# **ВВЕДЕНИЕ**

За последние годы развитие информационных технологий достигло такого уровня, что уже трудно представить жизнь без них. Все организации переходят на автоматизированные системы управления. Системы управления базами данных помогают организовывать и вести базы данных. Это, в свою очередь, позволяет им увеличить безопасность хранения данных, получать более быстрый и доступный поиск информации.

Школа не является исключением из этих организаций. Для любой школы необходимо иметь четко структурированную и упорядоченную базу данных. Для удобства учителей, а также учеников, в школах вводят электронные журналы, онлайн расписание и т.д. Все эти атрибуты и образуют единый информационный прототип школы.

Целью курсового проекта является реализация системы управления базой данных школы, позволяющей хранить и обрабатывать данные.

Задачи курсового проекта:

- исследование предметной области

- определение функциональных требований системы

- построение макетов прототипа системы

- построение диаграмм вариантов использования

- построение схемы базы данных в виде ER-модели;

- разработка и представление прототипа информационной системы;

- оформление текста пояснительной записки к курсовому проекту.

# **ГЛАВА 1. ПРЕДМЕТНАЯ ОБЛАСТЬ**

## 1.1 Исследование предметной области

Объектом исследования данной работы является школа. Данная информационная система будет обрабатывать информацию и хранить ее, что в свою очередь поможет повысить эффективность работы школы, а именно быстрое получение данных, более высокая безопасность хранения данных, удобность получения информации. Система будет хранить данные о преподавателях, преподаваемых ими дисциплинах, а также информацию об учениках и получаемых ими оценках. Учителя имеют право ставить ученикам оценки и составлять расписание. Также имеется бухгалтер, который назначает зарплату работникам школы. Администратор данной системы следит за ее работоспособностью и корректностью, вносит и изменяет данные.

В интернете имеются автоматизированные системы управления школой

Вот некоторые из них:

**1С**  
Программный комплекс «1С:Управление школой» предоставляет следующие возможности:

* систематизация данных о сотрудниках и учащихся;
* оперативный сбор, учет и анализ результатов учебной деятельности учащихся;
* автоматизация вопросов планирования и организации учебного процесса;
* управление административно-финансовой и хозяйственной деятельностью;
* автоматизация библиотечной деятельности;
* организация учета питания.

**АВЕРС**

Компания АВЕРС, уже работающая 15 лет на рынке программного обеспечения, представляет комплекс программного обеспечения для управленческой, административной и финансово-хозяйственной деятельности образовательных учреждений, органов управления образованием муниципального и регионального уровня.

Сетевые программы для учреждений образования позволяют создавать автоматизированные рабочие места практически для всех работников: директор, заместители директора, классные руководители, секретарь, библиотекарь, бухгалтер, учителя, медработник, психолог и др.

Программные продукты АРМ "Директор", "Расписание", Библиотека", "Тарификация" работают на единой базе данных. Простые настройки, интуитивно понятный интерфейс, широчайшие возможности существенно сокращают временные затраты на поиск, систематизацию информации, формирование и вывод отчетов, таблиц, графиков (как стандартизированных так и свободно создаваемых) в ряд популярных приложений.   
Комплекс включает в себя базу данных сотрудников, учащихся, учебный план, ЕГЭ, расписание, тарификация, библиотечный фонд и многое другое. Представлено более 800 параметров на выбор, автозагрузка данных, интеграция с другими программными продуктами.

**Автоматизация управления школой**   
Программный пакет "1С:ХроноГраф Школа 2.5 ПРОФ" - это многофункциональная система, работающая как основа для формирования единого информационного пространства учреждения образования.   
Программа предоставляет широкие возможности для:

* создания базовой информации, включая информацию общего доступа и периодизированных компонент;
* автоматизации кадровой работы;
* систематизации данных об учащихся;
* администрирования учебно-воспитательного процесса;
* поддержки содержания образования;
* автоматизации финансовой и хозяйственной деятельности образовательного учреждения.

Функции подготовки отчетной документации позволяют формировать итоговые и статистические отчеты как произвольные, так и унифицированные формы Госкомстата РФ. Также в программе предоставляются возможности для самостоятельного конструирования отчетов. Для оптимизации поиска данных по базе учреждения в программе реализован отдельный интерфейс поиска по ключевым словам с возможностью определения области поиска.

Сравним вышеперечисленные систему управления школой.

## 1.2 Функциональные требования

Данная программа предназначена для автоматизации учебного процесса в школе она выполняет следующие функции:

* Систематизация данных об учениках, родителях и преподавателях школы
* Составление расписания для преподавателей и учеников
* Ведения учета заработной платы учителей
* Ведения учета оценок учеников
* Обеспечение доступа к программе родителей, преподавателей, учеников, бухгалтера и администратора.
* Ограничение прав доступа к информации пользователей разных ролей.

Будущие пользователи системы:

* Учителя (Права проставления и изменения оценок и составления и изменения расписания а также просмотра информации об учениках, учителях зарплате, расписании и оценках)
* Родители (Права просмотра информации об успеваемости учеников, списка учеников и учителей, а также их личной информации)
* Ученики (Права просмотра оценок, расписания, и списка учителей с их личной информацией)
* Бухгалтер (Права добавления и изменения заработной платы учителей, просмотр списка учителей и расписания)
* Администратор (Полные права)

## 1.3 Роли системы

* Учитель (Права на добавление и изменение оценок ученикам, изменения и добавления расписания, просмотр информации в БД) – имеет доступ к добавлению и изменению оценок ученикам, составления и изменения расписания, просмотр списка учеников, списка учителей, расписания, списка зарплаты, списка оценок.
* Родитель (Права на просмотр расписания ученика списка оценок, списка учителей, списка учеников)- имеет право на просмотр информации об учениках, учителях, оценках и расписании
* Ученик (права на просмотр расписания и списка оценок) – имеет доступ к просмотру информации об учениках, учителях, оценках, расписании.
* Бухгалтер (Права на добавление и изменения заработной платы учителей) – имеет права добавления и изменения заработной платы учителей, просмотр информации об учителях, расписании, зарплате.
* Администратор (Полные права) – имеет полные права, может просматривать, а также редактировать информацию (Список учителей, учеников, расписание, список зарплаты, список оценок)

## 1.4 Макеты прототипа системы

1. Форма страницы входа в систему. После ввода логина и пароля, происходит вход в систему и открывается следующая главная форма.

Рисунок 1. Стартовая форма.

2. Форма главной страницы (Вкладки Учителя, Расписание и тд. Аналогичны этой)

Данная форма содержит все основные данные системы в зависимости от роли их можно добавлять или изменять. Данная форма открывается при роли – Администратор.

Рисунок 2. Главная страница (роль Администратор)

Ниже представлены главные страницы для остальных ролей

Рисунок 3. Главная страница (роль Ученик также форма аналогична для роли Родитель).

Рисунок 4. Главная страница (роль Бухгалтер)

Рисунок 5. Главная страница (Роль учитель).

3. Следующие формы – это формы изменения и добавления информации. В зависимости от роли пользователь может изменять и удалять данные. Ниже представлены скриншоты форм.

Рисунок 6. Форма добавления ученика.

Рисунок 7. Форма изменения ученика (Остальные формы изменения аналогичны формам добавления. Единственное отличе в том, что формы изменения уже заполнены данными).

Рисунок 8. Форма добавления родителя.

Рисунок 9. Форма добавления учителя.

Рисунок 10. Форма добавления расписания

Рисунок 11. Форма добавления предмета.

Рисунок 12. Форма добавления зарплаты.

Рисунок 13. Форма добавления оценки.

1.5 Диаграмма вариантов использования

Рисунок 1. Диаграмма вариантов использования. (Бухгалтер, Администратор)

**Вариант использования "Войти в систему"**

*Краткое описание:*

Данный вариант использования описывает вход пользователя в систему регистрации курсов.

*Основной поток событий:*

Данный вариант использования начинает выполняться, когда пользователь хочет войти в систему регистрации курсов.

Система запрашивает имя пользователя и пароль.

Пользователь вводит имя и пароль.

Система подтверждает имя и пароль, после чего открывается доступ в систему.

*Альтернативные потоки:*

Неправильное имя/пароль: Если во время выполнения основного потока обнаружится, что пользователь ввел неправильное имя и/или пароль, система выводит сообщение об ошибке. Пользователь может вернуться к началу ос­новного потока или отказаться от входа в систему, при этом выполнение варианта использования завершается.

**Вариант использования «Изменение зарплаты»**

*Краткое описание:*

Данный вариант использования позволяет изменять информацию о зарплате.

*Основной поток событий:*

Данный вариант использования начинает выполняться, когда пользователь хочет изменить информацию о заработной плате.

1. Пользователь открывает окно зарплаты.
2. Пользователь выбирает необходимого учителя и нажимает кнопку изменить
3. Пользователь изменяет данные
4. Пользователь выбирает сохранить после чего данные сохраняются.

*Альтернативные потоки:*

1. Пользователь пытается ввести сумму зарплаты меньше 9000. В данном случае, система не даст поставить зарплату ниже 9000.
2. Пользователь пытается ввести рабочие часы меньше одного или больше 160. В таком случае система не даст пользователю выставить ниже 1 часа и выше 160 соответственно.

*Предусловия:*

Пользователь должен войти в систему как Бухгалтер или Администратор.

**Вариант использования «Просмотр зарплаты»**

*Краткое описание:*

Данный вариант использования позволяет просматривать информацию о зарплате.

*Основной поток событий:*

Данный вариант использования начинает выполняться, когда пользователь желает посмотреть информацию о зарплате.

1. Пользователь открывает окно графика зарплат.
2. Система отображает в окне данные по зарплате

*Предусловия:*

Пользователь должен войти в систему как Учитель, Администратор или Бухгалтер.

**Вариант использования «Добавление зарплаты»**

*Краткое описание:*

Данный вариант использования позволяет добавлять информацию о зарплате.

*Основной поток событий:*

Данный вариант использования начинает выполняться, когда пользователь желает добавить информацию о зарплате.

1. Пользователь открывает окно зарплаты.
2. Пользователь нажимает кнопку добавить
3. Пользователь вводит данные
4. Пользователь выбирает сохранить после чего данные сохраняются.

*Альтернативные потоки:*

1. Пользователь пытается ввести сумму зарплаты меньше 9000. В данном случае, система не даст поставить зарплату ниже 9000.
2. Пользователь пытается ввести рабочие часы меньше одного или больше 160. В таком случае система не даст пользователю выставить ниже 1 часа и выше 160 соответственно.
3. Пользователь ввел неполные данные. В этом случае система выдаст ошибку.

*Предусловия:*

Пользователь должен войти в систему как Администратор или Бухгалтер.

**Вариант использования «Просмотр учителей»**

*Краткое описание:*

Данный вариант использования позволяет просматривать информацию о учителях.

*Основной поток событий:*

Данный вариант использования начинает выполняться, когда пользователь хочет посмотреть информацию о учителях.

1. Пользователь открывает окно списка учителей.
2. Система отображает в окне список учителей.

*Предусловия:*

Пользователь должен войти в систему как Родитель, Бухгалтер, Ученик, Администратор, Учитель.

**Вариант использования «Добавление учителей»**

*Краткое описание:*

Данный вариант использования позволяет добавлять в базу данных новых учителей

*Основной поток событий:*

Данный вариант использования начинает выполняться, когда пользователь хочет добавить нового учителя.

1. Пользователь открывает окно списка учителей.
2. Пользователь нажимает кнопку добавить
3. Пользователь вводит необходимые данные.
4. Система добавляет данные о новом учителе в базу данных.

*Альтернативные потоки:*

1. Пользователь ввел неполные данные. В таком случае, система выдаст ошибку и попросит ввести данные до конца.
2. Возраст учителя меньше двадцати лет. В таком случае система выдаст ошибку о том, что учителю нет 20 лет.

*Предусловия:*

Пользователь должен войти в систему как Администратор.

**Вариант использования «Изменение учителей»**

*Краткое описание:*

Данный вариант использования позволяет изменять в базе данных информацию об учителях

*Основной поток событий:*

Данный вариант использования начинает выполняться, когда пользователь хочет изменить информацию об учителе.

1. Пользователь открывает окно списка учителей.
2. Пользователь нажимает кнопку изменить
3. Пользователь вводит необходимые данные.
4. Система изменяет данные о учителе в базе данных.

*Альтернативные потоки:*

1. Пользователь ввел неполные данные. В таком случае, система выдаст ошибку и попросит ввести данные до конца.
2. Возраст учителя меньше двадцати лет. В таком случае система выдаст ошибку о том, что учителю нет 20 лет.

*Предусловия:*

Пользователь должен войти в систему как Администратор.

**Вариант использования «Просмотр расписания»**

*Краткое описание:*

Данный вариант использования позволяет просматривать информацию о расписании.

*Основной поток событий:*

Данный вариант использования начинает выполняться, когда пользователь хочет посмотреть информацию о расписании.

1. Пользователь открывает окно расписания.
2. Система отображает в окне расписание.

*Предусловия:*

Пользователь должен войти в систему как Родитель, Ученик, Учитель, Администратор.

**Вариант использования «Изменение расписания»**

*Краткое описание:*

Данный вариант использования позволяет изменять информацию о расписании.

*Основной поток событий:*

Данный вариант использования начинает выполняться, когда пользователь хочет изменить информацию о расписании.

1. Пользователь открывает окно расписания предметов.
2. Пользователь нажимает кнопку изменить
3. Пользователь изменяет данные и нажимает кнопку сохранить
4. Система сохраняет информацию в БД

*Предусловия:*

Пользователь должен войти в систему как Администратор или Учитель.

**Вариант использования «Добавление расписания»**

*Краткое описание:*

Данный вариант использования позволяет добавлять информацию о расписании.

*Основной поток событий:*

Данный вариант использования начинает выполняться, когда пользователь хочет добавить информацию о расписании.

1. Пользователь открывает окно расписания предметов.
2. Пользователь нажимает кнопку добавить
3. Пользователь вводит данные и нажимает кнопку сохранить
4. Система сохраняет информацию в БД

*Альтернативные потоки:*

1. Пользователь ввел неполные данные. В таком случае, система выдаст ошибку и попросит ввести данные до конца.
2. Дата добавления меньше текущей. В таком случае система выдаст ошибку и закроет окно добавления расписания.

*Предусловия:*

Пользователь должен войти в систему как Администратор или Учитель.

**Вариант использования «Просмотр оценок»**

*Краткое описание:*

Данный вариант использования позволяет просматривать информацию об оценках.

*Основной поток событий:*

Данный вариант использования начинает выполняться, когда пользователь хочет посмотреть информацию об оценках.

1. Пользователь открывает окно списка оценок.
2. Система отображает в окне список оценок.

*Предусловия:*

Пользователь должен войти в систему как Родитель, Ученик, Администратор, Учитель.

**Вариант использования «Изменение оценок»**

*Краткое описание:*

Данный вариант использования позволяет изменять информацию об оценках учеников.

*Основной поток событий:*

Данный вариант использования начинает выполняться, когда пользователь хочет изменить информацию об оценках.

1. Пользователь открывает список оценок.
2. Пользователь нажимает кнопку изменить.
3. Пользователь изменяет данные и нажимает кнопку сохранить.
4. Система сохраняет информацию в БД.

*Предусловия:*

Пользователь должен войти в систему как Администратор или Учитель.

**Вариант использования «Добавление оценок»**

*Краткое описание:*

Данный вариант использования позволяет добавлять информацию об оценках учеников.

*Основной поток событий:*

Данный вариант использования начинает выполняться, когда пользователь хочет добавить информацию об оценках.

1. Пользователь открывает список оценок.
2. Пользователь нажимает кнопку изменить.
3. Пользователь изменяет данные и нажимает кнопку сохранить.
4. Система сохраняет информацию в БД.

*Альтернативные потоки:*

1. Пользователь ввел неполные данные. В таком случае система выдаст ошибку.

*Предусловия:*

Пользователь должен войти в систему как Администратор или Учитель.

**Вариант использования «Добавление родителей»**

*Краткое описание:*

Данный вариант использования позволяет добавлять в базу данных информацию о новых родителей

*Основной поток событий:*

Данный вариант использования начинает выполняться, когда пользователь хочет добавить нового родителя.

1. Пользователь открывает окно списка учеников.
2. Пользователь нажимает кнопку добавить.
3. В открывшемся окне пользователь нажимает кнопку добавить родителя.
4. Пользователь вводить данные и нажимает кнопку сохранить.
5. Система сохраняет данные.

*Альтернативные потоки:*

1. Пользователь ввел не все основные данные. В таком случае, система выдаст ошибку и попросит ввести данные до конца.

*Предусловия:*

Пользователь должен войти в систему как Администратор.

**Вариант использования «Добавление учеников»**

*Краткое описание:*

Данный вариант использования позволяет добавлять в базу данных информацию о новых учениках

*Основной поток событий:*

Данный вариант использования начинает выполняться, когда пользователь хочет добавить нового ученика.

1. Пользователь открывает список учеников.
2. Пользователь нажимает кнопку добавить.
3. Пользователь вводит необходимые данные и нажимает кнопку сохранить.
4. Система добавляет данные о новом ученике в базу данных.

*Альтернативные потоки:*

1. Пользователь ввел не все основные данные. В таком случае, система выдаст ошибку и попросит ввести данные до конца.
2. Пользователь ввел дату рождения ученика меньше 6 лет. В таком случае система выдаст ошибку.

*Предусловия:*

Пользователь должен войти в систему как Администратор.

**Вариант использования «Изменение учеников»**

*Краткое описание:*

Данный вариант использования позволяет изменять в базе данных информацию об учениках

*Основной поток событий:*

Данный вариант использования начинает выполняться, когда пользователь хочет изменить информацию об учениках.

1. Пользователь открывает список учеников.
2. Пользователь нажимает кнопку изменить.
3. Пользователь вводит необходимые данные и нажимает кнопку сохранить.
4. Система добавляет данные об ученике в базу данных.

*Альтернативные потоки:*

1. Пользователь ввел не все основные данные. В таком случае, система выдаст ошибку и попросит ввести данные до конца.
2. Пользователь ввел дату рождения ученика меньше 6 лет. В таком случае система выдаст ошибку.

*Предусловия:*

Пользователь должен войти в систему как Администратор.

**Вариант использования «Добавление предмета»**

*Краткое описание:*

Данный вариант использования позволяет добавлять в базу данных информацию о новых предметах

*Основной поток событий:*

Данный вариант использования начинает выполняться, когда пользователь хочет добавить новый предмет.

1. Пользователь открывает окно расписания.
2. Пользователь нажимает кнопку добавить.
3. Пользователь нажимает на кнопку добавить новый предмет.
4. Пользователь вводит необходимые данные и нажимает кнопку сохранить.
5. Система добавляет данные о новом предмете в базу данных.

*Альтернативные потоки:*

1. Пользователь ввел не все основные данные. В таком случае, система выдаст ошибку и попросит ввести данные до конца.

*Предусловия:*

Пользователь должен войти в систему как Администратор или Учитель.

**Вариант использования «Просмотр учеников»**

*Краткое описание:*

Данный вариант использования позволяет просмотреть информацию об учениках

*Основной поток событий:*

Данный вариант использования начинает выполняться, когда пользователь хочет просмотреть информацию об учениках.

1. Пользователь открывает окно списка учеников.
2. Система отображает список учеников.

*Предусловия:*

Пользователь должен войти в систему как Администратор, Учитель, Ученик, Родитель.

## 1.6 ER-Модель системы

Рисунок 1. ER-Модель системы.

Определяются первичные и внешние ключи для базы данных:

Таблица 1.1 – Сущность «Родитель»

Таблица 1.2 – Сущность «Ученик»

Таблица 1.3 – Сущность «Зарплата»

Таблица 1.4 – Сущность «Учитель»

Таблица 1.5 – Сущность «Предмет»

Таблица 1.6 – Сущность «Расписание»

Таблица 1.7 – Сущность «Оценка»

# **ГЛАВА 2. РАЗРАБОТКА ПРОТОТИПА СИСТЕМЫ**

## 2.1 Создание базы данных

Первым делом создаем базу данных в моем случае это программа pgAdmin4. Код базы данных:

Создаем таблицы:

CREATE TABLE Subjects(

Code\_subject serial NOT NULL primary key,

Name\_subj varchar(50) NOT NULL);

CREATE TABLE Parents(

Code\_parent serial NOT NULL primary key,

Surname\_parent varchar(50) NOT NULL,

Name\_parent varchar(50) NOT NULL,

Lastname\_parent varchar(50),

Adress varchar(50) NOT NULL);

CREATE TABLE Students(

Code\_stud serial NOT NULL primary key,

Surname\_stud varchar(50) NOT NULL,

Name\_stud varchar(50) NOT NULL,

Lastname\_stud varchar(50),

Birthday\_stud date NOT NULL,

Code\_parent int NOT NULL);

CREATE TABLE Lectors(

Code\_lector serial NOT NULL primary key,

Surname\_lector varchar(50) NOT NULL,

Name\_lector varchar(50) NOT NULL,

Lastname\_lector varchar(50),

Birthday\_lector date NOT NULL,

Post varchar(50) NULL);

CREATE TABLE Shedule(

Code\_shedule serial NOT NULL primary key,

Code\_stud int NOT NULL,

Code\_subject int NOT NULL,

Code\_lector int NOT NULL,

Date\_classes Date NOT NULL,

Time\_classes varchar(50) not null);

Create table Salary(

Code\_salary serial NOT NULL primary key,

Code\_lector int NOT NULL,

Work\_hour int,

Experience int default 1,

Mounth varchar (50) not null,

summ int not null);

CREATE TABLE Estimate(

Code\_est serial NOT NULL primary key,

Name\_est varchar(50) NOT NULL,

Code\_shedule int NOT NULL);

Создаем внешние ключи:

Alter Table Estimate add Foreign Key(Code\_shedule)

References Shedule(Code\_shedule);

Alter Table Students add Foreign Key(Code\_parent)

References Parents(Code\_parent);

Alter Table Salary add Foreign Key (Code\_lector)

References Lectors(Code\_lector);

Alter Table Shedule add Foreign Key (Code\_subject)

References Subjects(Code\_subject);

Alter Table Shedule add Foreign Key (Code\_stud)

References Students(Code\_stud);

Alter Table Shedule add Foreign Key (Code\_lector)

References Lectors(Code\_lector);

Заполняем данными таблицы:

Insert into parents (Surname\_parent,Name\_parent,Lastname\_parent,Adress)

Values ('Астафьева','Надежда','Ивановна','Октябрьск' );

Insert into parents (Surname\_parent,Name\_parent,Lastname\_parent,Adress)

Values ('Амирьянова','Любовь','Анатольевна','Уфа' );

Insert into parents (Surname\_parent,Name\_parent,Lastname\_parent,Adress)

Values ('Быкова','Екатерина','Викторовна','Казань' );

Insert into parents (Surname\_parent,Name\_parent,Lastname\_parent,Adress)

Values ('Валеева','Марина','Николаевна','Кукмор' );

Insert into parents (Surname\_parent,Name\_parent,Lastname\_parent,Adress)

Values ('Дроздова','Наталья','Дмитриевна','Пермь' );

Insert into parents (Surname\_parent,Name\_parent,Lastname\_parent,Adress)

Values ('Смирнова','Татьяна','Александровна','Казань' );

Insert into parents (Surname\_parent,Name\_parent,Lastname\_parent,Adress)

Values ('Суворова','Юлия','Максимовна','Казань' );

Insert into parents (Surname\_parent,Name\_parent,Lastname\_parent,Adress)

Values ('Хуснутдинова','Эльвира','Сергеевна','Стерлитамак' );

Insert into parents (Surname\_parent,Name\_parent,Lastname\_parent,Adress)

Values ('Червоненко','Людмила','Генадьевна','Сарапул' );

Insert into students (Surname\_stud,Name\_stud,Lastname\_stud,birthday\_stud,code\_parent)

Values ('Астафьев','Руслан','Викторович','12-03-1999',1);

Insert into students (Surname\_stud,Name\_stud,Lastname\_stud,birthday\_stud,code\_parent)

Values ('Амирьянова','Ангелина','Ахметбековна','23-10-1998',2);

Insert into students (Surname\_stud,Name\_stud,Lastname\_stud,birthday\_stud,code\_parent)

Values ('Быков','Максим','Владимирович','07-05-1999',3);

Insert into students (Surname\_stud,Name\_stud,Lastname\_stud,birthday\_stud,code\_parent)

Values ('Валеев','Дмитрий','Анатольевич','07-12-1998',4);

Insert into students (Surname\_stud,Name\_stud,Lastname\_stud,birthday\_stud,code\_parent)

Values ('Дроздова','Анастасия','Павловна','21-02-2000',5);

Insert into students (Surname\_stud,Name\_stud,Lastname\_stud,birthday\_stud,code\_parent)

Values ('Смирнов','Максим','Леонидович','30-06-1999',6);

Insert into students (Surname\_stud,Name\_stud,Lastname\_stud,birthday\_stud,code\_parent)

Values ('Суворов','Никита','Константинович','21-11-1998',7);

Insert into students (Surname\_stud,Name\_stud,Lastname\_stud,birthday\_stud,code\_parent)

Values ('Хуснутдинов','Азат','Гафурович','13-01-1999',8);

Insert into students (Surname\_stud,Name\_stud,Lastname\_stud,birthday\_stud,code\_parent)

Values ('Червоненко','Александр','Михайлович','01-04-1997',9);

Insert into Subjects (Name\_subj)

Values ('СИИ');

Insert into Subjects (Name\_subj)

Values ('ПИС');

Insert into Subjects (Name\_subj)

Values ('ЧМ');

Insert into Subjects (Name\_subj)

Values ('ВП');

Insert into Subjects (Name\_subj)

Values ('МО');

Insert into Subjects (Name\_subj)

Values ('СИАКОД');

Insert into Subjects (Name\_subj)

Values ('АиГ');

Insert into Subjects (Name\_subj)

Values ('БД');

Insert into Subjects (Name\_subj)

Values ('Сетевые технологии');

Insert into Subjects (Name\_subj)

Values ('ОС');

Insert into Lectors (Surname\_lector,Name\_lector,Lastname\_lector,Birthday\_lector,Post)

Values ('Мангушева','Алина','Раисовна','12-04-1989', 'Старший программист');

Insert into Lectors (Surname\_lector,Name\_lector,Lastname\_lector,Birthday\_lector,Post)

Values ('Панченко','Оксана','Владимировна','23-11-1983', 'Веб-программист');

Insert into Lectors (Surname\_lector,Name\_lector,Lastname\_lector,Birthday\_lector,Post)

Values ('Антонова','Полина','Валерьевна','01-05-1987', 'Программист по ИИ');

Insert into Lectors (Surname\_lector,Name\_lector,Lastname\_lector,Birthday\_lector,Post)

Values ('Гильмутдинов','Руслан','Фаридович','09-04-1986', 'Программист');

Insert into Lectors (Surname\_lector,Name\_lector,Lastname\_lector,Birthday\_lector,Post)

Values ('Подъяпольский','Николай','Васильевич','12-09-1970', 'Программист');

Insert into Lectors (Surname\_lector,Name\_lector,Lastname\_lector,Birthday\_lector,Post)

Values ('Кашаев','Анвар','Умидович','15-07-1975', 'Старший программист');

Insert into Lectors (Surname\_lector,Name\_lector,Lastname\_lector,Birthday\_lector,Post)

Values ('Югов','Андрей','Валерьевич','11-11-1978', 'Программист');

Insert into Lectors (Surname\_lector,Name\_lector,Lastname\_lector,Birthday\_lector,Post)

Values ('Путин','Дмитрий','Анатольевич','18-10-1974', 'Программист по ИИ');

Insert into Salary (Code\_lector,Work\_hour,Experience,mounth,summ)

Values (2,10,2,'May',25000 );

Insert into Salary (Code\_lector,Work\_hour,Experience,mounth,summ)

Values (1,12,1.5,'May',45000 );

Insert into Salary (Code\_lector,Work\_hour,Experience,mounth,summ)

Values (5,15,1,'May',20000 );

Insert into Salary (Code\_lector,Work\_hour,Experience,mounth,summ)

Values (4,10,1.5,'May',35000 );

Insert into Salary (Code\_lector,Work\_hour,Experience,mounth,summ)

Values (7,10,2,'May',30000 );

Insert into Salary (Code\_lector,Work\_hour,Experience,mounth,summ)

Values (3,10,1,'May',15000 );

Insert into Salary (Code\_lector,Work\_hour,Experience,mounth,summ)

Values (8,10,1.5,'May',32000 );

Insert into Salary (Code\_lector,Work\_hour,Experience,mounth,summ)

Values (6,10,1,'May',28000 );

Insert into shedule (Code\_stud,Code\_subject,Code\_lector,Date\_classes,Time\_classes)

Values (3,4,1,'2019-06-04','8-00');

Insert into shedule (Code\_stud,Code\_subject,Code\_lector,Date\_classes,Time\_classes)

Values (3,2,2,'2019-06-04','11-00');

Insert into shedule (Code\_stud,Code\_subject,Code\_lector,Date\_classes,Time\_classes)

Values (3,1,3,'2019-06-04','12-00');

Insert into shedule (Code\_stud,Code\_subject,Code\_lector,Date\_classes,Time\_classes)

Values (2,5,6,'2019-06-04','17-00');

Insert into shedule (Code\_stud,Code\_subject,Code\_lector,Date\_classes,Time\_classes)

Values (2,10,2,'2019-06-04','16-00');

Values (2,2,2,'2019-06-04','9-00');

Insert into shedule (Code\_stud,Code\_subject,Code\_lector,Date\_classes,Time\_classes)

Values (2,1,3,'2019-06-04','10-00');

Insert into shedule (Code\_stud,Code\_subject,Code\_lector,Date\_classes,Time\_classes)

Values (4,8,1,'2019-06-04','10-00');

Insert into shedule (Code\_stud,Code\_subject,Code\_lector,Date\_classes,Time\_classes)

Values (4,1,3,'2019-06-04','12-00');

Insert into shedule (Code\_stud,Code\_subject,Code\_lector,Date\_classes,Time\_classes)

Values (5,6,4,'2019-06-04','17-00');

Insert into shedule (Code\_stud,Code\_subject,Code\_lector,Date\_classes,Time\_classes)

Values (5,7,7,'2019-06-04','12-00');

Insert into shedule (Code\_stud,Code\_subject,Code\_lector,Date\_classes,Time\_classes)

Values (5,9,8,'2019-06-04','11-00');

Insert into shedule (Code\_stud,Code\_subject,Code\_lector,Date\_classes,Time\_classes)

Values (6,2,2,'2019-06-04','15-00');

Insert into shedule (Code\_stud,Code\_subject,Code\_lector,Date\_classes,Time\_classes)

Values (6,3,4,'2019-06-04','11-00');

Insert into shedule (Code\_stud,Code\_subject,Code\_lector,Date\_classes,Time\_classes)

Values (6,10,2,'2019-06-04','8-00');

Insert into shedule (Code\_stud,Code\_subject,Code\_lector,Date\_classes,Time\_classes)

Values (7,8,2,'2019-06-04','13-00');

Insert into shedule (Code\_stud,Code\_subject,Code\_lector,Date\_classes,Time\_classes)

Values (7,1,1,'2019-06-04','17-00');

Insert into shedule (Code\_stud,Code\_subject,Code\_lector,Date\_classes,Time\_classes)

Values (8,3,4,'2019-06-04','10-00');

Insert into shedule (Code\_stud,Code\_subject,Code\_lector,Date\_classes,Time\_classes)

Values (8,6,5,'2019-06-04','14-00');

Insert into shedule (Code\_stud,Code\_subject,Code\_lector,Date\_classes,Time\_classes)

Values (9,1,3,'2019-06-04','15-00');

Insert into shedule (Code\_stud,Code\_subject,Code\_lector,Date\_classes,Time\_classes)

Values (9,4,1,'2019-06-04','12-00');

Insert into shedule (Code\_stud,Code\_subject,Code\_lector,Date\_classes,Time\_classes)

Values (9,10,2,'2019-06-04','9-00');

Insert into estimate (Name\_est,Code\_shedule)

Values ('5',2);

Insert into estimate (Name\_est,Code\_shedule)

Values ('3',5);

## 2.2 Вход в систему

После запуска программы открывается стартовое окно, в котором необходимо ввести логин и пароль. Для каждой из 5 ролей свой логин и пароль. Администратор – логин admin пароль admin. Бухгалтер – логин buh пароль buh. Учитель - логин teach пароль teach. Ученик – логин stud пароль stud. Родитель – логин par пароль par.

Данный метод проверяет логин и пароль и в случае его совпадения выполняет вход в систему.

Рисунок 1. Код проверки логина и пароля.

## 2.3 Основная форма

После запуска основной формы в зависимости от роли скрываются кнопки и окна для пользователей.

Рисунок 2. Скрытие кнопок.

Затем идет заполнение данными таблиц из базы данных (Пример – Расписание)

Рисунок 3. Заполнение таблиц данными.

## 2.4 Формы добавления и изменения информации

При желании пользователя добавить информацию в БД открывается новая форма, которая сохраняет информацию в базу данных.

Рисунок 4. Добавление данных о новом предмете.

При желании пользователя изменить информацию в БД открывается новая форма, которая изменяет и сохраняет информацию в базу данных.

Рисунок 5. Изменение данных о ученике.

# **ЗАКЛЮЧЕНИЕ**

В результате выполнения курсового проекта был разработан прототип информационной системы «Школа». Было проведено тестирование и проверка программы.

В результате физического проектирования программа позволяет добавлять и изменять данные, что в свою очередь, помогает пользователям данной системы быстрее получать информацию, более надежное и удобное ее хранение.

Самым главным в программе является эффективность использования данной среды для получения всей необходимой информации пользователям в самые короткие сроки, тем самым уменьшая временные затраты, а также увеличивая безопасность хранения данных.

# **СПИСОК ЛИТЕРАТУРЫ**

1. Мамиконов, А.Г. Методы разработки автоматизированных систем управления. - М.: Энергия,2015. - 336 с.
2. Малыхина, М.П. Базы данных: Основы, проектирование, использование / М. П. Малыхина –  СПб.: БХВ-Петербург, 2017  512 с.: ил.
3. «MySQL, 3-е издание», Поль Дюбуа – СПб.: Питер 2006.
4. «MySQL 5: базовый курс», Роберт Шелдон, Джоффрей Мойе.
5. «MySQL. Руководство администратора», Давид Уолл, Вильямс 2008.
6. C# разработка компонентов в MS Visual Studio. / Автор: Агуров П. Год издания 2008, 480 страниц.
7. [http://msdn.microsoft.com/ru-ru/library/bb383962(v=vs.90).aspx](https://infourok.ru/go.html?href=http%3A%2F%2Fmsdn.microsoft.com%2Fru-ru%2Flibrary%2Fbb383962%28v%3Dvs.90%29.aspx) Интерактивный учебник по Visual C#.
8. Павловская Т.А. C#. Программирование на языке высокого уровня. Учебник для вузов. – СПб.: Питер, 2007. – 432с
9. Трей Нэш, C# 2010. Ускоренный курс для профессионалов: Пер. с- англ. [Текст]/ Трей Нэш – М.: ООО «И. Д. Вильямс», 2010. – 592 с. ил.
10. Агуров П. В., C#. Сборник рецептов. [Текст]/ П. В. Агуров – СПб.: БХВ-Петербург, 2007. – 432 с. ил.