Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет
имени Н. Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н. Э. Баумана)

ФАКУЛЬТЕТ «Информатика и системы управления»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

К КУРСОВОЙ РАБОТЕ

НА ТЕМУ:

«Разработка базы данных для IoT-платформы умный дом»

Студент ИУ7-66Б (Группа) _____ (Подпись, дата) Мамврийский И. С. (И. О. Фамилия)

Руководитель курсовой работы _____ (Подпись, дата) Гаврилова Ю. М. (И. О. Фамилия)

2024 г.

СОДЕРЖАНИЕ

Введение

В настоящее время технологии интернета вещей (IoT) становятся неотъемлемой частью нашей повседневной жизни, и одним из наиболее заметных примеров их применения является IoT-платформа для умного дома. Она объединяет различные устройства в доме, от светильников до умных термостатов, в единую сеть, которая может управляться и контролироваться через интернет.

Основной целью IoT-платформы умного дома является создание интеллектуальной инфраструктуры, способной адаптироваться к потребностям и предпочтениям пользователей. Она обеспечивает возможность контролировать освещение, отопление, кондиционирование воздуха, безопасность и другие аспекты жизни в доме с помощью смартфона или другого устройства с доступом в интернет.

С учетом быстрого развития технологий IoT и роста спроса на умные системы, платформы умного дома становятся все более интегрированными, расширяя свои возможности и предлагая новые функции для улучшения качества жизни пользователей.

Целью курсовой работы является разработка базы данных для IoT-платформы умный дом. Для достижения поставленной цели необходимо выполнить следующие задачи:

1. провести анализ существующих решений;
2. сформулировать требования к разрабатываемой базе данных;
3. проанализировать существующие базы данных на основе данной задачи;
4. спроектировать и разработать базу данных;
5. спроектировать и разработать приложение для взаимодействия с базой данных;
6. провести исследование зависимости времени выполнения запросов с использованием индексов и без.

1 Аналитическая часть

1.1 Существующие решения

Так как IoT-вещей активно развиваются, то на рынке уже представлены различные платформы умных домов. Рассмотрим наиболее популярные из них:

— Apple Homekit

— Intel IoT Platform

— MTS IoT HUB

— Xiaomi MI

Выделим следующие критерии для сравнения выбранных платформ:

1) Многопользовательский доступ

2) Возможность создания нескольких домов

3) История работы устройств

Сравнение выбранных платформ по указанным критериям представлены в таблице 1.1:

Таблица 1.1 – Сравнение существующих решений

Решение	1	2	3
Apple Homekit	+	+	-
Intel IoT Platform	-	-	+
MTS IoT HUB	-	+	+
Предполагаемое решение	+	+	+

Таким образом, ни одна из плафторм не удовлетворяет всем критериям.

1.2 Формулирвока требований к разрабатываемой базе данных и приложению

В ходе выполнения курсовой работы необходимо разработать базу данных для хранения ифнормации о пользователях, умных домах, устройствах.

Помимо этого, нужно спроектировать Web-приложение, которое будет предоставлять интерфейс для взаимодействия с базой данных с возможностью создавать умные дома, добавлять новые устройтва в свой дом, просматривать историю работы устройств.

Необходимо также предусмотреть многопользовательский доступ к дому, возможность добавления других пользователь в свой дом для совместного управления. Требуется реализовать функциональность для разных категорий пользователей, каждый из которых получает свой определенный набор прав.

1.3 Формализация данных

Разрабатываемая база данных для IoT платформы умного дома должна содержать информаицю о пользователях, устройствах, умных домах, истории работы устройств. Данные категории показаны на ER-диаграмме в нотации Чена 1.1.

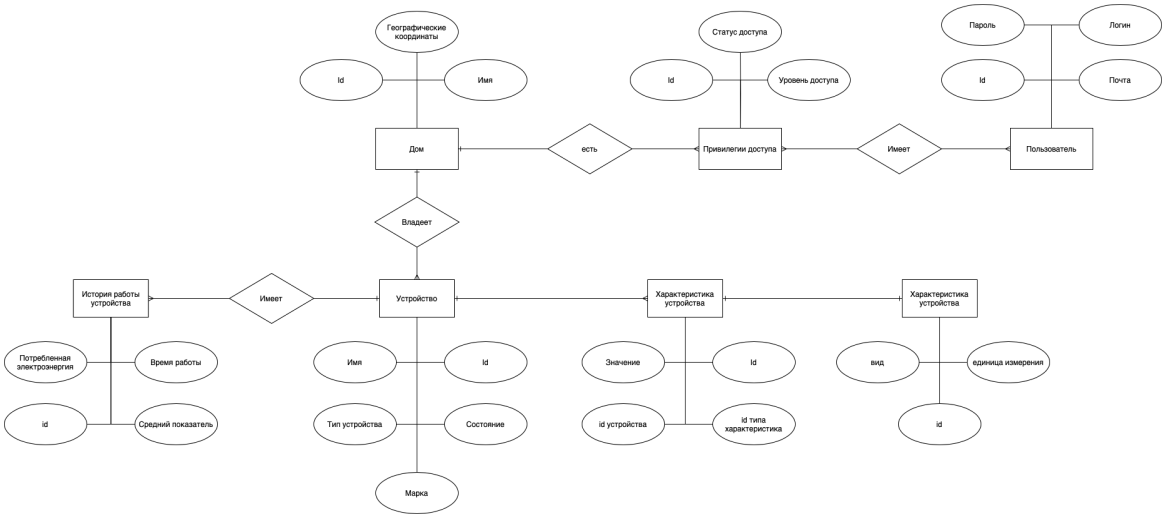

Рисунок 1.1 – ER-диаграмма в нотации Чена

1.4 Формализация пользователей

В зависимости от типа пользователя должен различаться набор доступных действий, таким образом были выделены следующие категории пользователей:

1. Гость – пользователь, имеющий возможность зарегистрироваться или авторизоваться на IoT-платформе умного дома.

2. Авторизованный пользователь – пользователь, который прошел прошел авторизацию. Он получает возможность создать собственный дом или присоединиться к уже существующему.

3. Владелей дома – пользователь, который создал дом. Ему доступны все возможные функции управления домом.

4. Участник дома – пользователь, который имеет доступ к дому другого пользователя. Набор доступных действий определяет хозяин дома.

Все доступные действия для каждой категории пользователей представлены на следующей Use-case диаграмме 1.2

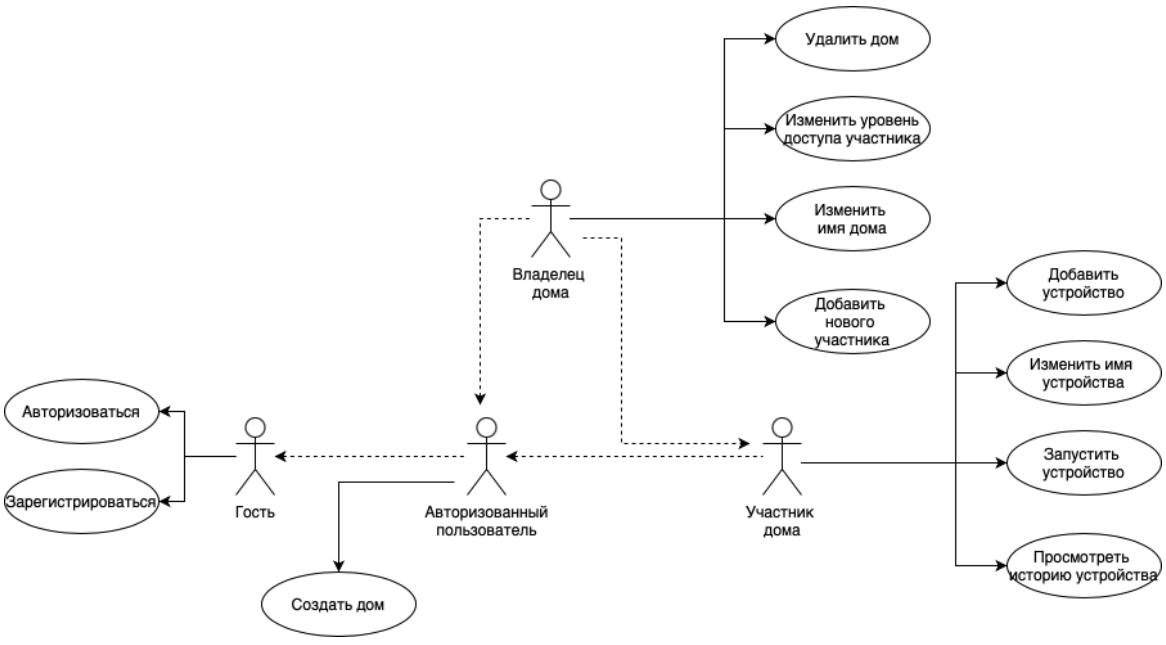

Рисунок 1.2 – Use-case диаграмм пользователей

1.5 Анализ существующих баз данных на основе формализации задачи

База данных – **самодокументированное** собрание **нтегрированных** записей. Рассмотрим части данное определения:

1. База данных является самодокументированной, то есть содержит описание собственной струкутры, которое называется словарем данных, каталогом данных или метаданными.

2. База данных – собрание интегрированных записей, она содержит:

   — файлы данных,

   — метаданные,

   — индексы,

   — может содеражтьетаданные приложений.

3. База данных является информационной моделью пользовательской модели предметной области.

Модель базы данных определяет логическую структуру базы данных и то, каким образом данные будут храниться, организовываться и обрабатываться.

Существует три основных типа моделей базы данных:

— дореляционные;

— реляционные;

— постреляционные.

1.5.1 Дореляционная база данных

К дореляционным моделям баз данных относятся иерархическая и сетевая модели.

Иерархическая модель состоит из объектов с указателями от родительских объектов к потомкам, соединяя вместе связанную информацию. Она может быть представлена в виде дерева. Одним из больших минусов данной модели является невозможность отношения "многие-ко-многим"

Основными понятиями сетевой модели базы данных являюся узел и связь. Узел – совокупность атрибутов данных, описывающих некоторый объект. Данная база данных может быть предствлена в виде графа. При изменении структуры данной модели придется изменять и приложние, так как логика процедуры выборки данных зависит от физической организации этих данных.

1.5.2 Реляционная база данных

В реляционных моделях данные организованы в набор двумерных взаимосвязанных таблиц. Каждая из которых представляет собой набор столбцов и строк, где столбец представляет атрибуты сущности, а строки представляют записи. Такое представление обеспечивает простой и эффективный способ хранения структурированной информации, доступа к ней, а также легкую сортировку.

Также в данной модели происходит разделение между физическим и логическим уровнями, что позволяет управлять физической системой хранения, не меняя данных, содержащихся в логической структуре.

1.5.3 Постреляционные

Нереляционная база данных —- это база данных, в которой в отличие от большинства традиционных систем баз данных не используется табличная схема строк и столбцов. В этих базах данных применяется модель хранения, оптимизированная под конкретные требования типа хранимых данных. Например, данные могут храниться как простые пары "ключ — значение документы JSON или граф, состоящий из ребер и вершин. Все эти хранилища данных не используют реляционную модель.

Недостатком такой модели является сложность решения проблемы обеспечения целостности и непротиворечивости хранимых данных.

Вывод

В данном разделе проведен анализ аналогов IoT-платформ для умного дома. Ни одно из исследованных решений не соответствовало всем установленным критериям сравнения. При рассмотрении моделей баз данных было принято решение в пользу реляционной модели. Это обосновано необходимостью обеспечения целостности хранящихся данных в разрабатываемой базе

данных для IoT-платформы, а также простотой хранения структурированной информации и возможностью ее сортировки.

Кроме того, были формализованы поставленная задача, данные и категории пользователей.

2 Конструкторская часть

В данном разделе будут описаны сущности и построена диаграмма базы данных. Также будет формализована ролевая модель на уровне базы данных и описаны проектируемые функции.

2.1 Сущности базы данных

На рисунке 2.1 представлена диаграмма разрабатываемой базы данных.

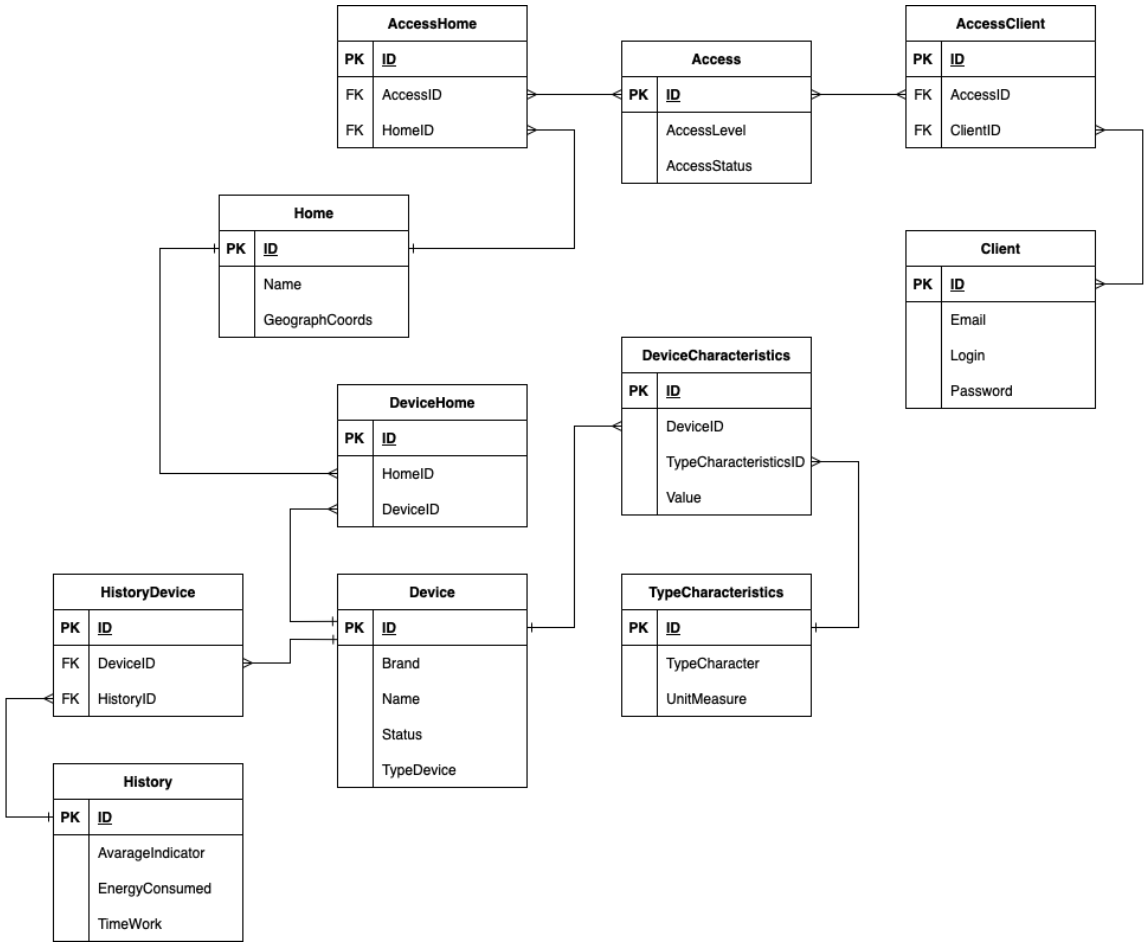

Рисунок 2.1 – диаграмма разрабатываемой базы данных

На диаграмме представлены следующие таблицы:

1) Client – таблица, содержащая ифнормацию о клиентах, состоит из следующих компонентов:
   - ID – уникальный идентификатор(serial4), первичный ключ таблицы;
   - Email – почта пользователя (символьный тип с переменным значением, непустая строка)
   - Login – логин пользователя (символьный тип с переменным значением, непустая строка)
   - Password – пароль пользователя (символьный тип с переменным значением, непустая строка), который в базе данных хранится как хешированная строка;
2) Access – таблица привилегий доступа, содержащая:
   - ID – уникальный идентификатор(serial4), первичный ключ таблицы;
   - AccessLevel – уровень доступа пользователя к дому;
   - AccessStatus – текущий доступ к системе умного дома;
3) Home – таблица, предоставляющая информацию о домах, содержит следующие поля:
   - ID – уникальный идентификатор(serial4), первичный ключ таблицы;
   - Name – имя дома;
   - GeographCoords – географические координаты, показывающие расположение дома;
4) Device – таблица, предоставляющая информацию об устройствах, содержит следующие поля:
   - ID – уникальный идентификатор(serial4), первичный ключ таблицы;
   - Name – имя устройства;
   - Brand – бренд устройства;
   - Status – текущее состояние устройства;

— TypeDevice – тип устройства;

5) History – таблица, содержащая историю работы устройств, содержит следующие поля:

— ID – уникальный идентификатор(serial4), первичный ключ таблицы;

— AvarageIndicator – средний показатель характеристики устройства;

— EnergyConsumed – потребленная энергия устройством за время работы;

— TimeWork – время работы устроства;

6) AccessClient – таблица, которая реализует отношение один-ко-многим» между таблицами «Client» и «Access», имеет два уникальных идентификатора (клиент и доступ).

7) AccessHome – таблица, которая реализует отношение один-ко-многим» между таблицами «Home» и «Access», имеет два уникальных идентификатора (дом и доступ).

8) DeviceHome – таблица, которая реализует отношение «многие-ко-многим» между таблицами «Device» и «Home», имеет два уникальных идентификатора (устройство и дом).

9) HistoryDevice – таблица, которая реализует отношение «многие-ко-многим» между таблицами «Device» и «History», имеет два уникальных идентификатора (устройство и история).

2.2 Функции базы данных

Когда устройство начинает работать, необходимо обновлять статус для того, чтобы пользователь мог увидеть текущее состояние устройства. Для этого была описана функция, которая после окончания работы обновляет статус устройства с «Active» на «Inactive» и при запуске устройства с «Inactive» на «Active». Схема алгоритма данной функции указана на следующем рисунке 2.2:

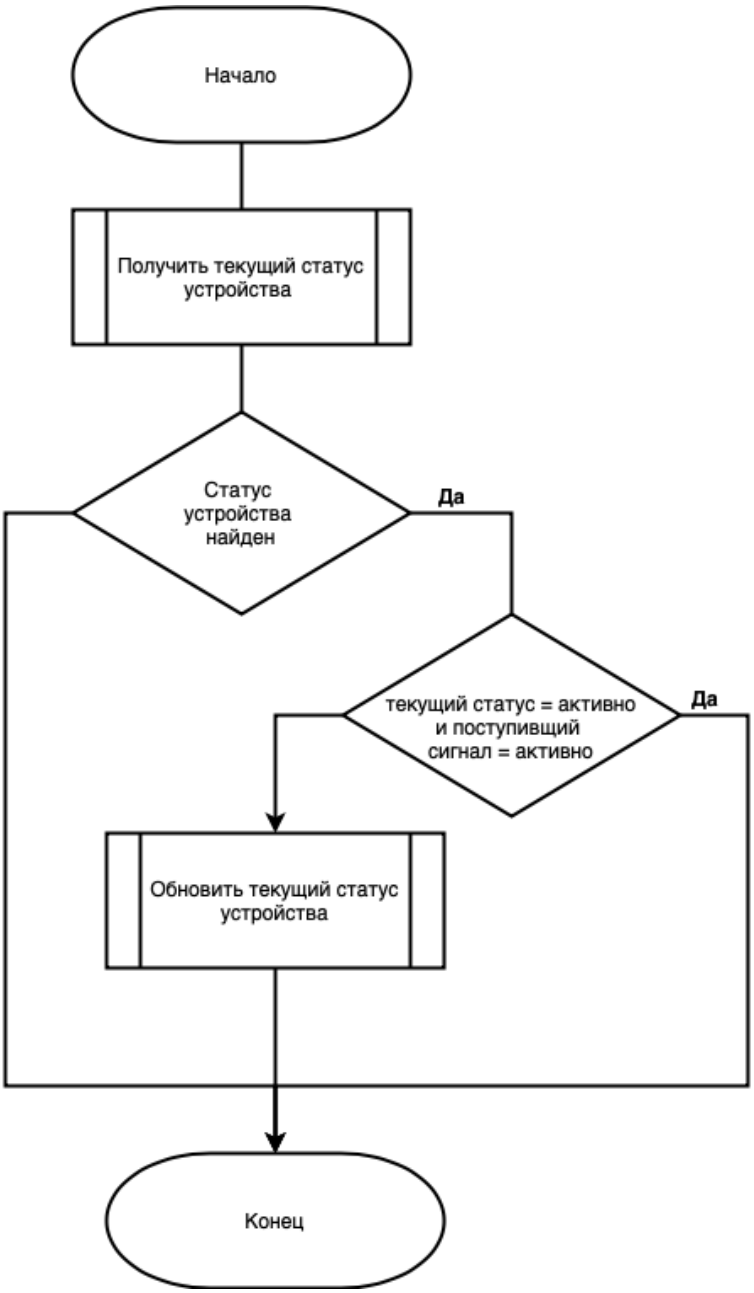

Рисунок 2.2 – ER-диаграмма в нотации Чена

2.3 Роли базы данных

Для управления доступом к таблицам применяются роли. В аналитической части были установлены следующие роли: гость (неавторизованный пользователь), авторизованный пользователь, владелец дома и участник дома. Для каждой из этих ролей необходимо разработать соответствующие права доступа:

1) гость(неавторизованный пользователь) должен иметь доступ на изменение и просмотр данных в таблице «Client»;

2) авторизованный пользователь имеет все права гостя, так же ему предоставлены изменение и добавление данных в таблицу «Home»;

3) участник дома должен иметь все права авторизованного пользователя, так же ему доступно просмотр таблиц «History», «Device», «TypeCharacteristics», «Access», в зависимости от предоставленных прав владельцем дома участнику может быть доступно добавление, изменение данных в указанных таблицах;

3) владелец должен иметь возможность просматривать и изменять записи во всех существующих таблицах.

Вывод

В данном разделе были описаны сущности и построена диаграмма базы данных, описана ипользуемая функция и приведена схема ее алгоримта. Так же были формализованы роли на уровене базы данных, чтобы обеспечить управление доступом к таблицам.

3 Технологическая часть

В данном разделе происходит выбор средств реализации базы данных и приложения, листинги кода, а так же будет показан интерфейс программы.

3.1 Выбор системы управления базой данных

В аналитическом разделе была выбрана реляционная система управления, наиболее распространнеными представителями данной системы являются: Microsoft SQL, PostgreSQL, Oracle, а также MySQL.

Выберем следующие критерии для сравнения выбранных систем:

1) возможнсоть бесплатного использования;

2) опыт работы с данной системой;

3) наличие подробной документации.

Сравнение указанных систем управления базой данных(СУБД) представлены в таблице 3.1:

Таблица 3.1 – Сравнение СУБД по указаным критериям

СУБД	Microsoft SQL	PostgreSQL	Oracle	MySQL
1	-	+	-	+
2	-	+	-	-
3	+	+	+	+

По результатам сравнения для реляционной была выбрана PostgreSQL, так как это единственная система, которая удовлетворяет всем критериям.

3.2 Выбор средств реализации

Реализация части приложения, которая обеспечивает доступ к базе данных, была выполнена с использованием языка программирования GoLang. Данный выбор обусловлен следующими причинами:

— сборщик мусора, который позволяет автоматически освобождать память;

— статическая типизация, которая помогает обнаруживать ошибки на этапе компиляции;

— встроенная поддержка параллельных вычислений, что позволяет делать разработку многопоточной;

Графический интерфейс был разработан с использованием HTML и CSS, так же был выбран язык JavaScript для обработки событий и динамического обновления элементов.

Причина выбора данных средств заключается в следующем:

— HTML и CSS поддерживаются всеми основными веб-браузерами, что гарантирует доступность содержимого на различных устройствах;

— опыт работы с HTML и CSS;

— JS имеет возможность получать доступ к структуре HTML-документа, что позволяет манипулировать элементами, стилями, атрибутами и событиями.

3.3 Выбор среды разрабоки

В качестве среды разработки был выбран Visual Studio Code. Данный выбор обусловлен следующими причинами:

— бесплатный доступ;

— поддержка выбранных языков;

— опыт работы в данной среде.

3.4 Архитектура приложения

Для разрабатываемого приложения была выбрана чистая архитектура, то есть приложение будет состоять из трех слоев: графический интерфейс, бизнес логика, доступ к данным.

Данная разделение имеет следующие преимущества:

— приложение не зависимосит от используемых библиотек и фреймворков;

— удобство тестирования;

— возможность использования различных баз данных, так как бизнес логика не зависит напрямую от используемой базы данных;

3.5 Реализация функции

3.6 Тестирование

3.7 Графический интерфейс