id = $this->works()->withPivot(**'id'**)->wherePivot(**'work\_id'**, $work->**id**)->first()->**pivot**->**id**;  
 **return** $homework\_id;  
 }  
  
 **public function** getKidDisciplines($schoolkid\_id)  
 {  
 $entities = $this  
 ->join(**'homeworks'**, **'schoolkids.id'**, **'='**, **'homeworks.schoolkid\_id'**)  
 ->join(**'works'**, **'homeworks.work\_id'**, **'='**, **'works.id'**)  
 ->join(**'teachers'**, **'works.teacher\_id'**, **'='**, **'teachers.id'**)  
 ->join(**'disciplines'**, **'teachers.discipline\_id'**, **'='**, **'disciplines.id'**)  
 ->where(**'schoolkids.id'**, $schoolkid\_id)  
 ->select(**'disciplines.\*'**)  
 ->distinct()  
 ->get();  
 **return** $entities;  
 }  
  
 **public function** edit($id, $data)  
 {  
 $entity = $this->find($id);  
  
 **if** ($err = $this->validate($data, $entity->**user**->**id**)) {  
 **return** $err;  
 }  
  
 DB::*beginTransaction*();  
  
 $entity->**user**->**email** = $data[**'email'**];  
 **if** (**isset**($data[**'login'**])) {  
 $entity->**user**->**login** = $data[**'login'**];  
 }  
  
 $email\_is\_change = $entity->**user**->isDirty();  
 $entity->**user**->save();  
  
 $entity->**firstname** = $data[**'firstname'**];  
 $entity->**middlename** = $data[**'middlename'**];  
 $entity->**lastname** = $data[**'lastname'**];  
 **if** (**isset**($data[**'grade'**])) {  
 $entity->**grade\_id** = $data[**'grade'**];  
 }  
  
 $data\_is\_change = $entity->isDirty();  
 $entity->save();  
  
 DB::*commit*();  
  
 **if** ($email\_is\_change || $data\_is\_change) {  
 **return** $entity;  
 } **else** {  
 **return** [**'errors'** => [**'no\_chenge'** => **'Изменений нет'**]];  
 }  
 }  
  
 **public function** validate($data, $id)  
 {  
 $validator = Validator::*make*($data,  
 [  
 **'email'** =>  
 [  
 **'nullable'**,  
 **'email'**,  
 **'max:255'**,  
 Rule::*unique*(**'users'**)->ignore($id)  
 ],  
 **'login'** =>  
 [  
 **'max:255'**,  
 Rule::*unique*(**'users'**)->ignore($id)  
 ],  
 **'firstname'** => **'required|string|max:255'**,  
 **'middlename'** => **'max:255'**,  
 **'lastname'** => **'required|string|max:255'** ],  
 [  
 **'email.unique'** => **'Почта занята'**,  
 **'login.unique'** => **'Логин занят'**,  
 **'firstname.required'** => **'Необходимо указать имя'**,  
 **'firstname.max'** => **'Поле не должно быть больше 255 символов'**,  
 **'middlename.max'** => **'Поле не должно быть больше 255 символов'**,  
 **'lastname.required'** => **'Необходимо указать фамилию'**,  
 **'lastname.max'** => **'Поле не должно быть больше 255 символов'**,  
 ]);  
  
 **if** ($validator->fails()) {  
 **return** [**'errors'** => $validator->errors()];  
 }  
 }  
  
 **public function** change\_password($id, $data)  
 {  
 $entity = $this->find($id);  
  
 **if** ($err = $this->validate\_password($data)) {  
 **return** $err;  
 }  
  
 **if** (Hash::*check*($data[**'password'**], $entity->**user**->**password**)) {  
 $entity->**user**->**password** = Hash::*make*($data[**'password-new'**]);  
 $entity->**user**->save();  
 **return** $entity;  
 }  
 **return** [**'errors'** => [**'password'** => **'Неправильный пароль'**]];  
 }  
  
 **public function** validate\_password($data)  
 {  
 $validator = Validator::*make*($data,  
 [  
 **'password'** => **'required|string|min:6'**,  
 **'password-new'** => **'required|string|min:6|confirmed'**,  
 ],  
 [  
 **'password.required'** => **'Укажите пароль'**,  
 **'password.min'** => **'Пароль должен содержать как минимум 6 символов'**,  
 **'password-new.required'** => **'Укажите новый пароль'**,  
 **'password-new.min'** => **'Пароль должен содержать как минимум 6 символов'**,  
 **'password-new.confirmed'** => **'Новый пароль и подтверждение пароля не совпадают'**,  
 ]);  
  
 **if** ($validator->fails()) {  
 **return** [**'errors'** => $validator->errors()];  
 }  
 }  
  
 **public function** getStat($homeworks)  
 {  
 $homeworks\_count = $homeworks->count();  
 $done\_homeworks\_count = $homeworks->where(**'computer\_mark'**, **'!='**, **NULL**)->count();  
  
 $done\_percent = ($done\_homeworks\_count / $homeworks\_count) \* 100;  
 $done\_percent = *round*($done\_percent, 2);  
 $sum\_of\_marks = 0;  
  
 **foreach** ($homeworks **as** $homework) {  
 **if** (!**empty**($homework->**teacher\_mark**))  
 $sum\_of\_marks = $sum\_of\_marks + $homework->**teacher\_mark**;  
 **else** $sum\_of\_marks = $sum\_of\_marks + $homework->**computer\_mark**;  
 }  
  
 $average\_percent = ($sum\_of\_marks / $homeworks\_count);  
 $average\_percent = *round*($average\_percent, 2);  
  
 **return** [  
 **'homeworks\_count'** => $homeworks\_count,  
 **'done\_percent'** => $done\_percent,  
 **'average\_percent'** => $average\_percent,  
 ];  
 }  
  
 **public function** kill($id)  
 {  
 $entity = $this->find($id);  
 DB::*beginTransaction*();  
  
 $entity->user()->delete();  
  
 DB::*commit*();  
 }  
  
 *//Eloquent: Relationships* **public function** user()  
 {  
 **return** $this->belongsTo(**'App\User'**);  
 }  
  
 **public function** works()  
 {  
 **return** $this->belongsToMany(**'App\Work'**, **'homeworks'**)  
 ->withPivot(**'date\_to\_completion'**)  
 ->withTimestamps();  
 }  
  
 **public function** grade()  
 {  
 **return** $this->belongsTo(**'App\Grade'**);  
 }  
  
 **public function** homeworks()  
 {  
 **return** $this->hasMany(**'App\Homework'**);  
 }  
}

Teacher.php

**<?php  
  
namespace** App;  
  
**use** Illuminate\Database\Eloquent\Model;  
**use** Validator;  
**use** Illuminate\Support\Facades\Hash;  
**use** Illuminate\Validation\Rule;  
**use** Illuminate\Support\Facades\DB;  
  
**class** Teacher **extends** Model  
{  
  
 **public $timestamps** = **false**;  
  
 */\*\*  
 \* The attributes that are mass assignable.  
 \*  
 \** ***@var*** *array  
 \*/* **protected $fillable** = [  
 **'firstname'**, **'middlename'**, **'lastname'**, **'discipline\_id'** ];  
  
 **public function** getOne($id)  
 {  
 $entity = $this->find($id);  
 **return** $entity;  
 }  
  
 **public function** getFIO($id)  
 {  
 $teacher = $this->find($id);  
 $fio = $teacher->**firstname** . **' '** . $teacher->**middlename** . **' '** . $teacher->**lastname**;  
 **return** $fio;  
 }  
  
 **public function** getAllPaginated($num)  
 {  
 $entities = $this->orderBy(**'lastname'**)->paginate($num);  
 **return** $entities;  
 }  
  
 **public function** edit($id, $data)  
 {  
 $entity = $this->find($id);  
  
 **if** ($err = $this->validate($data, $entity->**user**->**id**)) {  
 **return** $err;  
 }  
  
 DB::*beginTransaction*();  
  
 $entity->**user**->**email** = $data[**'email'**];  
  
 **if** (**isset**($data[**'login'**])) {  
 $entity->**user**->**login** = $data[**'login'**];  
 }  
  
 $email\_is\_change = $entity->**user**->isDirty();  
  
 $entity->**user**->save();  
  
 $entity->**firstname** = $data[**'firstname'**];  
 $entity->**middlename** = $data[**'middlename'**];  
 $entity->**lastname** = $data[**'lastname'**];  
  
 **if** (**isset**($data[**'discipline'**])) {  
 $entity->**discipline\_id** = $data[**'discipline'**];  
 }  
  
 $data\_is\_change = $entity->isDirty();  
  
 $entity->save();  
  
 DB::*commit*();  
  
 **if** ($email\_is\_change || $data\_is\_change) {  
 **return** $entity;  
 } **else** {  
 **return** [**'errors'** => [**'no\_chenge'** => **'Изменений нет'**]];  
 }  
 }  
  
 **public function** validate($data, $id)  
 {  
 $validator = Validator::*make*($data,  
 [  
 **'email'** =>  
 [  
 **'nullable'**,  
 **'email'**,  
 **'max:255'**,  
 Rule::*unique*(**'users'**)->ignore($id)  
 ],  
 **'login'** =>  
 [  
 **'max:255'**,  
 Rule::*unique*(**'users'**)->ignore($id)  
 ],  
 **'firstname'** => **'required|string|max:255'**,  
 **'middlename'** => **'max:255'**,  
 **'lastname'** => **'required|string|max:255'** ],  
 [  
 **'email.unique'** => **'Почта занята'**,  
 **'login.unique'** => **'Логин занят'**,  
 **'firstname.required'** => **'Необходимо указать имя'**,  
 **'firstname.max'** => **'Поле не должно быть больше 255 символов'**,  
 **'middlename.max'** => **'Поле не должно быть больше 255 символов'**,  
 **'lastname.required'** => **'Необходимо указать фамилию'**,  
 **'lastname.max'** => **'Поле не должно быть больше 255 символов'**,  
 ]);  
  
 **if** ($validator->fails()) {  
 **return** [**'errors'** => $validator->errors()];  
 }  
 }  
  
 **public function** change\_password($id, $data)  
 {  
 $entity = $this->find($id);  
  
 **if** ($err = $this->validate\_password($data)) {  
 **return** $err;  
 }  
  
 **if** (Hash::*check*($data[**'password'**], $entity->**user**->**password**)) {  
 $entity->**user**->**password** = Hash::*make*($data[**'password-new'**]);  
 $entity->**user**->save();  
 **return** $entity;  
 }  
 **return** [**'errors'** => [**'password'** => **'Неправильный пароль'**]];  
 }  
  
 **public function** validate\_password($data)  
 {  
 $validator = Validator::*make*($data,  
 [  
 **'password'** => **'required|string|min:6'**,  
 **'password-new'** => **'required|string|min:6|confirmed'**,  
 ],  
 [  
 **'password.required'** => **'Укажите пароль'**,  
 **'password.min'** => **'Пароль должен содержать как минимум 6 символов'**,  
 **'password-new.required'** => **'Укажите новый пароль'**,  
 **'password-new.min'** => **'Пароль должен содержать как минимум 6 символов'**,  
 **'password-new.confirmed'** => **'Новый пароль и подтверждение пароля не совпадают'**,  
 ]);  
  
 **if** ($validator->fails()) {  
 **return** [**'errors'** => $validator->errors()];  
 }  
 }  
  
 **public function** kill($id)  
 {  
 $entity = $this->find($id);  
 DB::*beginTransaction*();  
  
 $entity->user()->delete();  
  
 DB::*commit*();  
 }  
  
 *//Eloquent: Relationships* **public function** user()  
 {  
 **return** $this->belongsTo(**'App\User'**);  
 }  
  
 **public function** discipline()  
 {  
 **return** $this->belongsTo(**'App\Discipline'**);  
 }  
  
 **public function** tasks()  
 {  
 **return** $this->hasMany(**'App\Task'**);  
 }  
  
 **public function** tests()  
 {  
 **return** $this->hasMany(**'App\Test'**);  
 }  
  
 **public function** materials()  
 {  
 **return** $this->hasMany(**'App\Material'**);  
 }  
  
 **public function** works()  
 {  
 **return** $this->hasMany(**'App\Work'**);  
 }  
}

User.php

**<?php  
  
namespace** App;  
  
**use** Illuminate\Notifications\Notifiable;  
**use** Illuminate\Foundation\Auth\User **as** Authenticatable;  
  
**class** User **extends** Authenticatable  
{  
 **use** Notifiable;  
  
 */\*\*  
 \* The attributes that are mass assignable.  
 \*  
 \** ***@var*** *array  
 \*/* **protected $fillable** = [  
 **'role'**, **'login'**, **'email'**, **'password'** ];  
  
 */\*\*  
 \* The attributes that should be hidden for arrays.  
 \*  
 \** ***@var*** *array  
 \*/* **protected $hidden** = [  
 **'password'**, **'remember\_token'**,  
 ];  
  
 *//Eloquent: Relationships* **public function** schoolkid()  
 {  
 **return** $this->hasOne(**'App\Schoolkid'**);  
 }  
  
 **public function** teacher()  
 {  
 **return** $this->hasOne(**'App\Teacher'**);  
 }  
  
}

Work.php

**<?php  
  
namespace** App;  
  
**use** Illuminate\Database\Eloquent\Model;  
**use** Illuminate\Support\Facades\DB;  
**use** Validator;  
  
**class** Work **extends** Model  
{  
 **protected $fillable** = [  
 **'teacher\_id'**, **'theme'** ];  
  
 **public function** getAllPaginated()  
 {  
 $entities = $this->orderBy(**'id'**, **'desc'**)->paginate(10);  
 **return** $entities;  
 }  
  
 **public function** getDisciplineWorksPaginated($discipline\_id)  
 {  
 $entities = $this  
 ->join(**'teachers'**, **'works.teacher\_id'**, **'='**, **'teachers.id'**)  
 ->join(**'disciplines'**, **'teachers.discipline\_id'**, **'='**, **'disciplines.id'**)  
 ->where(**'disciplines.id'**, $discipline\_id)  
 ->select(**'works.\*'**)  
 ->orderBy(**'works.id'**, **'desc'**)  
 ->paginate(10);  
 **return** $entities;  
 }  
  
 **public function** getTeacherWorks($teacher\_id)  
 {  
 $entities = $this  
 ->join(**'teachers'**, **'works.teacher\_id'**, **'='**, **'teachers.id'**)  
 ->join(**'disciplines'**, **'teachers.discipline\_id'**, **'='**, **'disciplines.id'**)  
 ->where(**'teachers.id'**, $teacher\_id)  
 ->select(**'works.\*'**)  
 ->orderBy(**'works.id'**, **'desc'**)  
 ->get();  
 **return** $entities;  
 }  
  
 **public function** getOne($id)  
 {  
 $entity = $this->find($id);  
 **return** $entity;  
 }  
  
 **public function** store($data, $teacher\_id)  
 {  
 **if** ($err = $this->validate($data)) {  
 **return** $err;  
 }  
  
 $newWork = [  
 **'teacher\_id'** => $teacher\_id,  
 **'theme'** => $data[**'theme'**],  
 ];  
 $this->fill($newWork);  
 $this->save();  
 **return** $this;  
 }  
  
 **public function** edit($id, $user\_id, $data)  
 {  
 **if** ($err = $this->validate($data)) {  
 **return** $err;  
 }  
  
 $entity = $this->find($id);  
  
 $entity->**teacher\_id** = $user\_id;  
 $entity->**theme** = $data[**'theme'**];  
  
 $entity->save();  
 **return** $entity;  
 }  
  
 **public function** validate($data)  
 {  
 $validator = Validator::*make*($data,  
 [  
 **'theme'** => **'required'**,  
 ],  
 [  
 **'theme.required'** => **'Необходимо указать тему'**,  
 ]);  
  
 **if** ($validator->fails()) {  
 **return** [**'errors'** => $validator->errors()];  
 }  
 }  
  
 **public function** kill($id)  
 {  
 $entity = $this->find($id);  
 **return** $entity->delete();  
 }  
  
 *//Eloquent: Relationships* **public function** teacher()  
 {  
 **return** $this->belongsTo(**'App\Teacher'**);  
 }  
  
 **public function** tasks()  
 {  
 **return** $this->belongsToMany(**'App\Task'**);  
 }  
  
 **public function** tests()  
 {  
 **return** $this->belongsToMany(**'App\Test'**);  
 }  
  
 **public function** materials()  
 {  
 **return** $this->belongsToMany(**'App\Material'**);  
 }  
  
 **public function** schoolkids()  
 {  
 **return** $this->belongsToMany(**'App\Schoolkid'**, **'homeworks'**)  
 ->withPivot(**'date\_to\_completion'**)  
 ->withTimestamps();  
 }  
  
 **public function** homeworks()  
 {  
 **return** $this->hasMany(**'App\Homework'**);  
 }  
}