Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерного проектирования

Кафедра проектирования информационно-компьютерных систем

Современные технологии проектирования информационных систем

Отчет

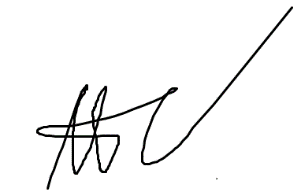
по лабораторной работе №5

на тему:

**ОРГАНИЗАЦИЯ ХРАНЕНИЯ ДАННЫХ. ОПИСАНИЕ СТАТИЧЕСКИХ И ДИНАМИЧЕСКИХ АСПЕКТОВ ПОВЕДЕНИЯ ПРОГРАММНЫХ ОБЪЕКТОВ**

Проверил \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ А. В. Шелест

(подпись)



Выполнил 11.11.2023 \_\_\_\_\_\_\_\_\_\_\_\_\_\_ А. И. Ткачук

(подпись) 014301

Минск, 2023

1 ОПИСАНИЕ СТРУКТУРЫ ДАННЫХ

Для описания логической модели данных используется диаграмма «Сущность-связь», представляющая логические группы информации предметной области и их взаимосвязи. Диаграмма представлена на рисунке 1. Можно выделить следующие сущности: Заявитель, Диспетчер, Заявление, Службы реагирования на ЧС, Тип ЧС, Вид ЧС, Характер ЧС.

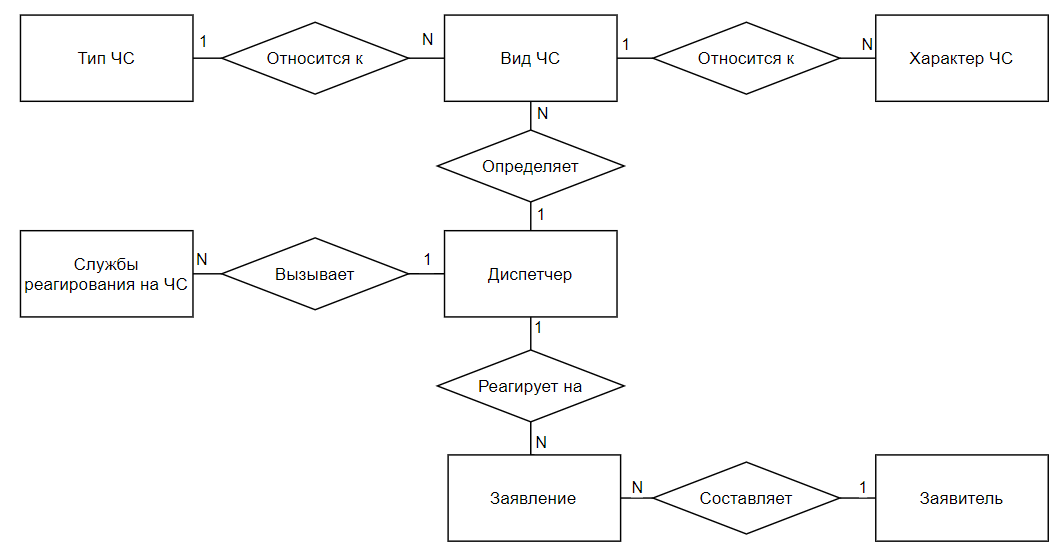


Рисунок 1.1 – Логическая модель данных

**Словарь данных.** Словарь данных в таблице 1.1 представляет собой набор подробной информации о часто используемых в приложении сущностях данных, вытекающих из особенностей предметной области программного средства.

Таблица 1.1 – Словарь данных программного средства

| **Элемент данных** | **Описание** | | **Тип данных и длина** | | **Значение** | | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Заявитель** | | | | | | | |
| идентификатор заявителя | Уникальный идентификатор пользователя, зарегистрировавшегося в мобильном приложении | | Автоинкрементный числовой тип | | Уникальное значение, первичный ключ | | |
| имя | Имя заявителя | | Строковое значение,  до 256 символов | | Не может быть пустым | | |
| фамилия | | Фамилия заявителя | | Строковое значение,  до 256 символов | | Не может быть пустым | |
| отчество | | Отчество заявителя, если имеется | | Строковое значение,  до 256 символов | |  | |
| домашний адрес | | Домашний адрес заявителя | | Строковое значение,  до 256 символов | | Не может быть пустым | |
| рабочий адреса | | Адрес работы заявителя, если имеется | | Строковое значение,  до 256 символов | |  | |
| email | | Почта заявителя | | Строковое значение,  до 256 символов | | Не может быть пустым, уникальное значение | |
| пароль | | Пароль заявителя в захешированном виде | | Строковое значение,  до 256 символов | | Не может быть пустым | |
| **Заявление** | | | | | | | |
| идентификатор заявления | | Уникальный идентификатор заявления, сформированного заявителем | | Автоинкрементный числовой тип | | Уникальное значение, первичный ключ | |
| тип ЧС | | Название типа ЧС, указанное заявителем | | Строковое значение,  до 256 символов | | Не может быть пустым | |
| дополнительная информация | | Дополнительная информация о ЧС, предоставленная заявителем | | Строковое значение,  до 256 символов | |  | |
| местоположение ЧС | | Адрес или описание места расположения ЧС | | Строковое значение,  до 256 символов | | Не может быть пустым | |
| время происшествия | | Время происшествия, указанное заявителем | | Дата и время | | Не может быть пустым | |
| наличие пострадавших | | Есть пострадавшие на месте происшествия или нет | | Логический тип | | true, false | |
| количество пострадавших | | Количество пострадавших, если они есть | | Строковое значение,  до 256 символов | |  | |
| статус заявителя | | Находится заявитель в опасности или нет | | Логический тип | | true, false | |
| статус заявления | | Произведены действия по реагированию на заявление или нет | | Логический тип | | true, false | |
| время получения заявления | Время получения заявления сервером | | | Дата и время | | | Не может быть пустым |
| время завершения действий | Время, определенное системой после подтверждения диспетчером окончания действий по реагированию на ЧС | | Дата и время | | Не может быть пустым | | |
| **Диспетчер** | | | | | | | |
| идентификатор диспетчера | Уникальный идентификатор пользователя десктоп-приложения | | Автоинкрементный числовой тип | | Уникальное значение, первичный ключ | | |
| логин | Логин диспетчера | | Строковое значение,  до 256 символов | | Не может быть пустым, уникальное значение | | |
| пароль | Пароль диспетчера | | Строковое значение,  до 256 символов | | Не может быть пустым | | |
| статус | Обладает пользователь правами администратора базы данных системы или нет | | Логический тип | | true, false | | |
| **Службы реагирования на ЧС** | | | | | | | |
| идентификатор службы реагирования | Уникальный идентификатор службы реагирования на ЧС | | Автоинкрементный числовой тип | | Уникальное значение, первичный ключ | | |
| название службы | Название службы реагирования на ЧС | | Строковое значение,  до 256 символов | | Не может быть пустым, уникальное значение | | |
| **Тип ЧС** | | | | | | | |
| идентификатор типа ЧС | Уникальное значение типа ЧС | | Автоинкрементный числовой тип | | Уникальное значение, первичный ключ | | |
| название | Название типа ЧС | | Строковое значение,  до 256 символов | | Не может быть пустым | | |
| рекомендации заявителю | Рекомендация, предоставляемая заявителю по действиям, которые он должен предпринять при данном типе ЧС | | Строковое значение,  до 256 символов | | Не может быть пустым, уникальное значение | | |
| **Вид ЧС** |  | |  | |  | | |
| идентификатор вида ЧС | Уникальное значение вида ЧС | | Автоинкрементный числовой тип | | Уникальное значение, первичный ключ | | |
| название | Название вида ЧС | | Строковое значение,  до 256 символов | | Не может быть пустым, уникальное значение | | |
| **Характер ЧС** |  | |  | |  | | |
| идентификатор характера ЧС | Уникальное значение характера ЧС | | Автоинкрементный числовой тип | | Уникальное значение, первичный ключ | | |
| название | Название характера ЧС | | Строковое значение,  до 256 символов | | Не может быть пустым, уникальное значение | | |

На этапе логического проектирования у сущностей выделяются атрибуты, описывающие необходимые в границах проекта свойства объектов. На рисунке 1.2 представлена логическая модель сущности «Заявитель» с атрибутами, которые к ней относятся. Так как заявитель является пользователем разрабатываемого программного средства, у него должны быть пароль и идентификатор заявителя, который и является ключевым атрибутом.

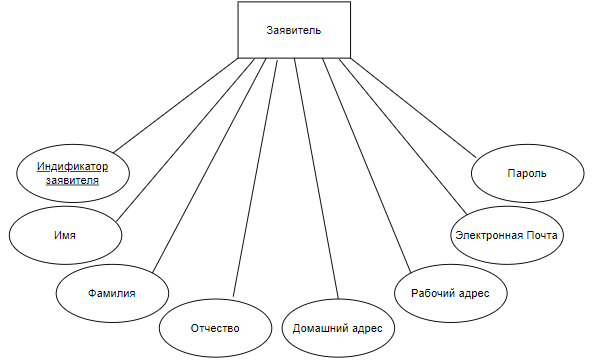


Рисунок 1.2 – Логическая модель сущности «Заявитель»

Логическая модель сущности «Заявление» представлена на рисунке 1.3. Она должна содержать указатель на заявителя, составившего заявление. В данном случаи – это электронная почта заявителя. Ключевым атрибутом заявления является идентификатор заявления.



Рисунок 1.3 – Логическая модель сущности «Заявление»

Было принято решение изменить сущность «диспетчер» на «Пользователя системы», так как программное средство предполагает присутствие в нем администратора базы данных. Администратор должен заполнять таблицы, относящиеся к категориям ЧС и службам реагирования на ЧС. Профиль пользователя должен иметь логин и пароль для входа в систему. Ключевым атрибутом является идентификатор пользователя. Логическая модель данной сущности представлена на рисунке 1.4.

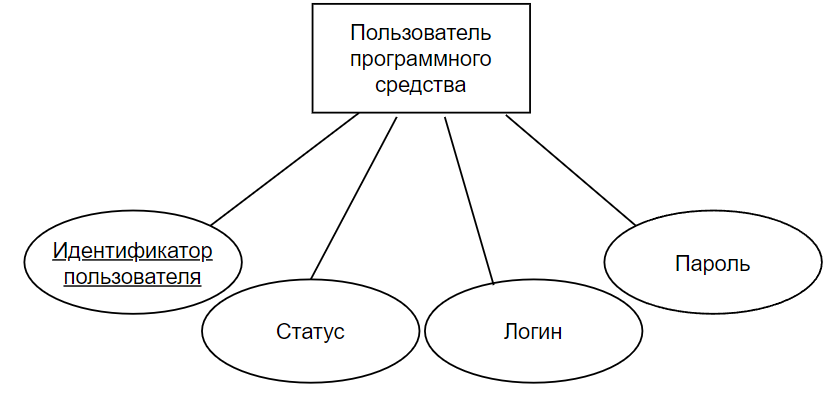


Рисунок 1.4 – Логическая модель сущности «Пользователь программного средства»

Логическая модель сущности «Характер ЧС» представлена на рисунке 1.5. Ключевой атрибут сущности – идентификатор Характера ЧС.

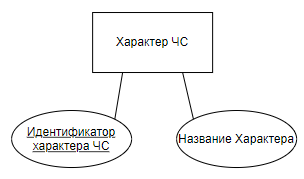


Рисунок 1.5 – Логическая модель сущности «Характер ЧС»

Логическая модель сущности «Вид ЧС» представлена на рисунке 1.6. Ключевым атрибутом сущности является идентификатор вида ЧС. Вид относится к ЧС определенного характера. Данная связь реализована через идентификатор характера ЧС.

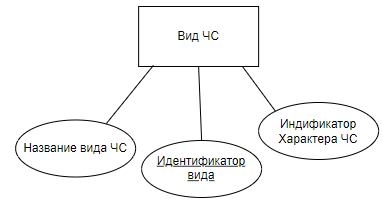


Рисунок 1.6 – Логическая модель сущности «Вид ЧС»

Логическая модель сущности «Тип ЧС» представлена на рисунке 1.7. Ключевой атрибут сущности – идентификатор типа. Связь с сущностью «Вид ЧС» реализована через идентификатор вида ЧС.

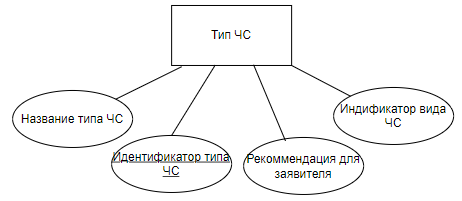


Рисунок 1.7 – Логическая модель сущности «Тип ЧС»

Логическая модель сущности «Службы реагирования на ЧС» представлена на рисунке 1.8. Ключевой атрибут сущности – идентификатор службы.

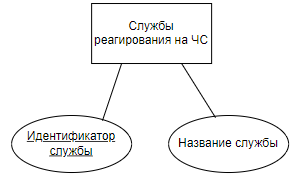


Рисунок 1.8 – Логическая модель сущности «Службы реагирования на ЧС»

На рисунке 1.9 представлена общая логическая модель базы данных.

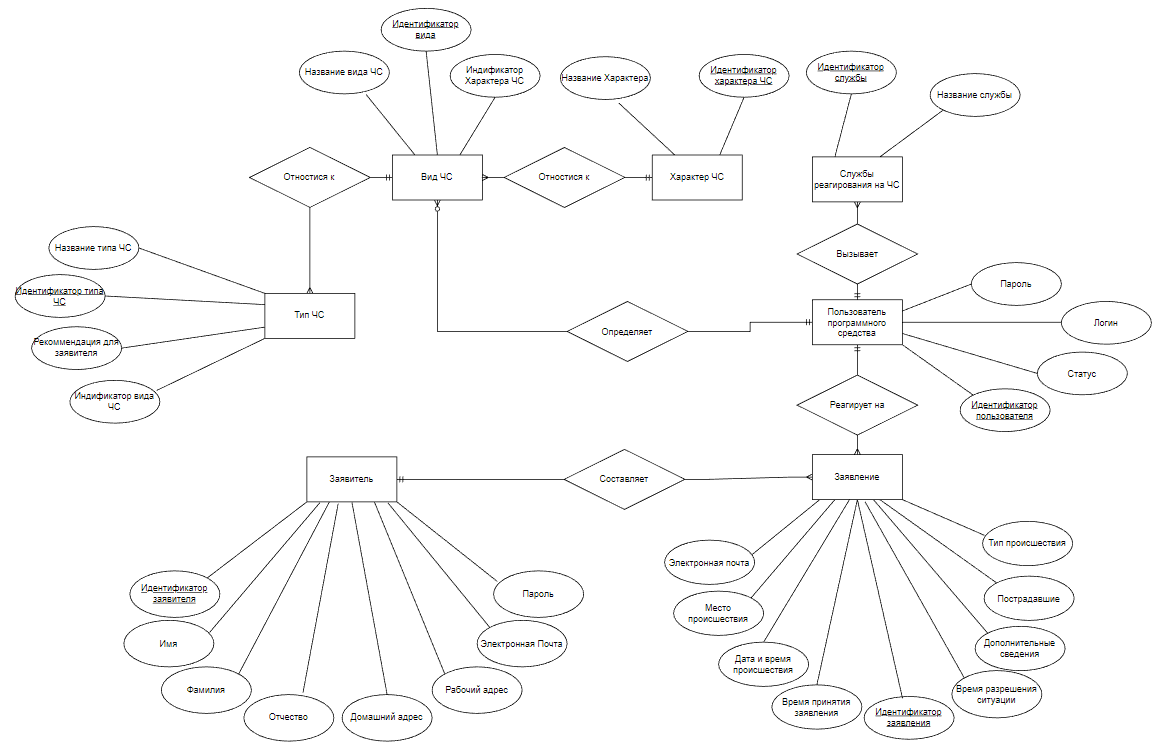


Рисунок 1.9 – Логическая модель базы данных

После анализа логической модели БД, был сделан вывод о необходимости добавления сущности «Выбор пользователя», а также сущности «Связь вида ЧС и служб реагирования». Они помогут обеспечить минимальную избыточность и физический объём базы данных.

На рисунке 1.10 представлена физическая модель базы данных, приведенная к третьей нормальной форме.

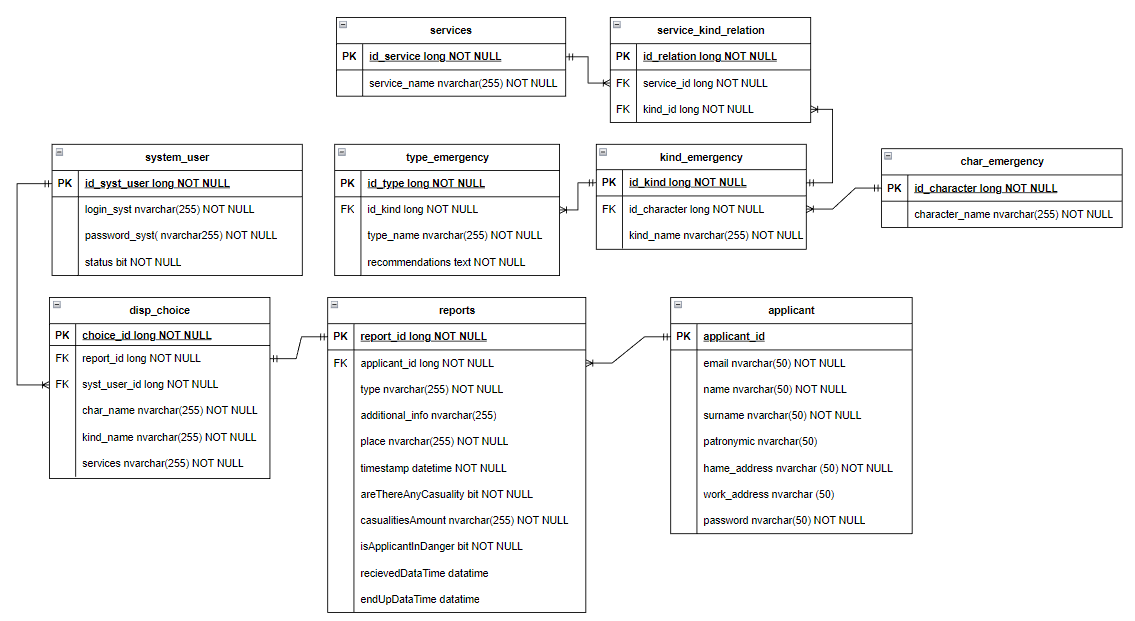


Рисунок 1.10 – Физическая модель базы данных

Рассмотрим подробнее полученные таблицы и их атрибуты в таблице 1.2.

Таблица 1.2 – Описание сущностей базы данных

| **Наименование поля** | | | | **Назначение атрибута** | | | | | | **Тип данных** | | | | **Примечание** | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *applicant* (заявитель) | | | | | | | | | | | | | | | |
| *applicant\_id* | | | | Уникальный идентификатор заявителя | | | | | | целое автоинкрементное по схеме СУБД число типа *BIGINT* | | | | первичный ключ | |
| *name* | | | | Имя заявителя | | | | | | строковое значение типа *VARCHAR* с максимальным количеством символов 255 | | | |  | |
| *surname* | | | | Фамилия заявителя | | | | | | строковое значение типа *VARCHAR* с максимальным количеством символов 255 | | | |  | |
| *patronymic* | | | | Отчество заявителя, если имеется | | | | | | строковое значение типа *VARCHAR* с максимальным количеством символов 255 | | | |  | |
| *home\_address* | | | | Домашний адрес заявителя | | | | | | строковое значение типа *VARCHAR* с максимальным количеством символов 255 | | | |  | |
| *work\_address* | | | | Адрес работы заявителя, если имеется | | | | | | строковое значение типа *VARCHAR* с максимальным количеством символов 255 | | | |  | |
| *email* | | | | Почта заявителя | | | | | | строковое значение типа *VARCHAR* с максимальным количеством символов 255 | | | |  | |
| *password* | | | | Пароль заявителя в захешированном виде | | | | | | строковое значение типа *VARCHAR* с максимальным количеством символов 255 | | | |  | |
| *Reports* (заявления) | | | | | | | | | | | | | | | |
| *id\_report* | | | | Уникальный идентификатор заявления | | | | | | целое автоинкрементное по схеме СУБД число типа *BIGINT* | | | | первичный ключ | |
| *type* | | | | Название типа ЧС, указанное заявителем | | | | | | строковое значение типа *VARCHAR* с максимальным количеством символов 255 | | | |  | |
| *additional\_info* | | | | Дополнительная информация о ЧС, предоставленная заявителем | | | | | | Тип данных TEXT | | | |  | |
| *place* | | | | Адрес или описание места расположения ЧС | | | | | | строковое значение типа *VARCHAR* с максимальным количеством символов 255 | | | |  | |
| *timestamp* | | | | Время происшествия, указанное заявителем | | | | | | Значение типа  *TIMESTAMP* | | | |  | |
| *areThereAnyCasualties* | | | | Есть пострадавшие на месте происшествия или нет | | | | | логический тип  *BOOLEAN*,  принимает значения  *true* или *false* | | | | Если на месте происшествия есть пострадавшие, то значение *true*, по умолчанию *false* | | |
| *casualtiesAmount* | | | | Количество пострадавших, если они есть | | | | | строковое значение типа *VARCHAR* с максимальным количеством символов 255 | | | |  | | |
| *isUserInDanger* | | | | Находится заявитель в опасности или нет | | | | | логический тип  *BOOLEAN*,  принимает значения  *true* или *false* | | | | Если заявитель в опасности, то значение *true*, по умолчанию *false* | | |
| *wasSeen* | | | | Произведены действия по реагированию на заявление или нет | | | | | логический тип  *BOOLEAN*,  принимает значения  *true* или *false* | | | | Если на заявление произведена реакция диспетчера, то значение *true*, по умолчанию *false* | | |
| *recieved\_date\_time* | | | | Время получения заявления сервером | | | | | Значение типа  *TIMESTAMP* | | | |  | | |
| *applicant\_id* | | | | Идентификатор заявителя | | | | | Целое число типа  *BIGINT*, соответствует значению в  таблице *applicant* | | | | внешний ключ,  связывает с  таблицей *applicant*,  кардинальность  связи – M:1 | | |
| *end\_up\_datatime* | | | | Время, определенное системой после подтверждения диспетчером окончания действий по реагированию на ЧС | | | | | Значение типа  *TIMESTAMP* | | | |  | | |
| *syst\_user* (пользователь системы) | | | | | | | | | | | | | | | |
| *id\_syst\_user* | Уникальный идентификатор пользователя десктоп-приложения | | | | целое автоинкрементное по схеме СУБД число типа *BIGINT* | | | | | | первичный ключ | | | | |
| *login\_syst* | Логин диспетчера | | | | строковое значение типа *VARCHAR* с максимальным количеством символов 255 | | | | | |  | | | | |
| *Password\_syst* | Пароль диспетчера | | | | строковое значение типа *VARCHAR* с максимальным количеством символов 255 | | | | | |  | | | | |
| *status* | Обладает пользователь правами администратора базы данных системы или нет | | | | логический тип  *BOOLEAN*,  принимает значения  *true* или *false* | | | | | | Если пользователь является администратором БД, то значение *true*, по умолчанию *false* | | | | |
| *service*(службы реагирования на ЧС) | | | | | | | | | | | | | | | |
| *id\_service* | Уникальный идентификатор службы реагирования на ЧС | | | | целое автоинкрементное по схеме СУБД число типа *BIGINT* | | | | | | первичный ключ | | | | |
| *service\_name* | Название службы реагирования на ЧС | | | | строковое значение типа *VARCHAR* с максимальным количеством символов 255 | | | | | |  | | | | |
| *type\_em* (тип ЧС) | | | | | | | | | | | | | | | |
| *id\_type* | Уникальное значение типа ЧС | | | | целое автоинкрементное по схеме СУБД число типа *BIGINT* | | | | | | первичный ключ | | | | |
| *name* | Название типа ЧС | | | | строковое значение типа *VARCHAR* с максимальным количеством символов 255 | | | | | |  | | | | |
| *recommendation* | Рекомендация, предоставляемая заявителю по действиям, которые он должен предпринять при данном типе ЧС | | | | Тип данных TEXT | | | | | |  | | | | |
| *id\_kind* | | Идентификатор вида ЧС, к которому относится тип | | | | | | Целое число типа  *BIGINT*, соответствует значению в  таблице *kind\_em* | | | | | внешний ключ,  связывает с  таблицей *kind\_em*,  кардинальность  связи – M:1 | | |
| *kind\_em* (вид ЧС) | | | | | | | | | | | | | | | |
| *id\_kind* | | Уникальный идентификатор вида ЧС | | | | | | целое автоинкрементное по схеме СУБД число типа *BIGINT* | | | | | первичный ключ | | |
| *kind\_name* | | Название вида ЧС | | | | | | строковое значение типа *VARCHAR* с максимальным количеством символов 255 | | | | |  | | |
| *id\_char* | | Идентификатор характера ЧС, к которому относится вид | | | | | | Целое число типа  *BIGINT*, соответствует значению в  таблице *char\_em* | | | | | внешний ключ,  связывает с  таблицей *char\_em*,  кардинальность  связи – M:1 | | |
| *char\_em* (характер ЧС) | | | | | | | | | | | | | | | |
| *id\_char* | | Уникальное значение характера ЧС | | | | | | целое автоинкрементное по схеме СУБД число типа *BIGINT* | | | | | первичный ключ | | |
| *char\_name* | | Название характера ЧС | | | | | | строковое значение типа *VARCHAR* с максимальным количеством символов 255 | | | | |  | | |
| *service\_kind\_relation* (связь служб с видами ЧС) | | | | | | | | | | | | | | | |
| *id\_relation* | | Уникальный идентификатор связи между услугами и видами чрезвычайных ситуаций | | | | | | целое автоинкрементное по схеме СУБД число типа *BIGINT* | | | | | первичный ключ | | |
| *service\_id* | | Идентификатор службы реагирования на ЧС | | | | Целое число типа  *BIGINT*, соответствует значению в  таблице *service* | | | | | | внешний ключ,  связывает с  таблицей *service*,  кардинальность  связи – M:1 | | |
| *kind\_id* | | Идентификатор вида ЧС | | | | Целое число типа  *BIGINT*, соответствует значению в  таблице *kind\_em* | | | | | | внешний ключ,  связывает с  таблицей *kind\_em*,  кардинальность  связи – M:1 | | |
| *disp\_choice* (выбор диспетчера) | | | | | | | | | | | | | | |
| *сhoice\_id* | | Уникальный идентификатор выбора пользователя для определенного заявления | | | | целое автоинкрементное по схеме СУБД число типа *BIGINT* | | | | | | первичный ключ | | |
| *name\_char* | | Название характера ЧС, определенного диспетчером | | | | строковое значение типа *VARCHAR* с максимальным количеством символов 255 | | | | | |  | | |
| *name\_kind* | | Названия вида ЧС, определенного диспетчером | | | | строковое значение типа *VARCHAR* с максимальным количеством символов 255 | | | | | |  | | |
| *syst\_user\_id* | | Идентификатор пользователя системы | | | | Целое число типа  *BIGINT*, соответствует значению в  таблице *syst\_user* | | | | | | внешний ключ,  связывает с  таблицей *syst\_user*,  кардинальность  связи – M:1 | | |
| *repot\_id* | | | Идентификатор заявления | | | | Целое число типа  *BIGINT*, соответствует значению в  таблице *report\_id* | | | | | внешний ключ,  связывает с  таблицей *reports*,  кардинальность  связи – 1:1 | | |
| *services* | | | Службы, выбранные диспетчером. | | | | строковое значение типа *VARCHAR* с максимальным количеством символов 255 | | | | |  | | |

Ниже представлен скрипт генерации базы данных.

-- Создание таблицы Характеров ЧС

create table char\_em

(

char\_id int auto\_increment

primary key,

char\_name varchar(255) not null

);

-- Создание таблицы видов ЧС

create table kind\_em

(

kind\_id int auto\_increment

primary key,

kind\_name varchar(255) not null,

id\_char int null,

constraint kind\_em\_ibfk\_1

foreign key (id\_char) references char\_em (char\_id)

);

create index id\_char

on kind\_em (id\_char);

-- Создание таблицы служб реагирования на ЧС

create table service

(

service\_id int auto\_increment

primary key,

service\_name varchar(255) not null

);

-- Создание таблицы связи видов и служб

create table service\_kind\_relation

(

id\_relation int auto\_increment

primary key,

service\_id int null,

kind\_id int null,

constraint service\_kind\_relation\_ibfk\_1

foreign key (service\_id) references service (service\_id),

constraint service\_kind\_relation\_ibfk\_2

foreign key (kind\_id) references kind\_em (kind\_id)

);

create index kind\_id

on service\_kind\_relation (kind\_id);

create index service\_id

on service\_kind\_relation (service\_id);

-- Создание таблицы пользователей системы

create table syst\_user

(

id\_syst int auto\_increment

primary key,

login\_syst varchar(255) charset utf8mb3 not null,

password varchar(255) charset utf8mb3 not null,

status\_syst tinyint(1) not null

);

-- Создание таблицы типов ЧС

create table type\_em

(

id\_type int auto\_increment

primary key,

name varchar(255) null,

recommendations text null,

id\_kind int null,

constraint type\_em\_pk

unique (name),

constraint type\_em\_ibfk\_1

foreign key (id\_kind) references kind\_em (kind\_id)

);

create index id\_kind

on type\_em (id\_kind);

-- Создание таблицы заявителей

create table applicants

(

applicant\_id int auto\_increment

primary key,

name varchar(255) null,

surname varchar(255) null,

patronymic varchar(255) null,

home\_address varchar(255) null,

work\_address varchar(255) null,

email varchar(255) not null,

password varchar(255) not null,

constraint user\_data\_pk

unique (email)

);

-- Создание таблицы заявлений

create table reports

(

id\_report int auto\_increment

primary key,

type varchar(255) null,

additional\_info text null,

place varchar(255) null,

timestamp timestamp null,

areThereAnyCasualties tinyint(1) null,

casualtiesAmount varchar(255) null,

isUserInDanger tinyint(1) null,

wasSeen tinyint(1) null,

user\_email varchar(255) not null,

recieved\_date\_time datetime null,

end\_up\_datatime datetime null,

constraint reports\_user\_data\_email\_fk

foreign key (user\_email) references user\_data (email)

);

-- Создание таблицы выбора диспетчера

create table disp\_choice

(

id int auto\_increment

primary key,

name\_char varchar(255) null,

name\_type varchar(255) null,

services varchar(255) null,

repot\_id int not null,

syst\_user\_id int not null,

constraint disp\_choice\_reports\_id\_report\_fk

foreign key (repot\_id) references reports (id\_report),

constraint disp\_choice\_syst\_user\_id\_fk

foreign key (syst\_user\_id) references syst\_user (id\_syst)

);

Эти таблицы представляют собой структуру базы данных «emergency» с разными аспектами информации о чрезвычайных ситуациях, пользователях и службах, связанных с реагированием на критические ситуации.

2 СПРОЕКТИРОВАТЬ И ОПИСАТЬ СТАТИЧЕСКИЕ АСПЕКТЫ ПОВЕДЕНИЯ ПРОГРАММНЫХ ОБЪЕКТОВ

Диаграмма развертывания представляет собой схематическое изображение аппаратных и программных компонентов системы, их взаимодействие и конфигурацию. Она обозначает физические местоположения каждого компонента, например, серверов, клиентских устройств, баз данных, а также способы их связи через сетевые соединения.

Диаграмма развертывания разрабатываемого программного средства представлена на рисунке 2.1.

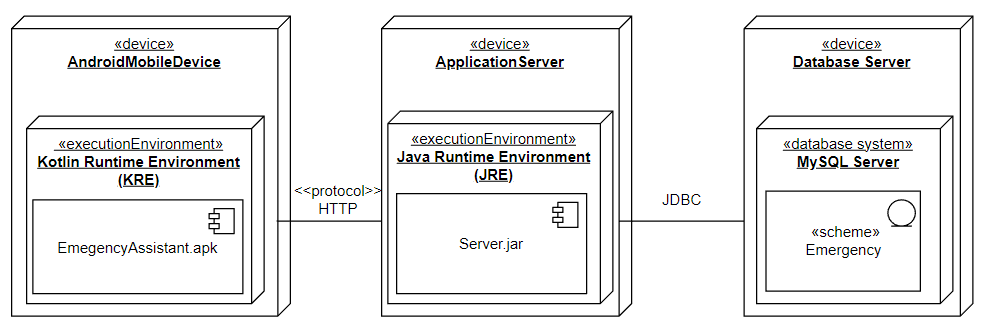


Рисунок 2.1 – Диаграмма развертывания

Эта диаграмма помогает визуализировать, как программное обеспечение развертывается на различных устройствах и серверах, и какие именно компоненты взаимодействуют для обеспечения работы системы.

Основной функциональностью серверной части разрабатываемого программного средства обладают контроллеры для FXML окон, обрабатывающие пользовательские события и управляющие потоком данных между пользовательским интерфейсом и логикой программы. Классы, связывающие данные с интерфейсом, играют ключевую роль в передаче информации между компонентами интерфейса и серверной частью программы, обеспечивая точную и своевременную передачу данных.

Дополнительно, для удобства хранения и доступа к изменяющимся значениям, используется шаблон проектирования «Синглтон». Он обеспечивает глобальный доступ к единственному экземпляру класса, хранящему переменные состояния, обеспечивая централизованное управление этими данными.

Классы, реализующие интерфейс HttpHandler из библиотеки HttpServer, ответственны за обработку запросов, поступающих на сервер. Они обрабатывают разнообразные виды запросов (GET, POST и другие), принимают данные от клиента, выполняют соответствующую обработку и формируют ответы, которые отправляются обратно клиенту. Диаграмма данных классов представлена на рисунке 2.2.

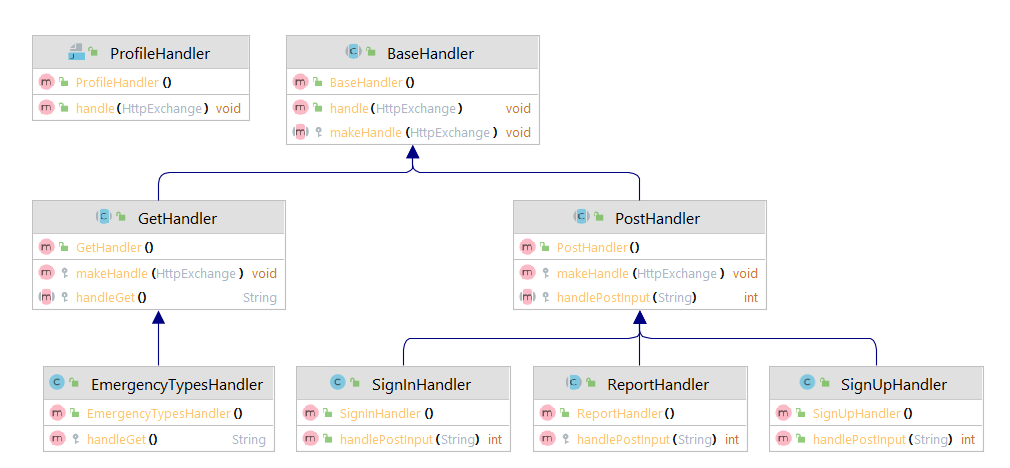


Рисунок 2.2 – Диаграмма классов-обработчиков запросов

Эти классы, реализующие интерфейс HttpHandler, обеспечивают стабильное взаимодействие между клиентской и серверной частями приложения. Они являются ключевым звеном для обработки входящих HTTP-запросов и обеспечивают правильное функционирование серверной части программы.

На рисунке 2.3 представлена диаграмма деятельности, на которой изображен процесс формирования заявления о ЧС заявителем.

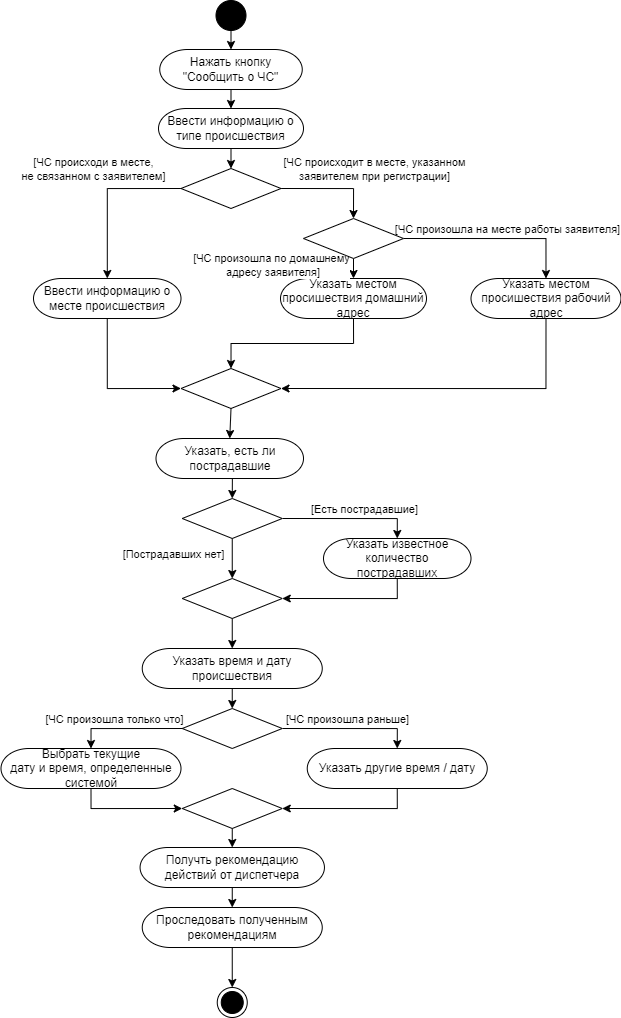


Рисунок 2.3 – Диаграмма деятельности процесса формирования заявления о ЧС заявителями

На данной диаграмме изображен процесс начала формирования заявления авторизованным заявителем, который начинает, нажав кнопку «Сообщить о ЧС». Далее, в открывшейся форме в приложении, он указывает тип произошедшей чрезвычайной ситуации, либо выбирает его из выпадающего списка.

Далее, заявителю необходимо выбрать место происшествия. В списке предложены его зарегистрированные домашний и рабочий адреса. Однако, если место происшествия не связано с этими адресами, заявитель может ввести адрес самостоятельно.

Также требуется указать информацию о пострадавших, если они есть. Заявитель сообщает о количестве пострадавших, о которых у него есть информация.

Система автоматически указывает текущую дату и время происшествия, однако заявитель может изменить их, если ситуация произошла ранее.

После ввода всех данных, заявитель получает рекомендации от диспетчера и следует за ними.

Далее рассмотрим диаграмму состояния объекта «заявление», представленную на рисунке 2.4.

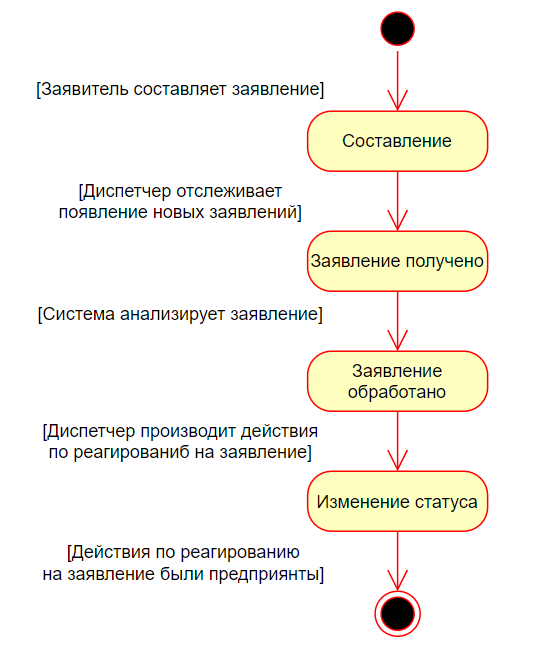


Рисунок 2.4 – Диаграмма состояний объекта «Заявление»

Диаграмма состояния – это визуальный инструмент моделирования, который позволяет представить различные состояния, в которых может находиться объект или система, и переходы между этими состояниями в ответ на определенные события.

Диаграмма состояний объекта «Заявление»:

1 Составление заявления. Начальное состояние, когда пользователь формирует заявление о чрезвычайной ситуации. Объект находится в состоянии «Составление».

2 Получение заявления диспетчером. После формирования заявления пользователем, оно передается диспетчеру. Объект переходит в состояние «Заявление получено».

3 Анализ системой. Система начинает анализировать данные заявления на основе имеющейся базы данных. Объект переходит в состоянии «Заявление обработано».

4 Рекомендации и реакция диспетчера. Диспетчер получает рекомендации от системы и предпринимает необходимые действия для реагирования на чрезвычайную ситуацию. После выполнения действий по реакции на заявление, диспетчер отмечает в системе факт завершения реакции. Объект переходит в состояние «Изменение статуса».

Таким образом, диаграмма состояний объекта «заявление о чрезвычайной ситуации» отражает последовательность состояний, через которые проходит объект в процессе своего жизненного цикла.

Диаграмма последовательности – это визуальное представление взаимодействия различных объектов или участников системы в определенной последовательности действий или обмене сообщениями. Она представляет собой отличный инструмент для моделирования, документирования и понимания взаимодействия между компонентами системы в определенном контексте.

Основная цель диаграммы последовательности - это описать порядок выполнения операций и передачу сообщений между объектами в системе или участниками. Она позволяет увидеть, как объекты взаимодействуют друг с другом в определенной ситуации или во времени, что позволяет прояснить логику работы системы или процесса.

На рисунке 2.5 представлена диаграмма последовательности процесса получения диспетчером рекомендаций по реагированию на чрезвычайные ситуации.

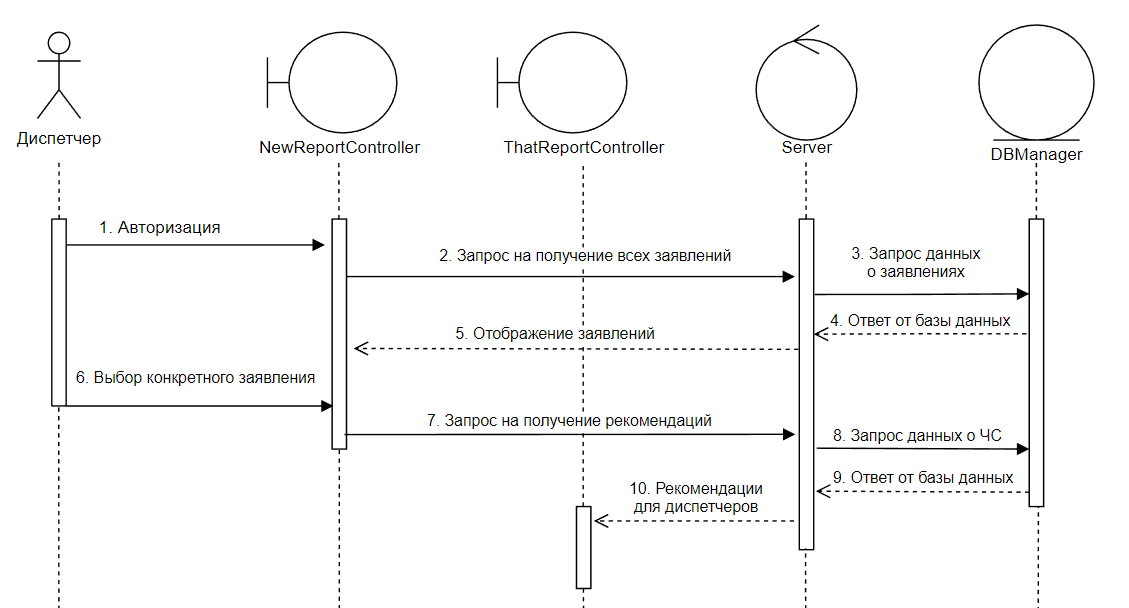


Рисунок 2.5 – Диаграмма последовательности процесса получения диспетчером рекомендаций по реагированию на ЧС

Диаграмма последовательности начинается с диспетчера, который использует десктоп-приложение для просмотра заявлений о чрезвычайных ситуациях. Диспетчер авторизуется, и система делает запрос к БД, чтобы получить данные о всех заявлениях. После, диспетчер выбирает конкретное заявление, по которому хочет получить рекомендации. Сервер делает запрос к базе данных, получает данные о ситуации и проводит анализ, определяя характер и вид чрезвычайной ситуации на основе информации из базы данных.

После этого, сервер формирует рекомендации для диспетчера, предоставляя ему список необходимых действий и служб для реагирования на данную ситуацию, которые отображаются на специальной форме. Получив рекомендации от сервера, диспетчер принимает решение о дальнейших шагах.

Таким образом происходит процесс получения диспетчером рекомендаций по реагированию на ЧС. Диаграмма последовательности в данном контексте помогает визуализировать и уточнить последовательность шагов, необходимых для получения рекомендаций диспетчером от программного средства. Она помогает лучше понять и документировать взаимодействие участников системы в процессе реагирования на подобные ситуации, обеспечивая более эффективное управление и координацию действий.

3 ВЫВОДЫ

В ходе выполнения лабораторной работы были проведены и описаны основные статические и динамические аспекты поведения программных объектов. Статические аспекты представлены на диаграмме классов, где проиллюстрирована структура программы. Также была разработана диаграмма развертывания, демонстрирующая физическое размещение компонентов системы.

Динамические аспекты моделировались через представление процессов в системе: диаграмма состояний объекта «заявление о ЧС» позволила визуализировать его жизненный цикл, а диаграмма деятельности процесса формирования заявления позволила увидеть последовательность шагов, выполняемых заявителями.

Особое внимание было уделено взаимодействию между пользователями и системой: диаграмма последовательности процесса получения диспетчером рекомендаций по реагированию на ЧС позволяет наглядно увидеть последовательность действий и передачу информации между участниками процесса.

Полученные диаграммы представляют важный этап проектирования системы, обеспечивая понимание ее структуры и взаимодействия между компонентами, что будет полезным при дальнейшей разработке и внедрении программного решения.