|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **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»\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*Лабораторная работа № 1*

*По предмету: «Проектирование программного обеспечения»*

Студент: Юмаев Артур Русланович

Группа: ИУ7-65Б

Оглавление

[1. Use – Case диаграмма 3](#_Toc35990387)

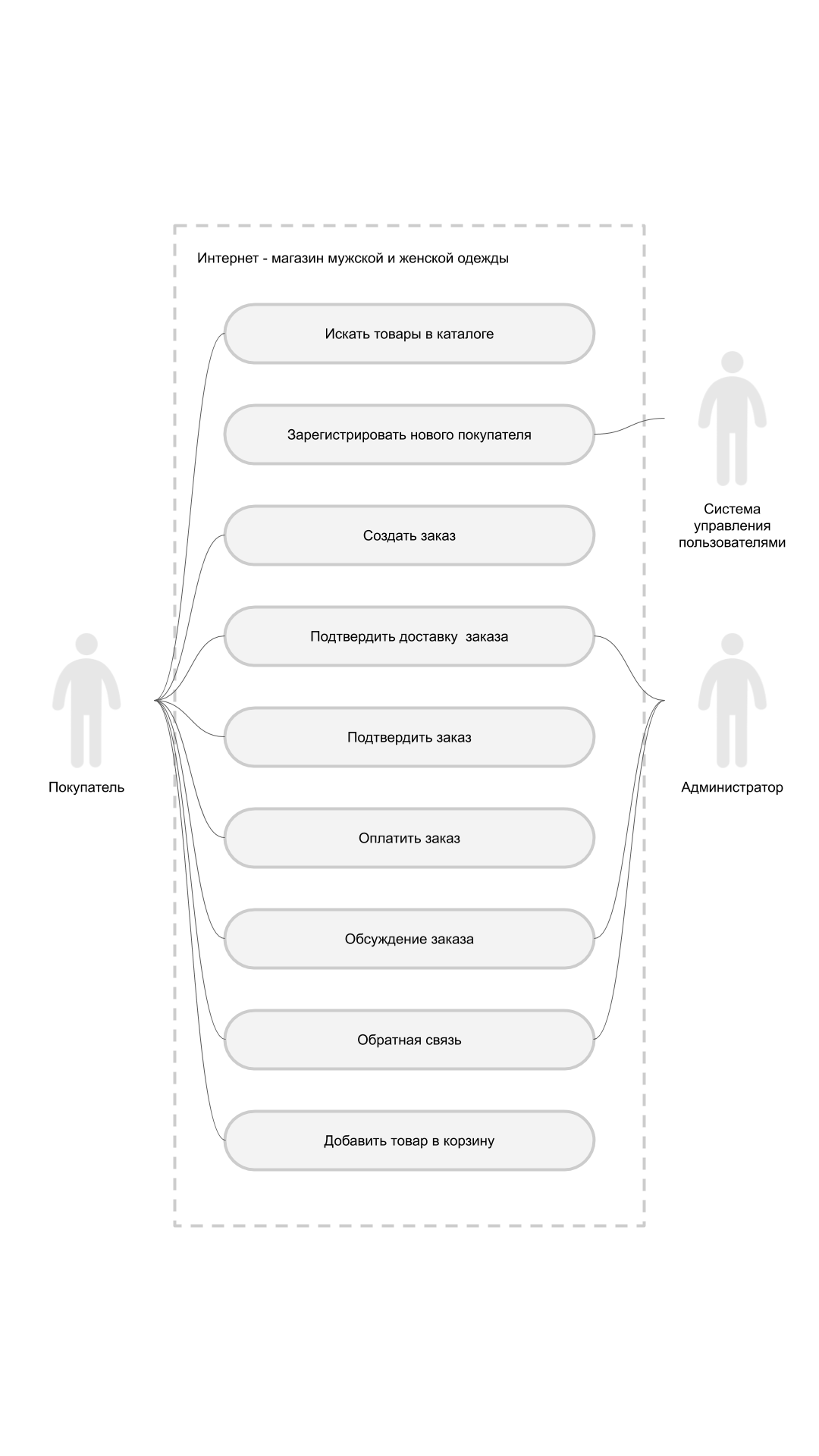
[2. ER – диаграмма сущностей 4](#_Toc35990388)

[3. Технологический стек 5](#_Toc35990389)

[4. UML диаграммы классов 6](#_Toc35990390)

[5. Программная реализация компонента доступа к данным 7](#_Toc35990391)

# 1. Use – Case диаграмма



# 2. ER – диаграмма сущностей

# 3. Технологический стек

Используемые технологии и подходы

Бэкенд

* Язык программирования Python
* Фреймворк Flask для быстрой реализации простых одностраничных веб приложений на модели MVC (Model – View – Controller)
* Apache HTTP-сервер – кроссплатформенный веб сервер для обработки соединений с юзер-агентом
* SQLite реляционная база данных
* Система контроля версий Git (GitHub)
* Docker — разработка, развертывание на сервере
* Microsoft Azure – облачный сервер c Ubuntu 16.04

Фронтенд

* Верстка макета с помощью HTML, CSS, JavaScript
* Twitter Bootstrap для быстрой и адаптивной разработки интерфейсов

# 4. UML диаграммы классов

# 5. Программная реализация компонента доступа к данным

В качестве программной реализации доступа к данным был выбран фреймворк sqlite3 версии 3.8.2 для Python 3.7.1. так как был опыт в использовании. Реализация доступа к базе данных указана на листингах 1-3.

Листинг 1. Подключение библиотеки sqlite3 и указание пути базы данных

|  |
| --- |
| **import** sqlite3  DEV\_MODE = 1  DB\_PATH = './database/catalog.db' **if** DEV\_MODE **else** \  '/var/www/webApp/webApp/database/catalog.db' |

Листинг 2. Функция добавления товара в пользовательскую корзину

|  |
| --- |
| **def** insert\_into\_cart(userid, vendor, size):  conn = sqlite3.connect(DB\_PATH)  c = conn.cursor()  sql\_query = 'insert into cart (userid, vendor, size) values '  sql\_query += '(\'{}\',{},{});'.format(userid, vendor, size)  c.execute(sql\_query)  conn.commit()  sql\_query = "select count(userid) from cart where userid = \  '{}';".format(userid))  goods\_amount = c.execute(sql\_query)  goods\_amount = [i **for** i **in** goods\_amount][0][0]  conn.close()  **return** goods\_amount |

Листинг 3. Функция получения количества товаров в корзине

|  |
| --- |
| **def** get\_items\_amount(userid):  conn = sqlite3.connect(DB\_PATH)  c = conn.cursor()  goods\_amount = c.execute("select count(userid) from cart \  where userid = '{}';".format(userid))  goods\_amount = [i **for** i **in** goods\_amount][0][0]  conn.close()  **return** goods\_amount |

На листинге 4 показана функция получения каталога товаров. В качестве аргумента она принимает объект request, который получает сервер от веб-клиента. В объекте request содержатся url аргумента, которые представляют собой словарь ключ - значение.

Листинг 4. Функция получения каталога товаров

|  |
| --- |
| **def** get\_data(request):  url\_args = list(request.args.keys())  **if** **all**(i **in** url\_args **for** i **in** ['sort', 'gender', 'cats']):  sort\_by = request.args.get('sort')  gender = request.args.get('gender')  categories = request.args.getlist('cats')  **else**:  sort\_by = 'Sort by'  gender = 'forall'  categories = 'None'  data = make\_sql\_query(gender, sort\_by, categories)  **return** data |

Листинг 5. Функция получения корзины пользователя

|  |
| --- |
| **def** select\_cart(userid):  sql\_query = """  *select \*, count(\*) as amount from (*  *select g.description,*  *g.category,*  *g.gender,*  *c.size,*  *g.color,*  *g.price \* (1 - g.discount) as price,*  *c.vendor*  *from cart c*  *join goods g*  *on c.vendor = g.vendor*  *where userid = '{0}'*  *) as items*  *group by vendor, size*  *having count(\*) >= 1*  """.format(userid)  conn = sqlite3.connect(DB\_PATH)  c = conn.cursor()  data = [i **for** i **in** c.execute(sql\_query)]  conn.close()  **return** data |