**Краткое описание предметной области.**

Для колледжа необходимо реализовать ИС «Учет обучающихся колледжа». В колледже обучаются студенты по разным направлениям в разных группах и на разных курсах. Помимо учета студентов, необходимо реализовать учет абитуриентов, которые по прошествии времени становятся студентам. Необходимо автоматизировать работу с документами (каждый студент может запросить получение справки), с данными студента, в том числе изображениями. Работать с ИС будут 2 категории пользователей: сотрудник учебной части, который будет вести учет студентов и абитуриентов, а также администратор, который занимается ведением учета других сотрудников.

Технически база данных должна храниться удаленно на сервере, а сотрудники с помощью клиентского приложения подключаться к ней.

**Техническое задание.**

**Спроектируйте ИС «Учет обучающихся колледжа»**. Для этого реализуйте следующие диаграммы:

- Диаграмма прецедентов (Use-Case);

- Диаграмма последовательности;

- Диаграмма развертывания;

- ER-диаграмма;

- User Story Mapping.

В соответствии с рисунком 1 представлен пример диаграммы прецедентов:



Рисунок 1 – Пример диаграммы прецедентов

В соответствии с рисунком 2 представлен пример диаграммы последовательности:

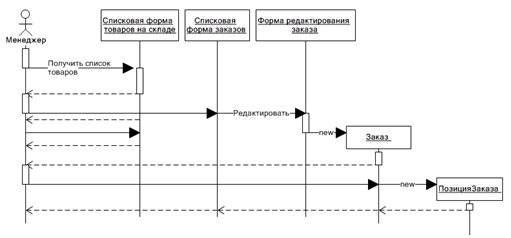


Рисунок 2 – Пример диаграммы последовательности

В соответствии с рисунком 3 представлен пример диаграммы развертывания:

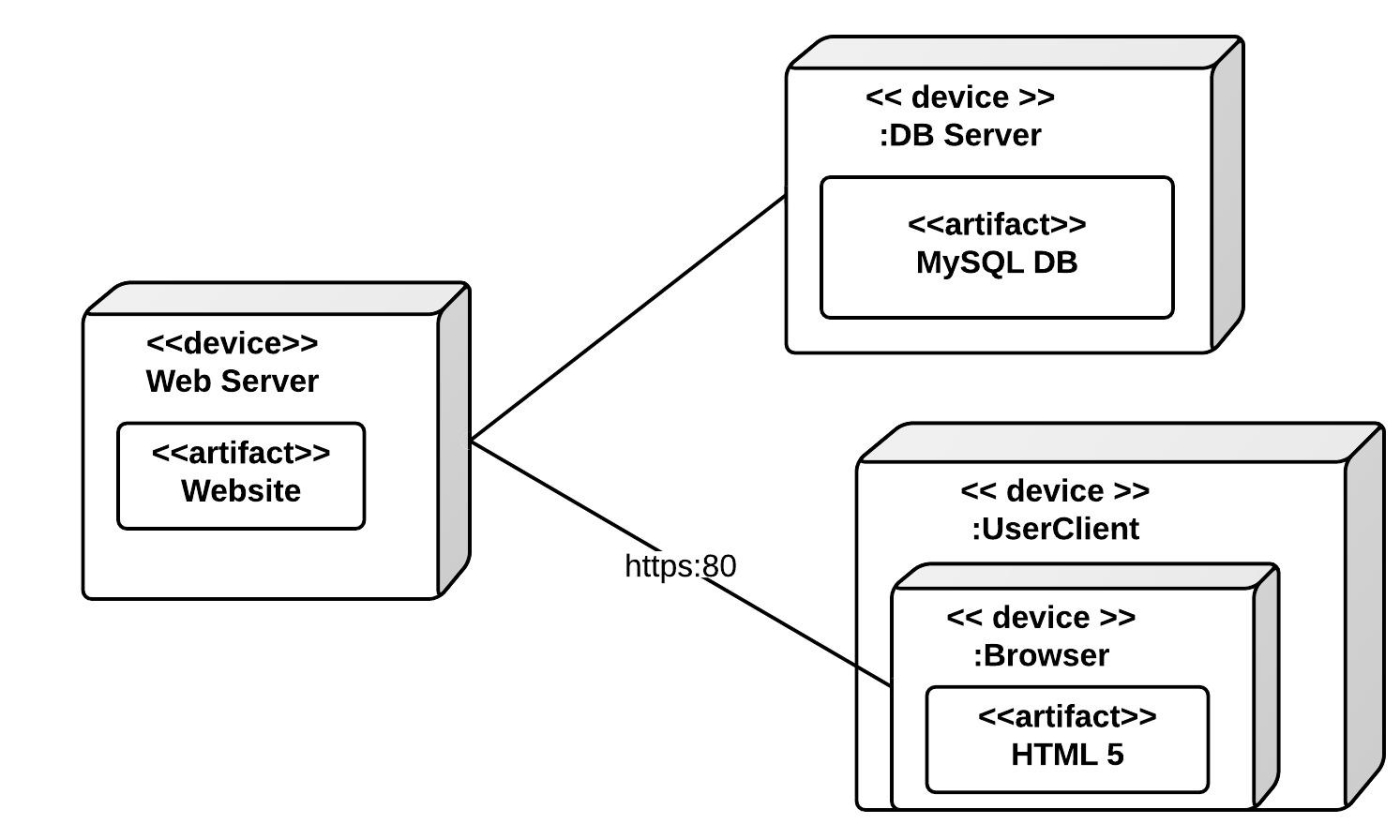


Рисунок 3 – Пример диаграммы развертывания

В соответствии с рисунком 4 представлен пример ER-диаграммы:

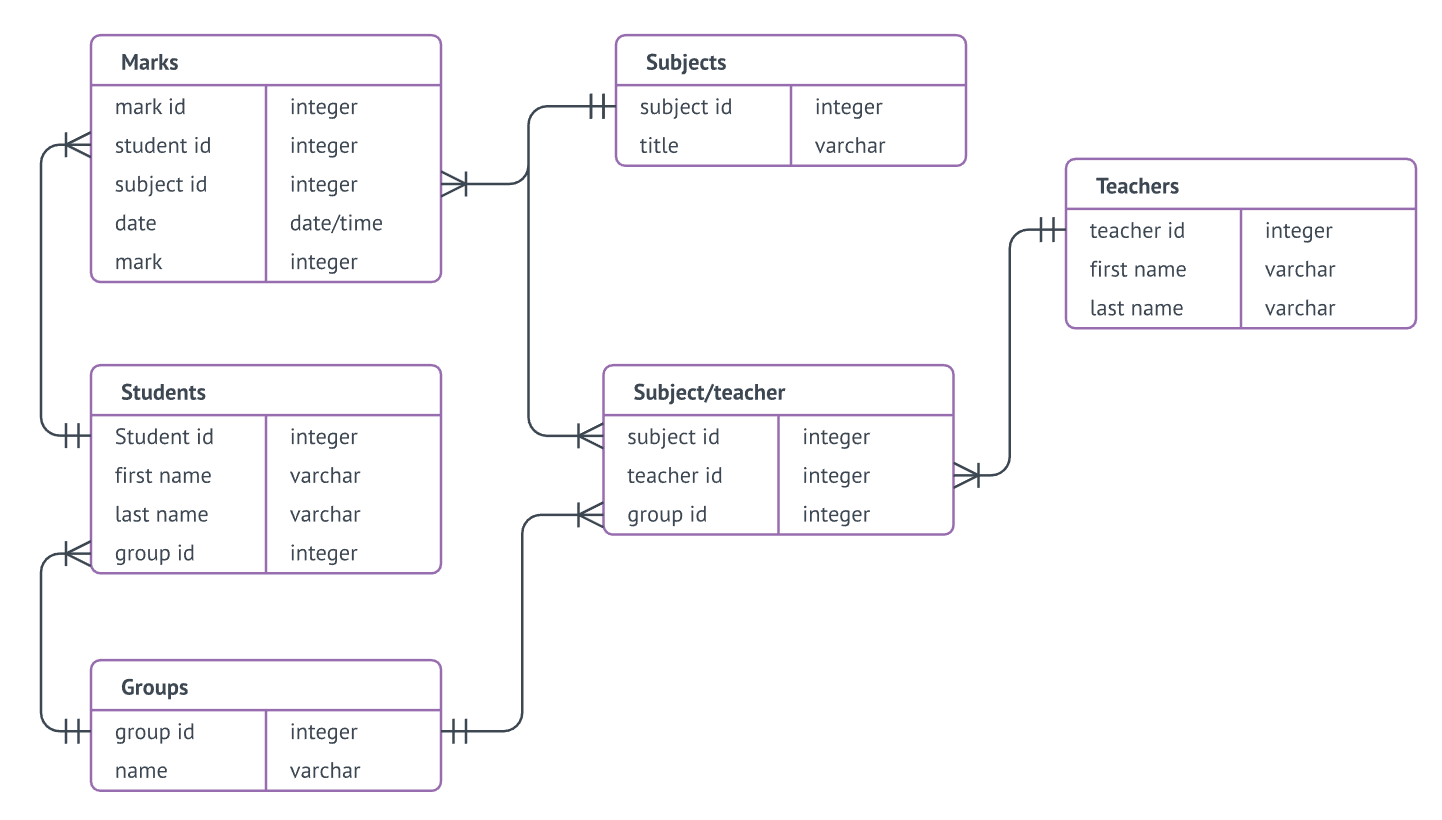


Рисунок 4 – Пример ER-диаграммы

В соответствии с рисунком 5 представлен пример User Story Mapping диаграммы:

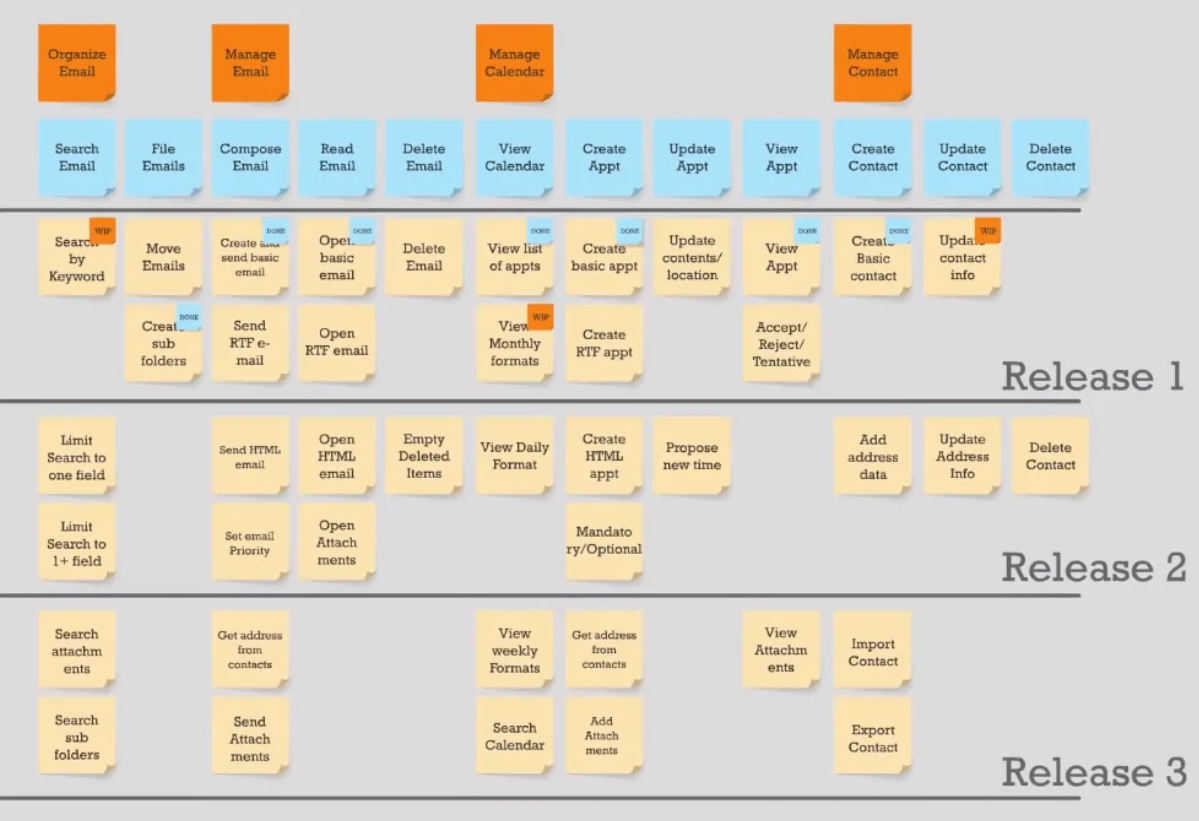


Рисунок 5 – Пример User Story Mapping диаграммы

**Реализуйте физическую модель базы данных**. Количество сущностей должно быть на ваше усмотрение, но не менее 10. Соблюдайте общие требования к реализации базы данных. В базе данных должна храниться информация о студентах, группах, сотрудниках, преподавателях и заявках на поступление.

Реализуйте графический интерфейс пользователя. Необходимо реализовать следующие обязательные формы:

- форма авторизации;

- форма восстановления доступа к системе;

- форма с реализацией функциональной части по работе со студентами;

- форма с реализацией функциональной части по работе с группами;

- форма с реализацией функциональной части по работе с преподавателями;

- форма с реализацией функциональной части по работе с заявками.

Соблюдайте общие требования к формированию графического интерфейса. Интерфейс должен быть выполнен в едином стиле. Необходимо реализовать логотип и иконку у приложения. Язык интерфейса – Русский.

**Описание функциональной части**

**Авторизация.**

При запуске настольного приложения перед потенциальным пользователем открывается окно авторизации. Для успешной авторизации пользователю необходимо ввести логин/почту и пароль для существующей учетной записи. Если пользователь ввёл логин и пароль, то происходит авторизацию и открытие формы в зависимости от типа пользователя: сотрудник учебной части или администратор. В случае, если авторизация неуспешна по ряду причин: не заполнены поля для ввода, учетной записи не существует, то в таком случае выводится сообщение об ошибке. Если пользователь 5 раз неверно ввёл логин или пароль, то кнопка авторизации должна быть заблокирована на 10 секунд, при этом остальные элементы управления должны быть доступны.

Если пользователь забыл свой пароль, то в таком случае он может перейти по гиперссылке на форму с восстановлением пароля по электронной почте. Пользователь указывает почту, если она проходит валидацию на правильность и существует, то в таком случае на почтовый адрес, указанный пользователем отправляется сообщение с временным паролем (он должен быть действителен только в рамках текущей сессии). В случае ввода некорректного адреса электронной почты, необходимо оповестить пользователя об ошибке, которую он допустил.

**Терминал сотрудника учебной части.**

**Форма со студентами.**

На форме со студентами выводится информация об студентах в виде таблицы. Необходимо реализовать фильтрацию с помощью выпадающих списков (ComboBox) по группе, статусу обучения (обучается или окончил), дате рождения, по форме обучения (Коммерция, Бюджет и Целевое).

Необходимо реализовать поиск по совпадению по верхнему и нижнему регистрам по ФИО студента.

Необходимо реализовать удаление студента по кнопке, находящейся с ним в одной строке и имеющей изображение. Перед удалением система должна уточнить у сотрудника, уверен ли он в своём решении. После данные в таблице должны автоматически обновиться.

Необходимо реализовать форму с добавлением нового студента в таблицу. Сотрудник вводит следующую информацию:

- Фамилия;

- Имя;

- Отчество;

- Дата рождения (DatePicker);

- Группа (выпадающий список);

- Статус студента (выпадающий список);

- Форма обучения (выпадающий список);

- Дата зачисления (выпадающий список);

- Дата выбытия (выпадающий список);

- Номер телефона.

Затем необходимо выбрать фотографию студента из проводника, после добавления студента, изображения с именем (ФИО\_студента.png) хранятся в отдельной директории. Если сотрудник не выбрал фотографию, то на её место ставится заглушка по аналогичному алгоритму. Перед добавлением студента система должна уточнить у сотрудника, действительно ли он хочет добавить студента в базу данных. После добавления все текстовые поля на форме должны очиститься, а таблица со студентами автоматически обновиться. В случае, если студент некорректно ввёл данные, то должна произойти обработка исключений и оповещение сотрудника об его ошибке. Также на форме должна быть предусмотрена кнопка очистки текстовых полей и сброса дат.

Необходимо реализовать редактирование студента. При двойном нажатии по строке в таблице с данными о студенте должна открываться форма редактирования. При редактировании данные переносятся в форму редактирования, которая аналогична форме добавления, только в ней уже заполнены все данные, выбраны все элементы в выпадающих списка и даты в DatePicker, а также фотографию студента. После ввода новой информации и прохождения валидации данных, система должна уведомить сотрудника, действительно ли он хочет обновить данные. После обновления, форма должна быть закрыта, а данные в таблице автоматически обновлены.

Необходимо реализовать вывод справки об очном обучении в текстовом документе (Документ формируется в формате .doc / .docx или в их аналогах) по шаблону с вашего учебного учреждения выбранному студенту. Все документы формируются в отдельной директории по ФИО студента и текущей дате. Кнопка для формирования документа находится в одной строке с данными студента.

**Форма с группами.**

Форма с группами выглядит аналогичным образом, как и форма со студентами, только данные должны быть представлены в виде списка (ListView).

Необходимо реализовать форму добавления новой группы, в которой сотрудник в текстовых полях указывает следующую информацию:

- Специальность (Выпадающий список);

- Количество студентов в группе;

- Количество свободных мест в группе;

- Префикс группы (А, Б, В, Г) (Выпадающий список);

- Курс (Выпадающий список);

- Дата формирования (DatePicker);

- Дата окончания обучения (DatePicker);

- Классный руководитель (Выпадающий список педагогов).

Название группы должно формироваться автоматически из курса, префикса, даты формирования, даты окончания обучения.

На форме необходимо реализовать возможность каскадного удаления групп, а также редактирования информации о выбранной группе.

Также необходимо реализовать просмотр списка обучающихся в выбранной группе.

Все вышеописанные функциональные особенности необходимо реализовать с учетом валидации данных и обратной связи с пользователем.

**Форма с педагогами.**

На данной форме отображается список педагогов с фотографиями в виде списка (ListView) с фотографией педагога.

Необходимо реализовать добавление педагогов со следующей информацией:

- Фамилия;

- Имя;

- Отчество;

- Дата рождения;

- Электронная почта;

- Номер телефона;

- Педагогический стаж.

Также у каждого педагога должна быть личная фотография и все функциональные реализации с фотографией по аналогии со студентами.

Необходимо реализовать редактирование и удаление по всем вышеописанным требованиям.

**Форма с заявками.**

Заготовьте 10 заявок абитуриентов для тестирования функциональной части приложения. Предположим, что все заявки формируются из веб-формы на сайте вашего учебного учреждения, после уже попадают в базу данных и отображаются в настольном приложении. В каждой заявке указывается следующая обязательная информация:

- Фамилия;

- Имя;

- Отчество;

- Дата рождения;

- Место обучения;

- Аттестат (в виде файла);

- Номер телефона;

- Электронная почта;

- Статус (по умолчанию на рассмотрении).

Сотрудник должен проверить корректность заполненных данных и документов. После этого либо одобрить заявку, либо отклонить её. В случае одобрения, открывается форма с добавлением абитуриента в выбранную группу. После все вышеописанных действий статус заявки меняется на «одобрена» или «отклонена». Если заявка была одобрена, то в таком случае в таблице со студентами появляется новая запись. Необходимо реализовать фильтрацию заявок по их статусам, а также поиск нужной заявки по ФИО.

Все одобренные заявки выделяются зеленым цветом, а не одобренные красным.

Необходимо соблюсти валидацию данных, а также обработку исключительных ситуаций и реализовать обратную связь с пользователем. По аналогии с предыдущими формами, заявки можно редактировать и удалять.

**Форма с личным кабинетом сотрудника.**

На данной форме отображается вся информация о сотруднике, включая логин и пароль (PasswordBox). Необходимо реализовать возможность изменения информации. После изменения информации сессия перезапускается. Для отображения пароля реализуйте кнопку с изображением глаза. Сотрудник не может изменить свой тип (этим занимается администратор).

**Терминал администратора.**

**Форма с учетом сотрудников**.

Необходимо реализовать форму, в которой в виде таблицы отображаются все сотрудники. Реализуйте возможность добавления, редактирования и удаления сотрудников по аналогии с другими формами. Не забывайте соблюдать валидацию данных, обработку исключений и обратную связь с пользователем.

У администратора также есть личный кабинет.

**Модульное тестирование.**

Реализуйте следующие классы по работе с данными из базы данных:

- Класс добавления данных;

- Класс редактирования данных;

- Класс удаления данных;

- Класс вывода данных.

В каждом классе реализуйте несколько методов по работе с данными.

На каждый метод реализуйте 2 unit-теста (по положительному сценарию и отрицательному).

Все unit-тесты должны быть успешно пройдены. Итоговое кол-во unit-тестов должно составлять 16 штук.

**Пользовательская документация.**

Составьте пользовательскую документацию по работе с разработанной систем в соответствии с предоставленным примером (Образец пользовательской документации).

Для документа необходимо сформировать титульный лист, оглавление и автоматическую нумерацию страниц. Пользовательская документация должна быть оформлена без орфографических ошибок и опечаток. При описании возможностей необходимо использовать ссылки на рисунки с разработанного приложения.

**Руководство по установке и инсталлятор приложения.**

Сформируйте инсталлятор для приложения с помощью расширения **Visual Studio Installer Projects**. После реализации инсталлятора протестируйте его работу и сформируйте руководство по установке со всеми требованиями по аналогии с предыдущим пунктом. Пример руководства по установке представлен в соответствии с предоставленным примером (Образец руководства по установке).

**Общие требования и рекомендации.**

Для приложения необходимо реализовать минимум 10 примеров тестовых данных в базе данных для каждой сущности. Избегайте орфографических ошибок и опечаток. Графический интерфейс должен быть выполнен в едином стиле. Соблюдайте стилистику кода (CamelCase для C#). Используйте комментарии в программном коде.