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 Робототехника и комплексная автоматизация (РК)

КАФЕДРА Системы автоматизированного проектирования (РК-6)

РАСЧЕТО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА К КУРСОВОМУ ПРОЕКТУ НА ТЕМУ:

«*Разработка информационной системы для магазина цветов*»

Студент РК6-54Б _____ **Глаголев Т. С.**
подпись, дата *фамилия, и.о.*

Преподаватель _____ **Пивоварова Н. В.**
подпись, дата *фамилия, и.о.*

Оценка _____

Москва, 2024 г.

Описание предметной области

Диспетчер цветочного салона принимает заказы от клиентов. Фамилии клиентов и их адреса и телефоны сохраняются в базе данных салона. Также сохраняются даты заказов для каждого клиента.
Клиент может заказать любое количество букетов из имеющегося списка.
Для букетов известны номер букета, название, стоимость.
Сформировав список заказа, диспетчер согласовывает время доставки и назначает курьера из списка доступных в указанное время курьеров.
О каждом курьере известны его уникальный номер, фамилия, дата рождения, дата приема на работу, дата увольнения, адрес, телефон.
Для оценки популярности букетов в службе каждый месяц составляется и сохраняется в БД отчет по форме:
Уникальный шифр букета, количество букетов, общая стоимость букетов, месяц, год.

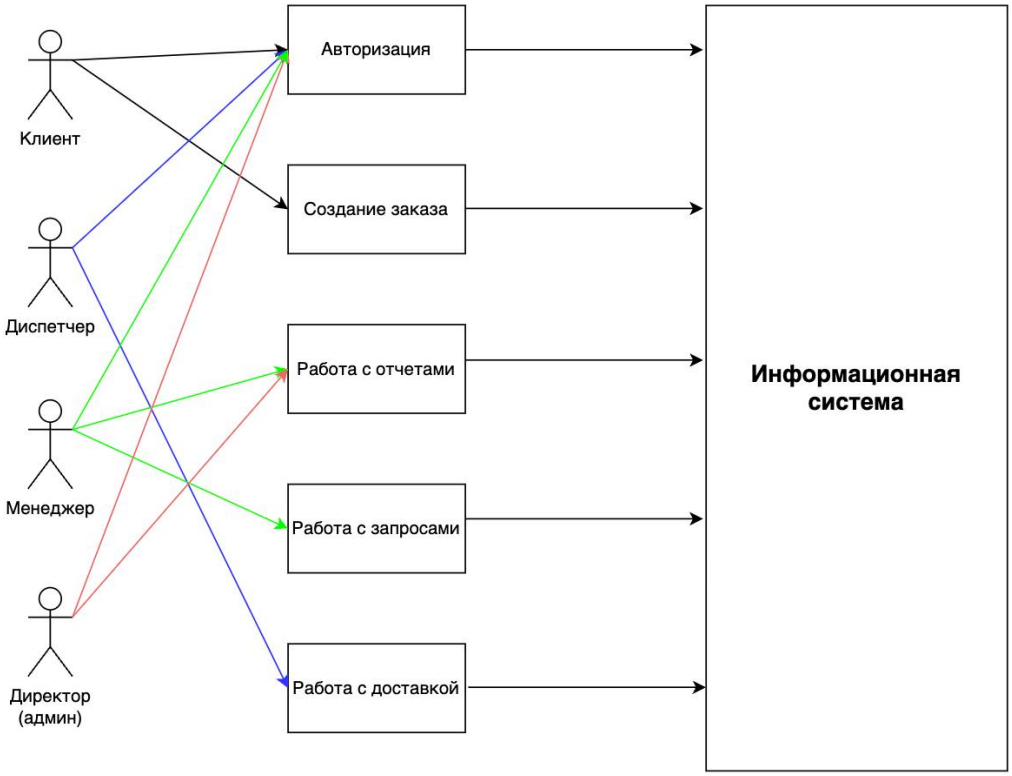
Определение конечных пользователей

В данной информационной системе в качестве конечных пользователей выделим следующих актеров:

- клиент
- диспетчер
- менеджер
- директор(админ)

Диспетчер, менеджер, директор являются внутренними пользователями. Клиент является внешним пользователем

Иерархия внутренних пользователей (в порядке возрастания): диспетчер – менеджер - директор



АВТОРИЗАЦИЯ ПОЛЬЗОВАТЕЛЕЙ

Предусловия: пользователь должен быть зарегистрированным в ИС
Гарантия: при авторизации пользователю будет позволено взаимодействовать с ИС в рамках его роли
Минимальная гарантия: В случае ошибки пользователь получить сообщение о ней и возможность повторить ввод

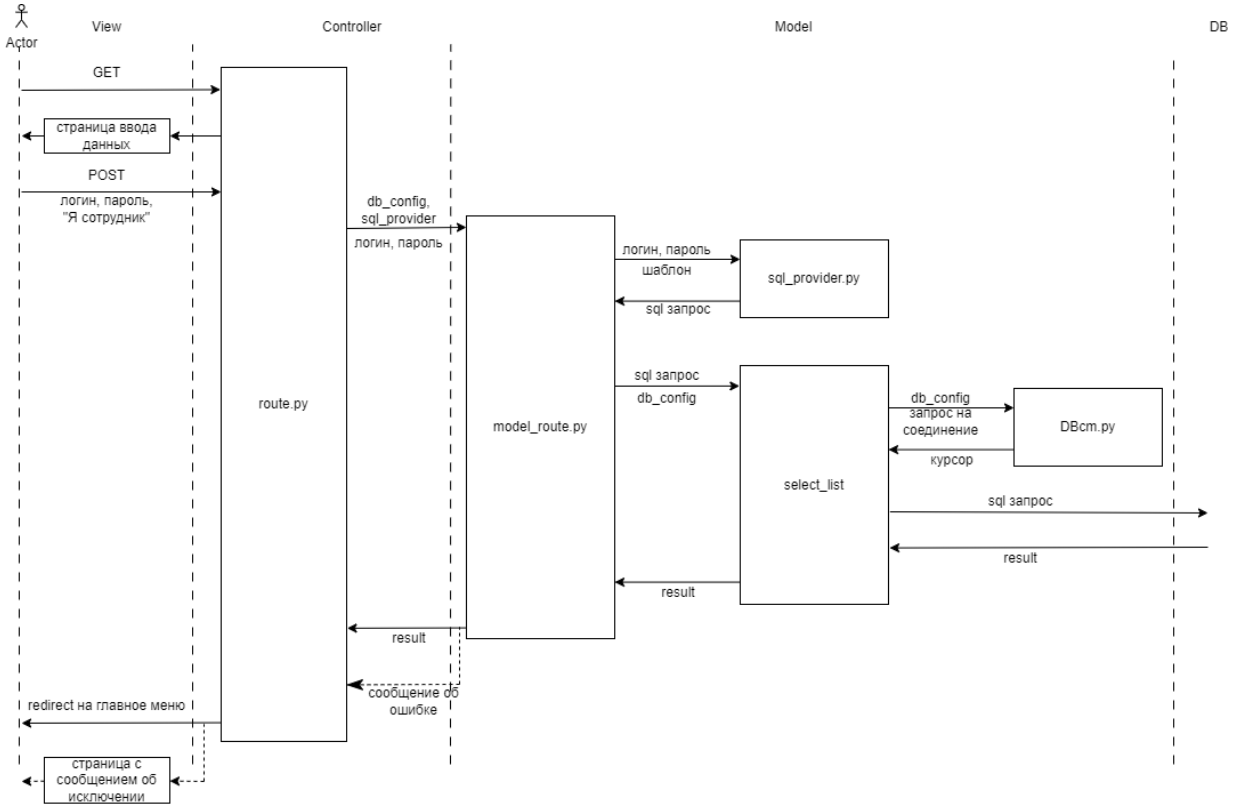
Удачный сценарий авторизации:

1. Пользователь входит в систему
2. Система присылает форму авторизации
3. Пользователь вводит логин и пароль, проставляет отметку сотрудника и отправляет их в систему
4. Система перенаправляет в личный кабинет пользователя

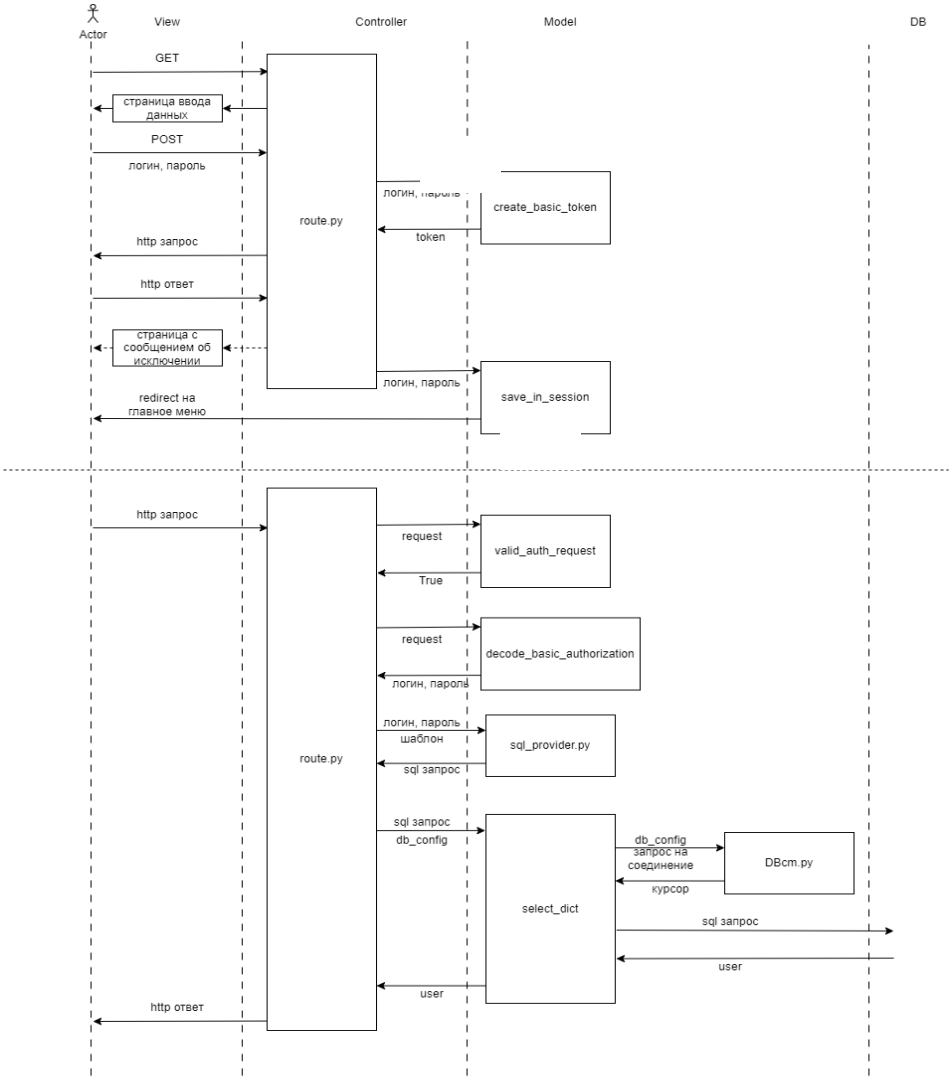
Исключения:

3.1 Если система не находит введенные логин и пароль, система выводит об этом сообщение и предлагает ввести логин и пароль заново

Авторизация внутреннего пользователя:



Авторизация внешнего пользователя



Тестовые данные для внутренних и внешних пользователей:

Успешный ввод:

Логин	Пароль	Я сотрудник
manag1	manag1	+
dis1	dis1	+
admin	admin	+
tim	glagolev	

Неуспешный ввод:

логин	пароль	Я сотрудник
login1	pass1	+
manag1	manag1	

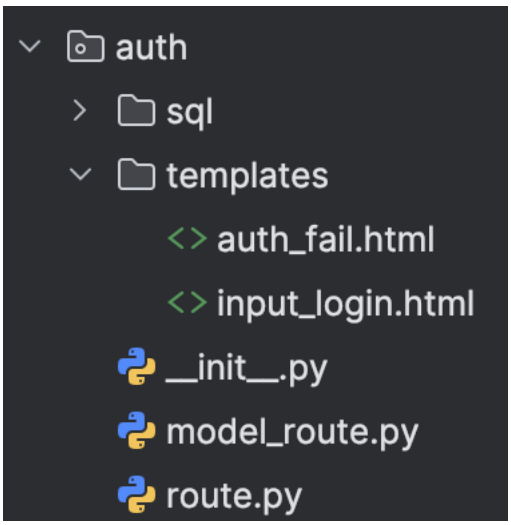
admin		+
tim	glagolev	+

Требование к шаблону авторизации

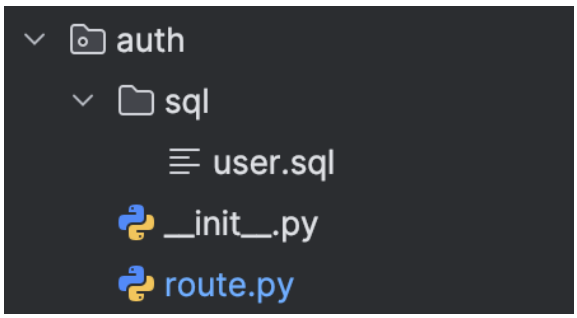
Страница выполнена в едином стиле. В верхней части страницы расположен заголовок с текстом "Введите логин и пароль", который выровнен по центру страницы. Ниже заголовка находится форма для ввода данных. В форме есть три поля: одно для ввода логина, второе — для пароля, и третье — флажок, с помощью которого можно отметить, является ли пользователь сотрудником. Все поля и элементы формы расположены вертикально и выровнены по центру. Под полями для ввода находится кнопка "Отправить" для отправки данных. Вся страница также выровнена по центру, создавая аккуратный и симметричный вид. Внизу страницы может отображаться сообщение с результатом обработки формы.

структура blueprint

структура основного сервиса



структура микро сервиса



Структура основного сервиса:

- **route.py**

В этом файле описана обработка маршрутов для аутентификации пользователей. При получении GET-запроса отображается форма для входа, а при отправке формы (POST-запрос) происходит проверка учетных данных пользователя. В зависимости

от типа пользователя (внутренний или внешний) выполняется аутентификация, создается сессия и происходит перенаправление на соответствующую страницу. В случае ошибки показывается сообщение об ошибке. Этот файл также отвечает за определение маршрутов и взаимодействие с моделью.

- **model_route.py**

В файле модели реализована логика работы с данными пользователей, включая аутентификацию, создание и проверку учетных данных, а также управление сессиями. Например, для каждого типа пользователя (внутренний или внешний) выполняется проверка, создается зашифрованный токен для микро-сервиса и управляется состояние сессии. Модель отвечает за обработку и проверку данных, в то время как route.py просто вызывает эти функции для отображения информации или перенаправления пользователя.

- **user.sql**

в файле содержится запрос, который извлекает данные для создания сессии из таблицы с информацией обо всех внутренних пользователях. Запрос выполняется для конкретного пользователя, идентифицируемого по паролю и логину.

- **input_login.html**

в файле описывается шаблон, который отображает страницу авторизации

- **auth_fail.html**

шаблон, отображающий ошибку доступа

Структура микро сервиса:

- **route.py**

в файле писаны основные функции получения токена от основного сервиса, расшифровка токена и нахождения пользователя в ИС, обработчик маршрута /find-user, который ищет пользователя по логину и паролю. Он проверяет авторизацию, декодирует данные и, если пользователь найден, возвращает его информацию с кодом 200. В случае ошибок возвращаются соответствующие сообщения с кодами 400 или 404.

- **user.sql**

в файле содержится запрос, который извлекает данные для создания сессии внешнего пользователя. Запрос выполняется для конкретного пользователя, идентифицируемого по паролю и логину.

РАБОТА С ЗАПРОСАМИ

Предусловие: пользователь должен быть авторизован, зарегистрирован в ИС, иметь разрешение на работу с запросами

Гарантия минимальная: никаких изменений в БД

Гарантия максимальная: пользователь получит результат

Успешный сценарий работы с меню:

1. Пользователь начинает работу с меню
2. Система отправляет на страницу меню с выбором запросов
3. Пользователь выбирает запрос
4. Пользователь перенаправляется на соответствующую страницу для работы с выбранным запросом

Удачный сценарий работы с запросами:

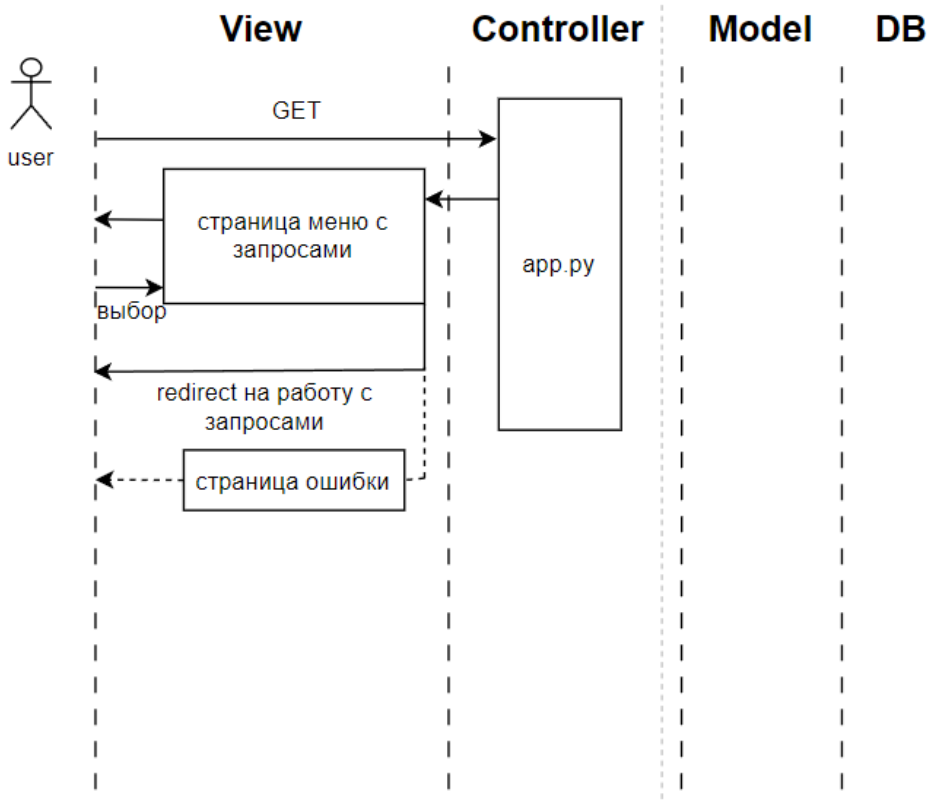
1. Пользователь начинает сценарий, перейдя по соответствующему URL.
2. Система присылает статичную страницу на выбор запроса
3. Пользователь выбирает запрос
4. Система перенаправляет на соответствующую HTML страницу
5. Система присылает форму для ввода параметров запроса.
6. Пользователь вводит параметры и отправляет их системе.
7. Система отправляет sql запросы и обрабатывает в соответствии с введенными параметрами и отправляет результат.
8. Пользователь заканчивает работу с системой и возвращается в главное меню.

Исключения:

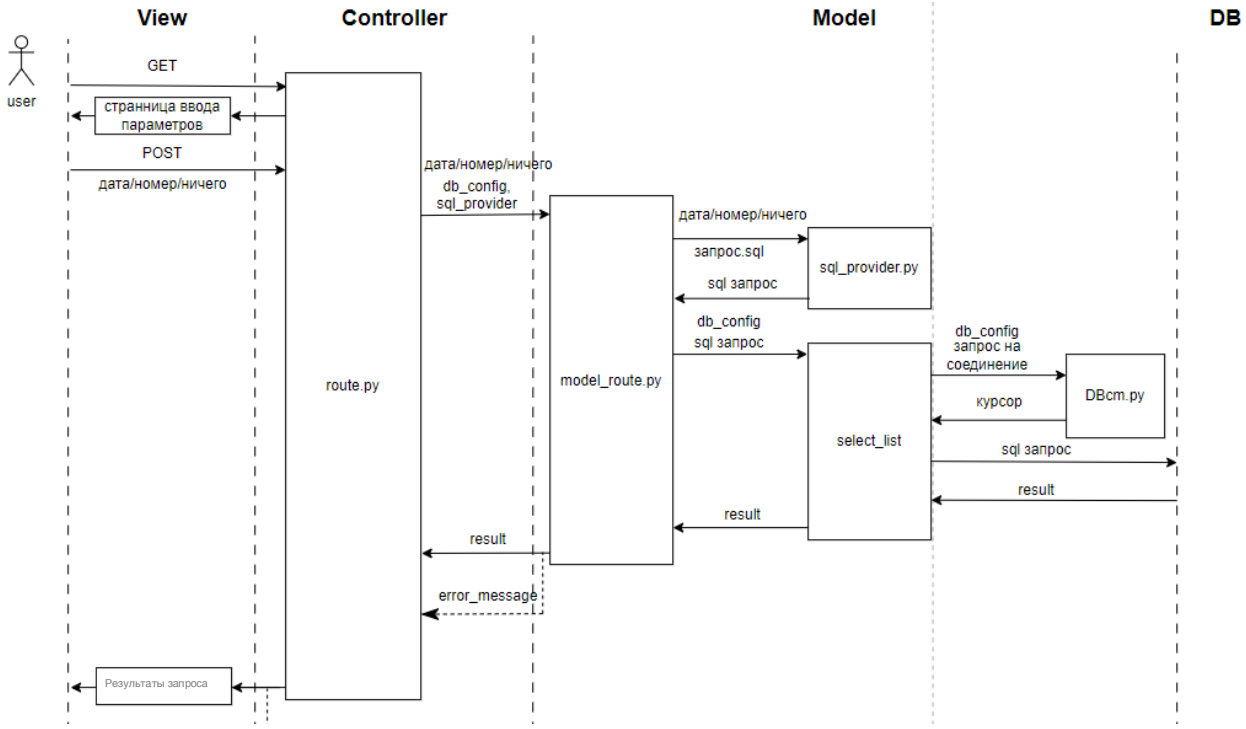
6.а) Параметры введены некорректно либо же их нет, система присылает сообщение, что результат не получен и предлагает вернуться к шагу 4

6. б) параметры введены корректно, но вернулся пустой

Меню запросов



Работа с запросами



Тестовые данные:

Клиенты за период:

Успешный ввод:

Год	Месяц 1	Месяц 2
2013	6	7
2024	2	4

Неуспешный ввод:

Год	Месяц 1	Месяц 2
2014	6	7
2024		4
2024	6	

Просмотр минимального заказа в году:

Успешный ввод:

Номер телефона
2010
2024
2020

Неуспешный ввод:

Номер телефона
2023
2011
2012

Требования к меню выбора запроса

В верхней части страницы расположен заголовок "Простые запросы". Под ним находится список ссылок для выполнения различных запросов:

- "Клиенты за период".
- "Минимальная цена заказа".
- "Курьеры без заказов".

В нижней части страницы расположена ссылка "Назад", оформленная в едином стиле со ссылками из списка.

Требование к шаблону вывода пользователей, зарегистрировавшихся в выбранный промежуток времени

В верхней части страницы расположена прямоугольная форма с закруглёнными краями цвета, в которой находится форма ввода. В форме расположены:

- Поле для ввода года.
- Поле для ввода начального месяца периода.
- Поле для ввода конечного месяца периода.
- Кнопка "Выполнить запрос".

Под формой отображается таблица с результатами, если запрос возвращает данные. Если данных нет, выводится сообщение об отсутствии результатов. В нижней части страницы расположена ссылка "Назад", оформленная в едином стиле с остальными кнопками.

Требование к шаблону вывода курьеров без заказов

Страница отображает список курьеров, у которых нет заказов. Содержит следующие элементы:

1. Заголовок с названием страницы.
2. Таблица для отображения курьеров без заказов, если данные есть.
3. Сообщение о том, что результатов нет, если запрос не возвращает данных.
4. Ссылка для возвращения на главную страницу запросов.

Под формой отображается таблица с результатами, если запрос возвращает данные. Если данных нет, выводится сообщение об отсутствии результатов. В нижней части страницы расположена ссылка "Назад", оформленная в едином стиле с остальными кнопками.

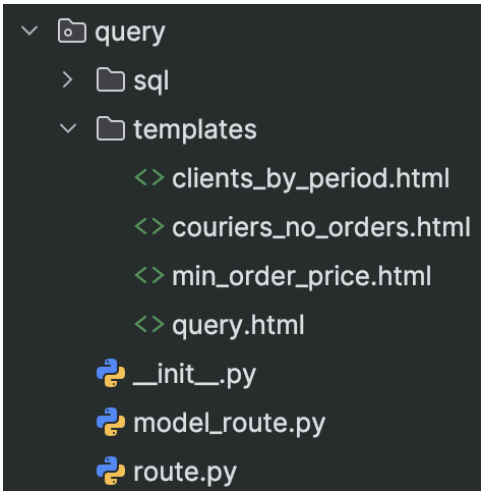
Требование к шаблону вывода заказа с минимальным ценником за конкретный год

Страница предоставляет форму для ввода года, чтобы получить минимальную цену заказа за этот период. Содержит следующие элементы:

1. Поле для ввода года.
2. Кнопка для выполнения запроса.
3. Результат минимальной цены, если данные есть.
4. Ссылка для возвращения на главную страницу запросов.

Под формой отображается таблица с результатами, если запрос возвращает данные. Если данных нет, выводится сообщение об отсутствии результатов. В нижней части страницы расположена ссылка "Назад", оформленная в едином стиле с остальными кнопками.

структура blueprint



В этом файле описаны маршруты для обработки запросов и отображения соответствующих HTML-шаблонов.

- **model_route.py** — файл содержит всю бизнес-логику: подготовку параметров для SQL-запросов, выполнение запросов с использованием SQL-провайдера и обработку полученных данных. Он включает функции для получения списка клиентов за указанный период, минимальной стоимости заказа за год, а также курьеров без заказов.
- **route.py** — файл отвечает за обработку HTTP-запросов и связывает их с функциями из model.py. Он принимает данные из формы запроса, передает их в модель для обработки, а затем возвращает результат в соответствующий шаблон. Маршруты предоставляют доступ к страницам с запросами, включая клиентов за период, минимальную стоимость заказа и курьеров без заказов.
- **clients_by_period.sql**
  Содержит SQL-запрос для получения информации о клиентах за определённый период на основе даты.
- **couriers_no_orders.sql**
  Содержит SQL-запрос для поиска информации о курьерах, у которых не было заказов за определенный период.
- **min_orders_price.sql**
  Содержит SQL-запрос для поиска минимальной цены заказа среди заказов за определённый год.

- **clients_by_period.html**
  HTML-шаблон для отображения данных о клиентах за указанный период, с возможностью фильтрации и представления информации в удобном формате.
- **couriers_no_orders.html**
  HTML-шаблон для отображения информации о курьерах, у которых отсутствуют заказы за указанный год.
- **min_orders_price.html**
  HTML-шаблон для отображения информации о минимальной цене заказа за определенный год.
- **query.html**
  Шаблон с формой для ввода параметров поиска, таких как даты и фильтры, используемых для формирования отчетов.

РАБОТА С ОТЧЁТАМИ

Предусловия: пользователь должен быть авторизован и зарегистрирован в ИС

Гарантия при создании отчёта: в БД будет создан отчёт

Гарантия при просмотра отчёта: никаких изменений в БД

Удачный сценарий использования

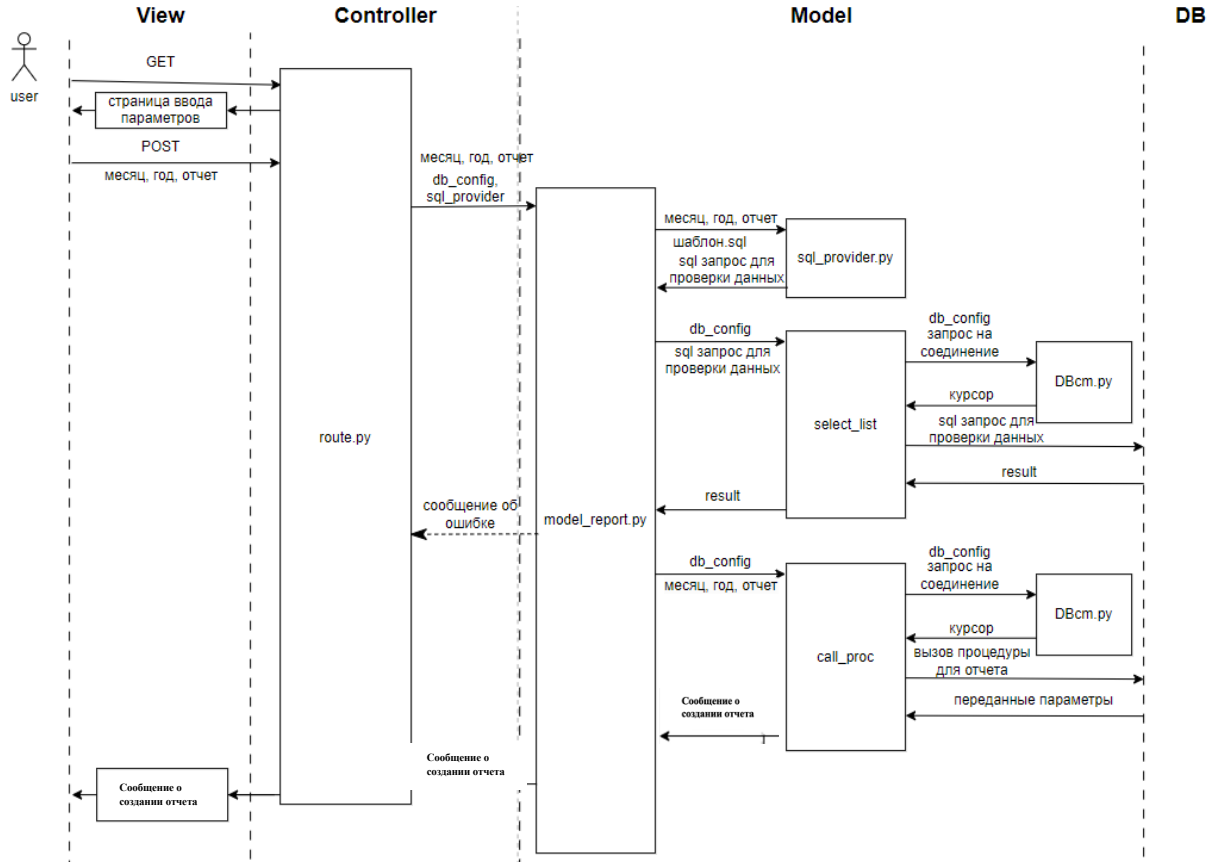
1) Пользователь начинает работу с отчетами.
2) Система присылает страницу с выпадающим списком доступных отчетов, присылает форму для ввода параметров отчетного периода и кнопками "создать" и "просмотр", зависящих от роли пользователя.
3) Пользователь выбирает тип отчета, вводит параметры и нажимает “создать”.
4) Система присылает сообщением об успешном создании отчета
5) Пользователь выбирает тип отчета, вводит параметры и нажимает “посмотреть”
6) Система присылает данные с таблицы с учетом введенных данных

Исключения:

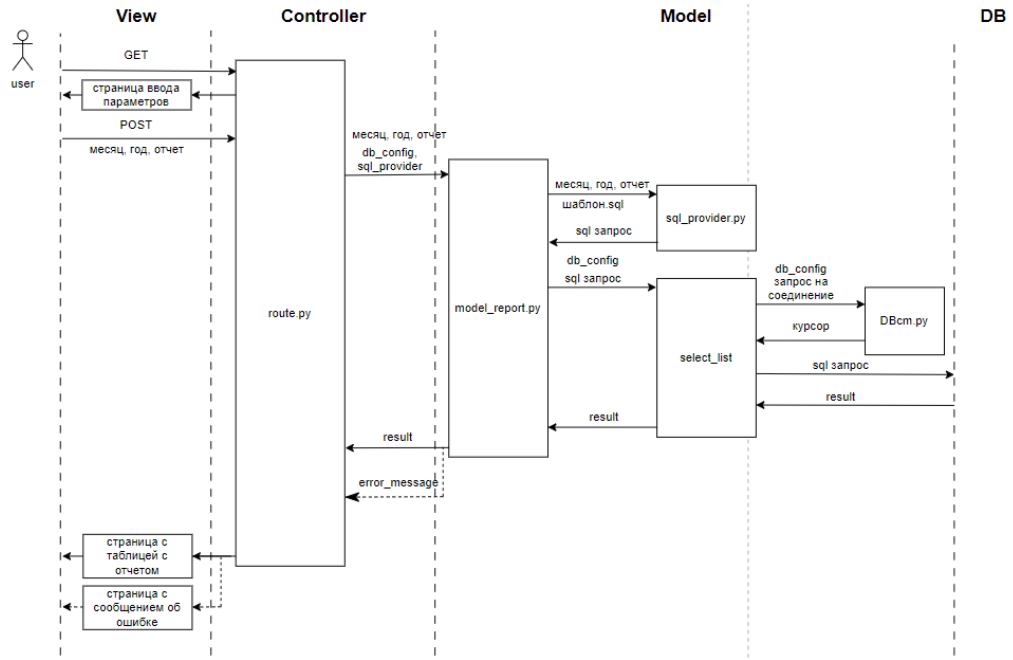
3.a) Отчет за указанный отчетный период уже существует в БД

b) Отчет за указанный отчетный период не существует в БД

Создание отчета



Просмотр отчета



Данные, которые должен содержать отчет

Отчет по курьерам:

Номер курьера	Имя курьера	Количество заказов	Начало работы
…	…	…	

Отчет по букетам:

Номер букета	Количество	Общая сумма
…	…	…

Тестовые данные:

Успешное создание отчета:

Отчёт	**Месяц**	**Год**
О курьерах	Декабрь	2024
О букетах	Декабрь	2024

Неуспешное создание отчета:

Отчёт	**Месяц**	**Год**
О курьерах	Июнь	2024
О букетах	Июнь	2024

Успешный просмотр отчета:

Отчёт	**Месяц**	**Год**
О курьерах	Декабрь	2024
О букетах	Декабрь	2024

Неуспешный просмотр отчета:

Отчёт	**Месяц**	**Год**
О курьерах	Июнь	2024
О букетах	Июнь	2024

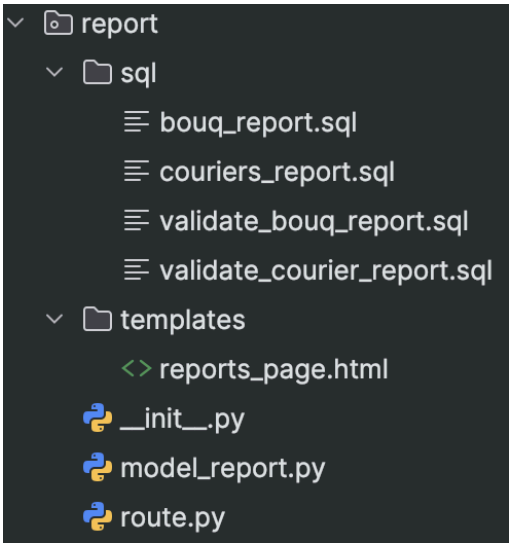
Требования к шаблону взаимодействия с отчётами

Страница оформлена в едином стиле. В верхней части страницы расположен крупный заголовок "Работа с отчетами”. Ниже заголовка отображается сообщение об ошибке или успешном выполнении действия (если оно присутствует). Далее идет форма для выбора параметров отчета, которая включает три поля: выбор месяца, выбор года и выбор типа отчета (по курьерам или по букетам). После выбора параметров, в зависимости от роли пользователя, отображаются кнопки для создания или просмотра отчета. Сотрудники с ролью "менеджер" могут создать

отчет, а пользователи с ролями "администратор" и "менеджер" — просматривать отчеты.

Если отчет был найден, то на странице выводится таблица с результатами за выбранный месяц и год. В таблице отображаются все столбцы и значения отчета. Внизу страницы находится кнопка для возврата на главную страницу. Вся страница выровнена по центру, создавая аккуратный и симметричный вид.

структура blueprint



Структура:

- **model_report.py**
  файл реализует логику работы с отчетами, включая выполнение SQL-запросов и вызов хранимых процедур. Он предоставляет функции для проверки существования отчета, добавления новых записей через вызов процедур, а также получения данных отчетов за определенный период.
- **route.py**
  файл отвечает за маршруты и обработку запросов, связанных с отчетами. Он передает данные из формы в функции модели для проверки, создания или получения отчетов. Также он управляет отображением сообщений и записей в шаблонах в зависимости от роли пользователя, передаваемой через сессию.
- **couriers_report.sql**
  запрос для вывода данных из таблицы report за выбранный год и месяц
- **bouq_report.sql**
  запрос для вывода данных из таблицы report за выбранный год и месяц
- **validate_courier_report.sql**

запрос для проверки записей за выбранный год и месяц в таблице courier_report

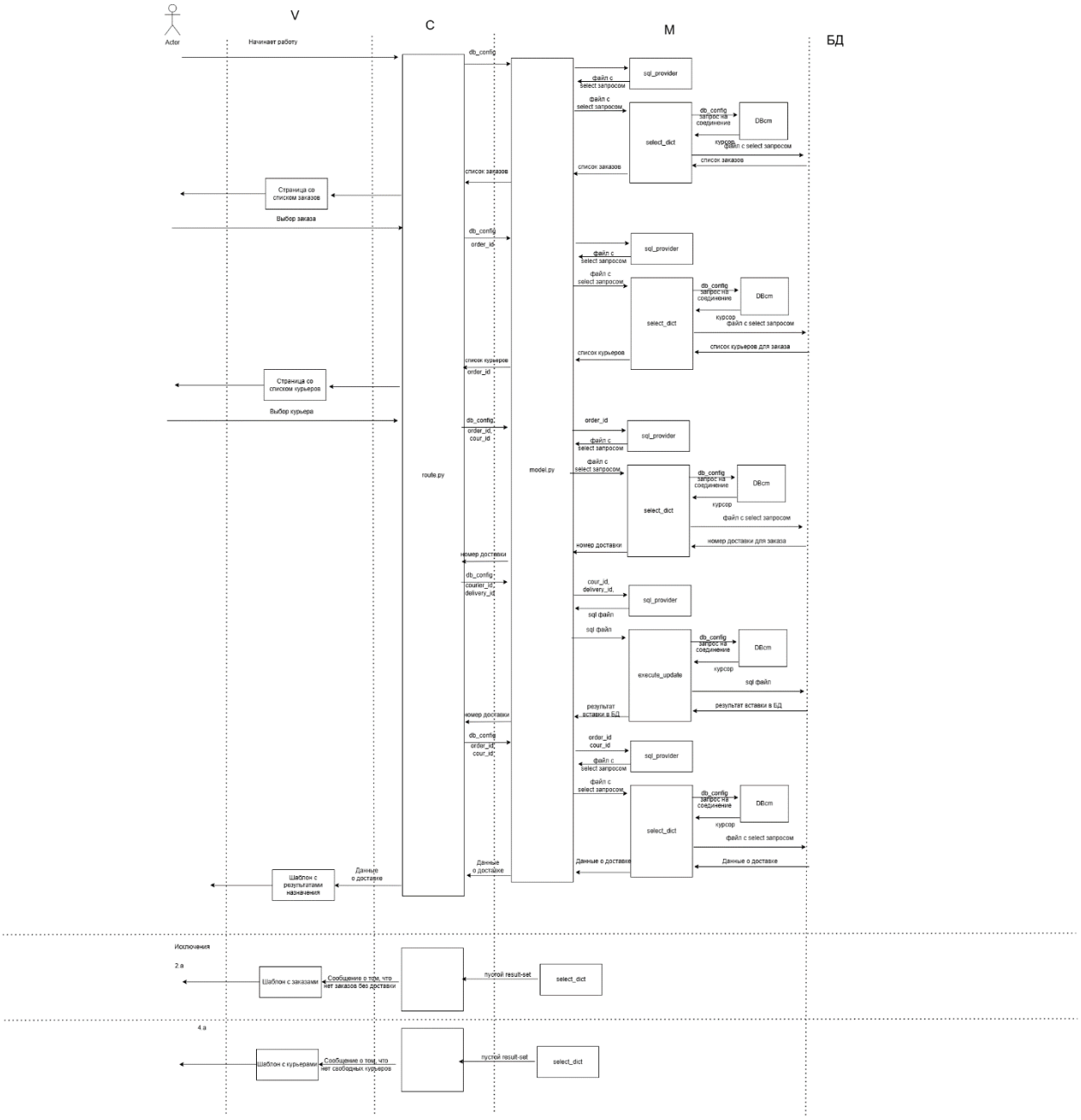
- **validate_bouq_report.sql**
  запрос для проверки записей за выбранный год и месяц в таблице report
- **reports_page.html**
  Основной шаблон для работы с отчетами. предоставляет интерфейс для работы с отчетами. Он отображает форму выбора параметров отчета (месяц, год, тип отчета), кнопки действий для создания или просмотра отчета в зависимости от роли пользователя, а также таблицу с результатами отчета, если данные найдены. При успешном создании или отсутствии данных выводится соответствующее сообщение.

РАБОТА С ДОСТАВКОЙ

Предусловия: пользователь должен быть авторизован, зарегистрирован в ИС, иметь разрешение на работу с запросами
Гарантия при работе с доставкой: в БД будет назначен курьер к заказу и доставке
Минимальная гарантия при работе с доставкой: никаких изменений в БД
Удачный сценарий использования

1. Пользователь начинает работу с доставкой
2. Система присылает шаблон с заказами, где выводятся все заказы, у которых не назначен курьер
3. Пользователь выбирает доступный заказ, нажав на кнопку “Выбрать”
4. Система присылает шаблон со свободными курьерами, у которых статус свободен
5. Пользователь выбирает курьера, нажимая кнопку “Назначить”
6. Система присылает шаблон, в которой указаны нужные данные о заказе и о курьере, которого назначили на заказ

Исключения:
2.а) Нет заказов без доставки
4.а) Нет свободных курьеров



Требования к шаблону orders.html (Выбор заказов без доставки)

В верхней части страницы расположен заголовок "Заказы без доставки", выделенный крупным шрифтом. В центре страницы отображается таблица. Заголовки таблицы: "Номер заказа", "Дата заказа", "Действие". В каждой строке таблицы выводятся данные о заказах, а в колонке "Действие" расположена ссылка "Выбрать курьера". Если заказов без доставки нет, вместо таблицы отображается

сообщение "Нет свободных заказов. В нижней части страницы размещена кнопка "Назад".

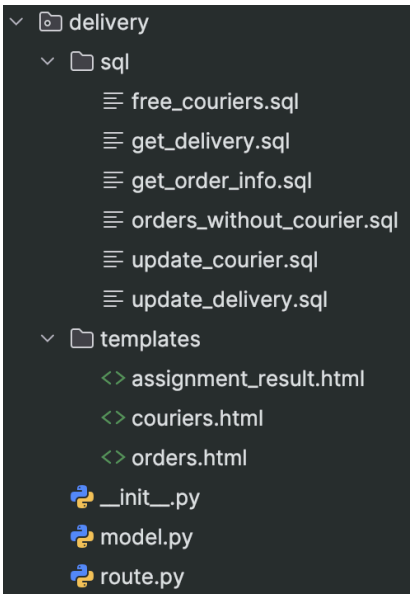
Требования к шаблону couriers.html (Выбор свободных курьеров)

В верхней части страницы расположен заголовок "Свободные курьеры для заказа №{order_id}. В центре страницы находится таблица. Заголовки таблицы: "Номер курьера", "Имя курьера", "Действие". В колонке "Действие" каждой строки расположена форма с кнопкой "Назначить". Если свободных курьеров нет, вместо таблицы отображается сообщение "Нет доступных курьеров. В нижней части страницы расположена кнопка "Назад".

Требования к шаблону assignment_result.html (Результат назначения курьера)

В верхней части страницы расположен заголовок "Результат назначения курьера". Под заголовком размещен прямоугольный блок информации. В блоке отображаются строки с информацией о заказе: номер заказа, имя и ID курьера, телефон курьера, дата создания заказа. Также выделен отдельный блок с данными о доставке: дата доставки, время доставки, место доставки. В верхней части блока находится сообщение "Заказ с ID: успешно обработан." В нижней части страницы расположена ссылка "Вернуться к списку заказов".

Структура blueprint



Структура:

- **model.py**

в файле реализована логика для работы с базой данных. Он включает функции для получения данных о заказах, курьерах и доставках, а также для назначения курьера на заказ и обновления соответствующих статусов в базе данных. Функции выполняют запросы к базе данных через SQL, используя вспомогательные методы для получения данных или выполнения обновлений.

- **route.py**

В файле обрабатываются маршруты для работы с данными, получаемыми из model.py. Эти маршруты обеспечивают отображение страниц с заказами и курьерами, а также обработку формы для назначения курьера на заказ. Каждая страница загружает соответствующие данные с помощью вызовов функций из модели и отображает результаты на веб-страницах.

- **orders.html**

Шаблон для отображения списка заказов без назначенного курьера. Предоставляет таблицу с данными о заказах и ссылками для выбора курьера для каждого заказа.

- **couriers.html**

Шаблон для отображения списка свободных курьеров для конкретного заказа. Показывает таблицу с информацией о курьерах и кнопками для назначения курьера на заказ.

- **assignment_result.html**

Шаблон для отображения результата назначения курьера на заказ. Показывает информацию о заказе, курьере и деталях доставки.

- **free_couriers.sql**

SQL-запрос для получения списка свободных курьеров, которые могут быть назначены на заказы.

- **get_delivery.sql**

SQL-запрос для получения данных о конкретной доставке на основании идентификатора.

- **get_order_info.sql**

SQL-запрос для получения детальной информации о заказе, включая дату создания, адрес доставки и другую связанную информацию.

- **orders_without_courier.sql**

SQL-запрос для получения списка заказов, которым еще не назначен курьер.

- **update_courier.sql**

SQL-запрос для обновления записи курьера, назначая его на конкретный заказ.

- **update_delivery.sql**

SQL-запрос для обновления данных доставки, таких как место, дата или время, связанные с заказом.

ГЛАВНЫЙ БИЗНЕС-ПРОЦЕСС

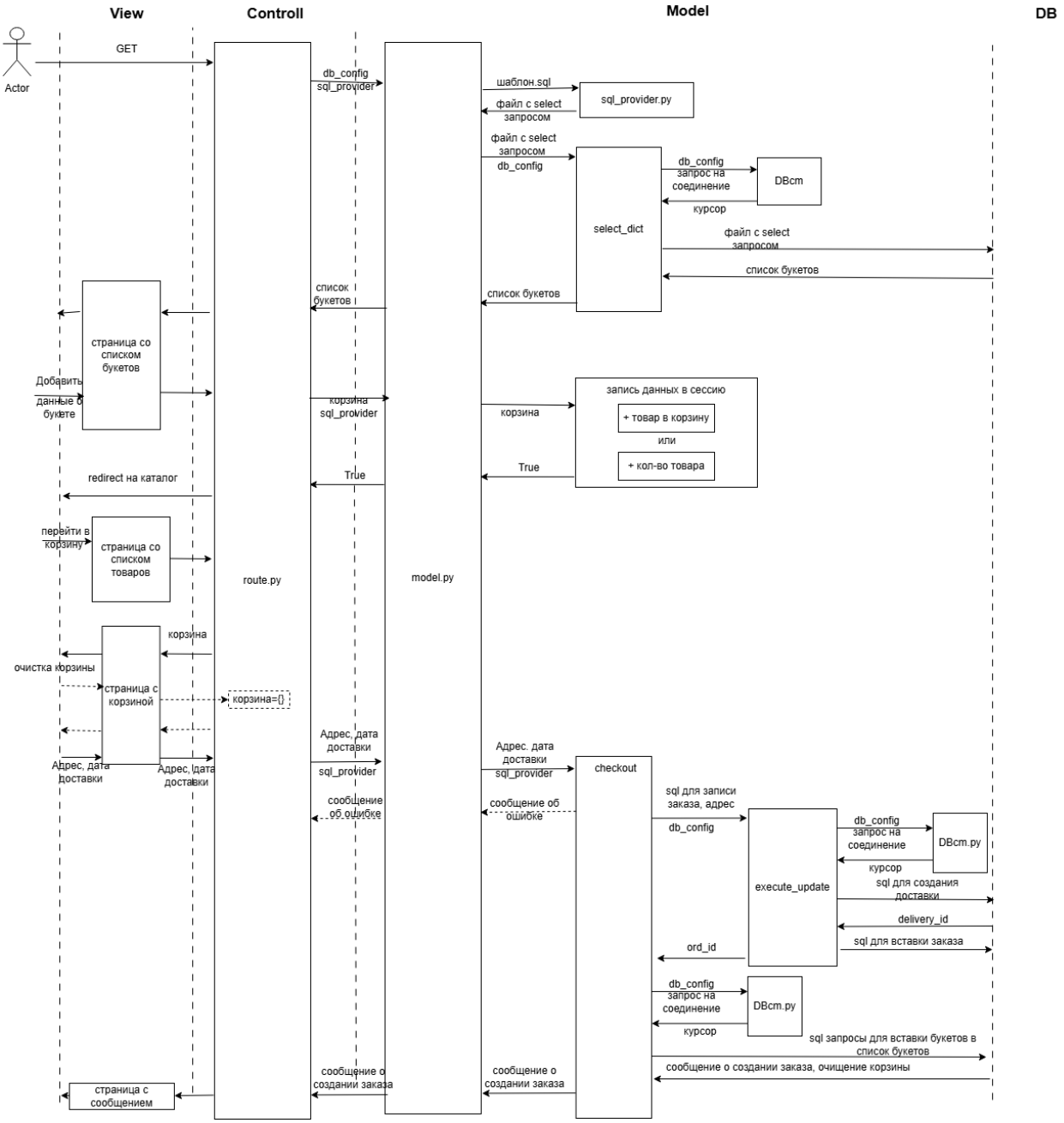
Предусловия: пользователь должен быть авторизован и зарегистрирован в ИС
Гарантия: *Пользователь создаст заказ с доставкой или просмотрит товары*
Минимальная гарантия: БД находится в согласованном состоянии и в случае неудачи сможет вернуться в главное меню

Сценарий использования

1) Пользователь запускает каталог
2) Пользователь получает список товаров, которые он может поместить в свою корзину
3) Пользователь нажимает кнопку добавить
4) Система заносит товар в корзину и возвращает его обратно к покупкам
5) Пользователь заходит в корзину
6) Система выдает шаблон с содержимым корзины и формой для оформления времени, места и даты доставки
7) Пользователь нажимает очистить корзину
8) Система очищает корзину
9) Пользователь выбирает дату, время и место доставки в форме
10) Пользователь нажимает оформить заказ
11) Система формирует заказ и отправляет ее в БД, после чего отправляет сообщение пользователю об успехе отправки заказа

Исключения:

9.а) Некорректные данные для доставки или нет даты заказа, времени или места заказа система требует ввести данные



Требования к шаблону каталога

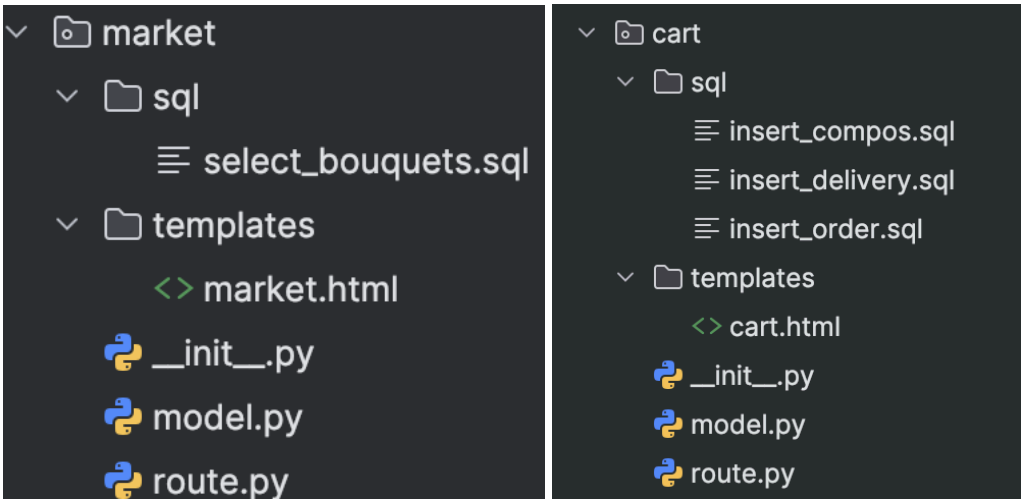
В центре страницы каталога цветов расположены заголовок **"Каталог цветов"** и кнопка **"Корзина"** с изображением и текстом, ведущая пользователя на страницу корзины. Далее расположена основная секция каталога, в которой представлены карточки всех доступных букетов. Каждая карточка содержит изображение букета (если изображение отсутствует, отображается placeholder с текстом "Фото в обработке"), название букета, цену, поле для ввода количества и кнопку **"Добавить**

в корзину". Внизу страницы присутствует кнопка **"Вернуться на главную страницу"**.

Требование к шаблону корзины

В центре страницы корзины расположены заголовок **"Корзина"**. Если корзина содержит товары, отображается кнопка **"Очистить корзину"**, позволяющая удалить все товары сразу. Также пользователю представлен список всех товаров в корзине, где отображаются название букета, количество, стоимость с учётом количества и кнопка **"Удалить из корзины"** для управления корзиной. Под списком находится форма оформления заказа с полями для ввода информации: дата доставки, время доставки и адрес доставки. Пользователь может завершить оформление заказа кнопкой **"Оформить заказ"**. Внизу страницы расположена кнопка **"Вернуться в магазин"**, которая позволяет пользователю легко вернуться к выбору товаров. Страница корзины также адаптивна и корректно отображается на всех устройствах.

структура каталога ***структура корзины***



Структура каталога

- **model.py**

отвечает за взаимодействие с базой данных. В нем реализована логика получения данных о букетах, а также добавления товаров в корзину. Если товар уже есть в корзине, его количество увеличивается, если нет — добавляется новая позиция.

- **route.py**

обрабатывает запросы от пользователей и связывает их с функциями из модели. Он формирует страницу со списком букетов, проверяет наличие соответствующих изображений, а также обрабатывает добавление товаров в корзину и уведомляет пользователя о результатах.

- **market.html**

- Шаблон для отображения каталога цветов. Показывает букет с изображением, названием, ценой и полем для выбора количества. Есть кнопки для добавления товара в корзину и навигация для возврата на главную страницу.

- **select_bouqets.sql**

- SQL-запрос для получения информации о всех букетах из базы данных.

Структура корзины

- **model.py**

файл содержит всю логику обработки данных: работу с сессией (инициализация корзины, очистка, удаление элементов), выполнение SQL-запросов для добавления доставки, заказа и связанных данных в базу, обработку ошибок, возникающих при взаимодействии с базой данных, а также отправку уведомлений пользователю (с помощью flash) о выполнении действий.

- **route.py**

файл регистрирует маршруты и отвечает за обработку GET и POST запросов для взаимодействия с корзиной, вызов функций из model.py для выполнения бизнес-логики, передачу данных в шаблоны для отображения содержимого корзины и перенаправление пользователя после выполнения действий (очистка корзины, удаление товара, оформление заказа).

- **cart.html**

- Шаблон для отображения корзины пользователя. Позволяет просматривать товары, удалять их по одному или очищать всю корзину, а также оформлять заказ с вводом данных доставки (дата, время, адрес).

- **insert_compos.sql**

- SQL-запрос для вставки данных о списке цветов в таблицу compos

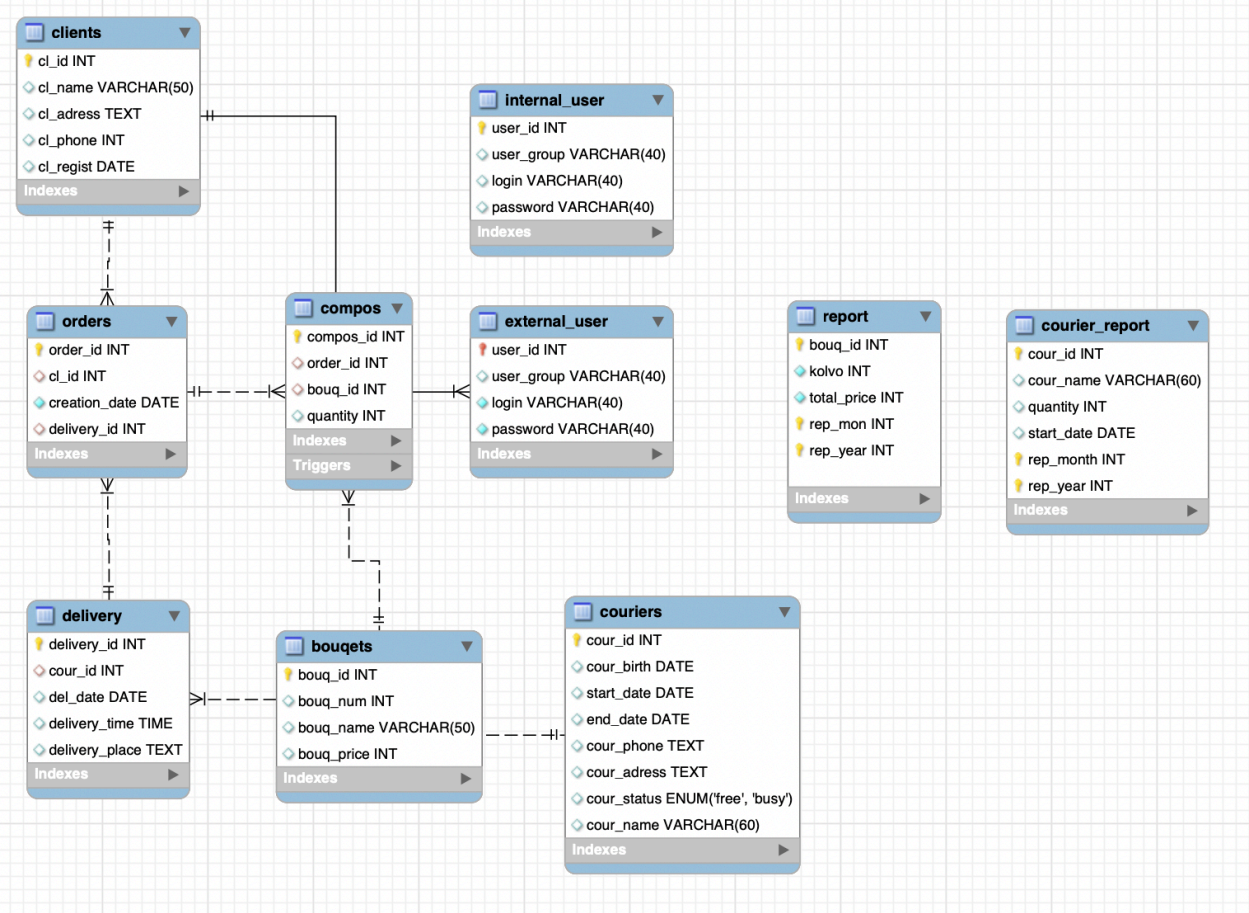
insert_delivery.sql

- SQL-запрос для вставки данных о доставке в таблицу delivery

- **insert_order.sql**

- SQL-запрос для вставки данных о заказе в таблицу orders

Схема базы данных



Заключение

В результате работы была создана информационная система для условного цветочного магазина, где была реализована возможность авторизации, просмотра различных данных, создание отчетов и заказов, путем изменения БД, работы с доставкой. В системе были добавлены уровни доступа для различных пользователей, а также микросервис для авторизации внешних пользователей.