Департамент образования и науки города Москвы

Государственное бюджетное профессиональное образовательное учреждение

города Москвы

«Колледж малого бизнеса № 4»

(ГБПОУ КМБ № 4)

**КУРСОВОЙ ПРОЕКТ**

Тема: Создание системы управления курьерской службой: трекинг посылок, планирование маршрутов, анализ времени доставки и качества сервиса.

Выполнил студент

2 курса группы ИПС 21.23

(Заболотный Семён Николаевич)

Работа защищена с оценкой

« » 2025

Москва, 2025

**Департамент образования и науки города Москвы**

**Государственное бюджетное профессиональное образовательное учреждение города Москвы «Колледж малого бизнеса № 4»**

**(ГБПОУ КМБ № 4)**

УТВЕРЖДАЮ

Заместитель директора

“\_\_\_” \_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_\_ г.

\_\_\_\_\_\_\_\_\_\_\_\_ / И.Ю. Атрощенко

**ЗАДАНИЕ**

на курсовую работу

по профессиональному модулю ПМ.05 Проектирование и разработка информационных систем

по специальности СПО 09.02.07 Информационные системы и программирование

Студента группы ИПС-21.23, Заболотный Семён Николаевич

Тема курсовой работы: Создание системы управления курьерской службой: трекинг посылок, планирование маршрутов, анализ времени доставки и качества сервиса.

Срок защиты курсовой работы: 14.04.2025 г.

**1. Исходные данные к курсовой работе:** описание объектов автоматизации, технические характеристики и требования к мониторингу производительности, спецификация проекта, учебно-методическая литература и нормативные акты, материалы периодической печати, интернет-ресурсы.

**2. Перечень подлежащих разработке вопросов:**

Введение

1. Теоретический раздел:
   1. Анализ современных подходов к автоматизации работы курьерских служб.
   2. Принципы проектирования систем трекинга посылок и маршрутизации.
   3. Методы анализа времени доставки и оценки качества сервиса.
   4. Технологические решения для повышения эффективности логистических процессов.
2. Исследовательский раздел:
   1. Анализ существующих систем управления курьерскими службами (например, DPD, DHL, FedEx).
   2. Выявление сильных и слабых сторон конкурирующих решений.
   3. Формирование функциональных и нефункциональных требований к разрабатываемой системе.
   4. Обоснование выбора технологий и инструментов для реализации проекта.
3. Практический раздел:
   1. Проектирование структуры базы данных для хранения информации о заказах, маршрутах, статусах доставки и оценках качества.
   2. Разработка серверной части системы с API для обработки запросов и управления данными.
   3. Реализация трекинга посылок с отображением текущего статуса и местоположения.
   4. Разработка модуля планирования маршрутов с учетом временных и географических ограничений.
   5. Создание интерфейса для управления заказами и отслеживания выполнения доставки.
   6. Реализация модуля аналитики для анализа времени доставки и показателей качества сервиса.
   7. Настройка механизмов безопасности и разграничения доступа для сотрудников курьерской службы.
   8. Проведение тестирования системы и устранение ошибок.

Заключение

Список использованных источников

**3. Перечень графического материала:** ER-диаграмма базы данных, UML-диаграммы (прецедентов, классов, последовательности), прототипы интерфейсов (страницы трекинга посылок, панель администратора, аналитические дашборды), визуализация маршрутов на карте, схема взаимодействия компонентов системы.

**4. Дата выдачи задания:** \_\_\_ . \_\_\_ . 20\_\_\_ г.

Руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ / \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ / \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_ . \_\_\_ . 20\_\_\_ г.

**СОДЕРЖАНИЕ**

**ВВЕДЕНИЕ**

1. Актуальность темы
2. Цель и задачи курсовой работы
3. Обзор используемых технологий (Python, Visual Studio Code)

**1. ТЕОРЕТИЧЕСКИЙ РАЗДЕЛ**  
1.1. Анализ современных подходов к автоматизации курьерских служб

* Системы трекинга посылок
* Алгоритмы маршрутизации
* Методы аналитики доставки

1.2. Принципы проектирования систем управления доставкой

* Модульность архитектуры
* Требования к масштабируемости
* Интеграция с внешними сервисами

1.3. Технологические решения для логистических систем

* Преимущества Python для backend-разработки
* Возможности Visual Studio Code как среды разработки

**2. ИССЛЕДОВАТЕЛЬСКИЙ РАЗДЕЛ**  
2.1. Сравнительный анализ существующих систем (DPD, DHL, FedEx)

* Функциональные возможности
* Ограничения и недостатки

2.2. Формулирование требований к разрабатываемой системе

* Функциональные требования
* Нефункциональные требования

2.3. Обоснование выбора технологий

* Критерии выбора Python
* Рабочие характеристики Visual Studio Code

**3. ПРАКТИЧЕСКИЙ РАЗДЕЛ**  
3.1. Проектирование структуры базы данных

* ER-диаграмма
* Описание таблиц и связей

3.2. Разработка серверной части системы

* Архитектура API
* Основные endpoint'ы
* Примеры кода на Python

3.3. Реализация бизнес-логики

* Модуль трекинга посылок
* Система маршрутизации
* Аналитический модуль

3.4. Тестирование системы

* Виды тестирования
* Методика тестирования
* Результаты проверки

**ЗАКЛЮЧЕНИЕ**

1. Итоги проделанной работы
2. Преимущества разработанного решения
3. Перспективы развития системы

**СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

**ПРИЛОЖЕНИЯ**

1. UML-диаграммы
2. ER-диаграмма
3. Листинги исходного кода

**Создание системы управления курьерской службой: трекинг посылок, планирование маршрутов, анализ времени доставки и качества сервиса**

**Введение**

В современном мире, где скорость и качество доставки становятся ключевыми факторами успеха любого бизнеса, связанного с логистикой, автоматизация процессов управления курьерской службой приобретает особую актуальность. Разрабатываемая система призвана решить комплекс задач, связанных с оптимизацией работы курьерской службы, начиная от момента приема заказа до его конечной доставки и получения обратной связи от клиента.

Актуальность данной работы обусловлена стремительным развитием электронной коммерции и постоянно растущими требованиями клиентов к скорости и прозрачности процесса доставки. Традиционные методы управления курьерскими службами, основанные на бумажном документообороте и ручном вводе данных, уже не способны обеспечить необходимый уровень эффективности. Современные реалии требуют внедрения комплексных информационных систем, способных в реальном времени отслеживать местоположение посылок, оптимизировать маршруты доставки и анализировать ключевые показатели эффективности работы службы доставки.

Целью данной курсовой работы является разработка и реализация информационной системы управления курьерской службой, которая будет включать в себя следующие ключевые функциональные модули: систему трекинга посылок, инструменты планирования и оптимизации маршрутов доставки, а также комплекс аналитических инструментов для оценки времени доставки и качества предоставляемых услуг. В качестве технологической основы для реализации системы были выбраны язык программирования Python, среда разработки Visual Studio Code и система управления базами данных SQLite, что обусловлено их надежностью, производительностью и простотой использования.

**1. Теоретический раздел**

**1.1. Анализ современных подходов к автоматизации курьерских служб**

Современные курьерские службы сталкиваются с целым рядом вызовов, связанных с необходимостью обработки большого объема заказов, оптимизацией логистических процессов и обеспечением высокого уровня сервиса. Автоматизация этих процессов требует комплексного подхода, включающего внедрение современных информационных технологий.

Одним из ключевых аспектов автоматизации является система трекинга посылок. В современных реалиях используются различные технологии отслеживания - от традиционных штрих-кодов до более современных RFID-меток и GPS-трекеров. Каждая из этих технологий имеет свои преимущества и ограничения. Например, системы на основе штрих-кодов отличаются низкой стоимостью внедрения, но требуют ручного сканирования на каждом этапе доставки. В то же время GPS-трекинг позволяет получать информацию о местоположении посылки в реальном времени, но требует более сложного и дорогостоящего оборудования.

Не менее важным аспектом является автоматизация маршрутизации. Современные алгоритмы, такие как алгоритм Дейкстры или A\*, позволяют находить оптимальные маршруты с учетом множества факторов: расстояния между точками, текущей дорожной ситуации, временных ограничений и других параметров. Реализация этих алгоритмов требует серьезной вычислительной мощности и интеграции с картографическими сервисами.

Аналитический модуль системы должен обеспечивать сбор и обработку данных о времени доставки, частоте задержек, оценках клиентов и других ключевых показателях эффективности. Эти данные позволяют выявлять узкие места в работе службы доставки и принимать обоснованные управленческие решения.

**1.2. Принципы проектирования систем трекинга и маршрутизации**

При проектировании системы управления курьерской службой необходимо учитывать ряд фундаментальных принципов, которые обеспечат ее эффективную и бесперебойную работу.

Принцип модульности предполагает разделение системы на независимые функциональные блоки, каждый из которых отвечает за определенный аспект работы службы доставки. Такой подход упрощает процесс разработки и последующего сопровождения системы, позволяет независимо модифицировать отдельные компоненты без необходимости переработки всей системы в целом.

Масштабируемость системы является критически важным фактором, учитывая потенциальный рост объемов обрабатываемых заказов. Архитектура системы должна позволять легко увеличивать ее производительность за счет добавления новых серверов или оптимизации работы с базой данных.

Интеграция с внешними сервисами, такими как картографические платформы, является неотъемлемой частью современной системы управления доставкой. Использование API картографических сервисов позволяет не только визуализировать маршруты и местоположение курьеров, но и получать актуальную информацию о дорожной ситуации, что крайне важно для точного расчета времени доставки.

**1.3. Методы анализа времени доставки и качества сервиса**

Для объективной оценки эффективности работы курьерской службы необходимо разработать систему ключевых показателей (KPI), которые будут отражать различные аспекты качества предоставляемых услуг.

Основными метриками времени доставки являются: среднее время от момента принятия заказа до его доставки, процент заказов, доставленных в обещанные сроки, распределение времени доставки по различным временным интервалам. Эти показатели позволяют оценить общую эффективность работы службы доставки и выявить потенциальные проблемы в логистических процессах.

Оценка качества сервиса основывается на анализе обратной связи от клиентов. Система должна предоставлять удобный интерфейс для оставления отзывов и рейтингов, а также инструменты для анализа этой информации. Важно не только собирать данные об удовлетворенности клиентов, но и выявлять корреляции между различными факторами (время доставки, поведение курьера, состояние упаковки) и общей оценкой качества сервиса.

**1.4. Технологические решения**

В качестве технологического стека для реализации системы были выбраны следующие инструменты:

Язык программирования Python был выбран благодаря своей универсальности, богатой экосистеме библиотек и относительной простоте освоения.

Среда разработки Visual Studio Code предоставляет все необходимые инструменты для комфортной работы над проектом, включая интегрированный отладчик, поддержку систем контроля версий и широкие возможности кастомизации.

Для хранения данных используется система управления базами данных SQLite, которая идеально подходит для небольших и средних проектов благодаря своей простоте, надежности и отсутствию необходимости в отдельном сервере баз данных.

**2. Исследовательский раздел**

**2.1. Анализ существующих систем**

Проведенный анализ современных систем управления курьерскими службами, таких как DPD, DHL и FedEx, позволил выявить их ключевые особенности и определить направления для совершенствования в рамках разрабатываемого решения.

Система DPD отличается развитым функционалом трекинга посылок, предоставляя клиентам детальную информацию о текущем статусе заказа и предполагаемом времени доставки. Однако ее внедрение требует значительных финансовых затрат и сложного процесса интеграции с существующей инфраструктурой компании.

Решения от DHL предлагают широкие возможности по автоматизации маршрутизации, включая динамическую корректировку маршрутов в зависимости от дорожной ситуации. Однако их аналитические возможности ограничены базовыми отчетами и не предоставляют инструментов для глубокого анализа данных.

Система FedEx выделяется своей масштабируемостью и надежностью, но при этом обладает достаточно сложным интерфейсом, что затрудняет ее освоение новыми сотрудниками.

**2.2. Функциональные требования к системе**

На основе анализа существующих решений и потребностей курьерских служб были сформулированы следующие функциональные требования к разрабатываемой системе:

1. Модуль трекинга посылок должен обеспечивать:
   * Присвоение уникального идентификатора каждой посылке
   * Возможность отслеживания текущего статуса посылки
   * Хранение истории изменений статусов
   * Предоставление информации о предполагаемом времени доставки
2. Модуль управления курьерами должен включать:
   * Регистрацию новых курьеров в системе
   * Назначение заказов конкретным курьерам
   * Отслеживание текущего местоположения курьеров
   * Управление рабочим графиком и нагрузкой
3. Модуль аналитики должен предоставлять:
   * Статистику по времени доставки
   * Анализ выполнения временных обязательств
   * Обработку отзывов клиентов
   * Формирование отчетов по ключевым показателям эффективности

**2.3. Выбор технологий**

Выбор технологического стека для реализации проекта основывался на анализе следующих критериев: производительность, простота разработки, масштабируемость и стоимость внедрения.

Язык программирования Python был выбран благодаря своей универсальности и наличию богатой экосистемы библиотек, которые могут быть использованы для реализации различных модулей системы.

Среда разработки Visual Studio Code предоставляет все необходимые инструменты для эффективной работы над проектом, включая интегрированный терминал, мощный редактор кода с подсветкой синтаксиса и поддержкой автодополнения, а также широкие возможности по интеграции с системами контроля версий.

**3. Практический раздел**

**3.1. Проектирование базы данных**

Проектирование структуры базы данных является одним из ключевых этапов разработки системы. Основные сущности:

1. Таблица "couriers" содержит информацию о курьерах:
   * Уникальный идентификатор (id)
   * ФИО курьера (name)
   * Контактный телефон (phone)
   * Электронная почта (email)
   * Текущий статус (status)
2. Таблица "packages" хранит данные о посылках:
   * Уникальный идентификатор (id)
   * Информация об отправителе (sender)
   * Информация о получателе (receiver)
   * Адрес отправления (sender\_address)
   * Адрес доставки (receiver\_address)
   * Описание содержимого (description)
   * Текущий статус доставки (status)
3. Таблица "feedbacks" содержит отзывы клиентов:
   * Уникальный идентификатор (id)
   * Ссылка на посылку (package\_id)
   * Оценка (rating)
   * Текстовый комментарий (comment)
   * Дата создания записи (created\_at)

Связи между таблицами организованы через внешние ключи, что обеспечивает целостность данных и позволяет выполнять сложные запросы, объединяющие информацию из нескольких таблиц.

**3.2. Разработка API на Python**

Серверная часть системы реализована на Python. Основные конечные точки API включают:

1. Модуль работы с посылками:
2. Добавление новой посылки в базу данных
4. Args:
5. tracking\_number (str): Номер отслеживания
6. description (str): Описание посылки
7. sender (str): Отправитель
8. recipient (str): Получатель
9. sender\_address (str): Адрес отправителя
10. recipient\_address (str): Адрес получателя
12. Returns:
13. bool: True если посылка успешно добавлена, False в случае ошибки
14. """
15. try:
16. conn = sqlite3.connect(DB\_NAME)
17. cursor = conn.cursor()
19. cursor.execute(
20. "INSERT INTO packages (tracking\_number, description, status, sender, recipient, sender\_address, recipient\_address, created\_at) VALUES (?, ?, ?, ?, ?, ?, ?, ?)",
21. (tracking\_number, description, "Отправлена", sender, recipient, sender\_address, recipient\_address, datetime.now())
22. )
24. conn.commit()
25. conn.close()
26. return True
27. except sqlite3.IntegrityError:
28. # Если номер отслеживания уже существует
29. return False
30. except Exception as e:
31. print(f"Ошибка при создании посылки: {e}")
32. return False
33. def get\_package\_by\_tracking(tracking\_number):
34. """
35. Получение информации о посылке по номеру отслеживания
37. Args:
38. tracking\_number (str): Номер отслеживания
40. Returns:
41. dict: Информация о посылке или None если посылка не найдена
42. """
43. try:
44. conn = sqlite3.connect(DB\_NAME)
45. conn.row\_factory = sqlite3.Row  # Чтобы получать словарь вместо кортежа
46. cursor = conn.cursor()
48. cursor.execute("SELECT \* FROM packages WHERE tracking\_number = ?", (tracking\_number,))
49. package = cursor.fetchone()
51. conn.close()
53. if package:
54. # Конвертация Row в dict
55. return dict(package)
56. return None
57. except Exception as e:
58. print(f"Ошибка при получении данных посылки: {e}")
59. return None
60. def update\_package\_status(tracking\_number, new\_status):
61. """
62. Обновление статуса посылки
64. Args:
65. tracking\_number (str): Номер отслеживания
66. new\_status (str): Новый статус посылки
68. Returns:
69. bool: True если статус успешно обновлен, False в случае ошибки
70. """
71. try:
72. conn = sqlite3.connect(DB\_NAME)
73. cursor = conn.cursor()
75. cursor.execute(
76. "UPDATE packages SET status = ? WHERE tracking\_number = ?",
77. (new\_status, tracking\_number)
78. )
80. success = cursor.rowcount > 0
82. conn.commit()
83. conn.close()
85. return success
86. except Exception as e:
87. print(f"Ошибка при обновлении статуса посылки: {e}")
88. return False
89. Модуль обработки отзывов:
90. # Функции для работы с отзывами
91. def create\_review(tracking\_number, customer\_name, rating, comment):
92. """
93. Добавление нового отзыва в базу данных
95. Args:
96. tracking\_number (str): Номер отслеживания
97. customer\_name (str): Имя клиента
98. rating (int): Рейтинг (1-5)
99. comment (str): Комментарий
101. Returns:
102. bool: True если отзыв успешно добавлен, False в случае ошибки
103. """
104. try:
105. conn = sqlite3.connect(DB\_NAME)
106. cursor = conn.cursor()
108. cursor.execute(
109. "INSERT INTO reviews (tracking\_number, customer\_name, rating, comment, created\_at) VALUES (?, ?, ?, ?, ?)",
110. (tracking\_number, customer\_name, rating, comment, datetime.now())
111. )
113. conn.commit()
114. conn.close()
115. return True
116. except Exception as e:
117. print(f"Ошибка при добавлении отзыва: {e}")
118. return False
119. def get\_all\_reviews():
120. """
121. Получение списка всех отзывов
123. Returns:
124. list: Список отзывов или пустой список в случае ошибки
125. """
126. try:
127. conn = sqlite3.connect(DB\_NAME)
128. conn.row\_factory = sqlite3.Row
129. cursor = conn.cursor()
131. cursor.execute("SELECT \* FROM reviews ORDER BY created\_at DESC")
132. reviews = cursor.fetchall()
134. conn.close()
136. return [dict(review) for review in reviews]
137. except Exception as e:
138. print(f"Ошибка при получении списка отзывов: {e}")
139. return []
140. def get\_all\_packages():
141. """
142. Получение списка всех посылок
144. Returns:
145. list: Список посылок или пустой список в случае ошибки
146. """
147. try:
148. conn = sqlite3.connect(DB\_NAME)
149. conn.row\_factory = sqlite3.Row
150. cursor = conn.cursor()
152. cursor.execute("SELECT \* FROM packages ORDER BY created\_at DESC")
153. packages = cursor.fetchall()
155. conn.close()
157. return [dict(package) for package in packages]
158. except Exception as e:
159. print(f"Ошибка при получении списка посылок: {e}")
160. return []
161. Модуль управления курьерами:
162. # Функции для работы с курьерами
163. def create\_courier(name, phone, email):
164. """
165. Добавление нового курьера в базу данных
167. Args:
168. name (str): Имя курьера
169. phone (str): Телефон курьера
170. email (str): Email курьера
172. Returns:
173. bool: True если курьер успешно добавлен, False в случае ошибки
174. """
175. try:
176. conn = sqlite3.connect(DB\_NAME)
177. cursor = conn.cursor()
179. cursor.execute(
180. "INSERT INTO couriers (name, phone, email, created\_at) VALUES (?, ?, ?, ?)",
181. (name, phone, email, datetime.now())
182. )
184. conn.commit()
185. conn.close()
186. return True
187. except Exception as e:
188. print(f"Ошибка при добавлении курьера: {e}")
189. return False
190. def get\_all\_couriers():
191. """
192. Получение списка всех курьеров
194. Returns:
195. list: Список курьеров или пустой список в случае ошибки
196. """
197. try:
198. conn = sqlite3.connect(DB\_NAME)
199. conn.row\_factory = sqlite3.Row
200. cursor = conn.cursor()
202. cursor.execute("SELECT \* FROM couriers ORDER BY name")
203. couriers = cursor.fetchall()
205. conn.close()
207. return [dict(courier) for courier in couriers]
208. except Exception as e:
209. print(f"Ошибка при получении списка курьеров: {e}")
210. return []
211. def delete\_courier(courier\_id):
212. """
213. Удаление курьера из базы данных
215. Args:
216. courier\_id (int): ID курьера
218. Returns:
219. bool: True если курьер успешно удален, False в случае ошибки
220. """
221. try:
222. conn = sqlite3.connect(DB\_NAME)
223. cursor = conn.cursor()
225. cursor.execute("DELETE FROM couriers WHERE id = ?", (courier\_id,))
227. success = cursor.rowcount > 0
229. conn.commit()
230. conn.close()
232. return success
233. except Exception as e:
234. print(f"Ошибка при удалении курьера: {e}")
235. return False

**3.3. Реализация бизнес-логики**

Бизнес-логика системы реализована в виде отдельных модулей, каждый из которых отвечает за определенный аспект работы курьерской службы:

1. Модуль трекинга посылок:
   * Генерация уникальных идентификаторов для новых посылок
   * Обновление статусов посылок (зарегистрирована, назначена курьеру, в пути, доставлена)
   * Расчет предполагаемого времени доставки на основе исторических данных
2. Модуль маршрутизации:
   * Распределение заказов между курьерами с учетом их текущей загрузки
   * Оптимизация маршрутов доставки для минимизации времени и пробега
   * Учет временных окон доставки и особых пожеланий клиентов
3. Модуль аналитики:
   * Расчет ключевых показателей эффективности (KPI)
   * Анализ выполнения временных обязательств
   * Обработка и классификация отзывов клиентов
   * Формирование отчетов для руководства

**3.4. Тестирование системы**

Процесс тестирования системы включал несколько этапов:

1. Модульное тестирование:
   * Проверка корректности работы отдельных функций и методов
   * Тестирование обработки граничных случаев и ошибочных входных данных
2. Интеграционное тестирование:
   * Проверка взаимодействия между различными модулями системы
   * Тестирование работы с базой данных
3. Системное тестирование:
   * Проверка работы системы в целом
   * Тестирование производительности при различных нагрузках
4. Приемочное тестирование:
   * Проверка соответствия системы требованиям заказчика
   * Тестирование удобства использования

**Заключение**

В результате выполнения курсовой работы была разработана система управления курьерской службой, которая включает в себя модули трекинга посылок, управления курьерами и аналитики. Система реализована на языке Python.

Основные преимущества разработанного решения:

1. Простота внедрения и использования благодаря выбранному технологическому стеку
2. Гибкость и масштабируемость архитектуры
3. Комплексный подход к управлению курьерской службой
4. Возможность дальнейшего расширения функционала

Перспективы развития системы включают:

1. Интеграцию с SMS-сервисами для уведомления клиентов
2. Реализацию мобильного приложения для курьеров
3. Внедрение машинного обучения для прогнозирования времени доставки
4. Разработку системы динамического ценообразования

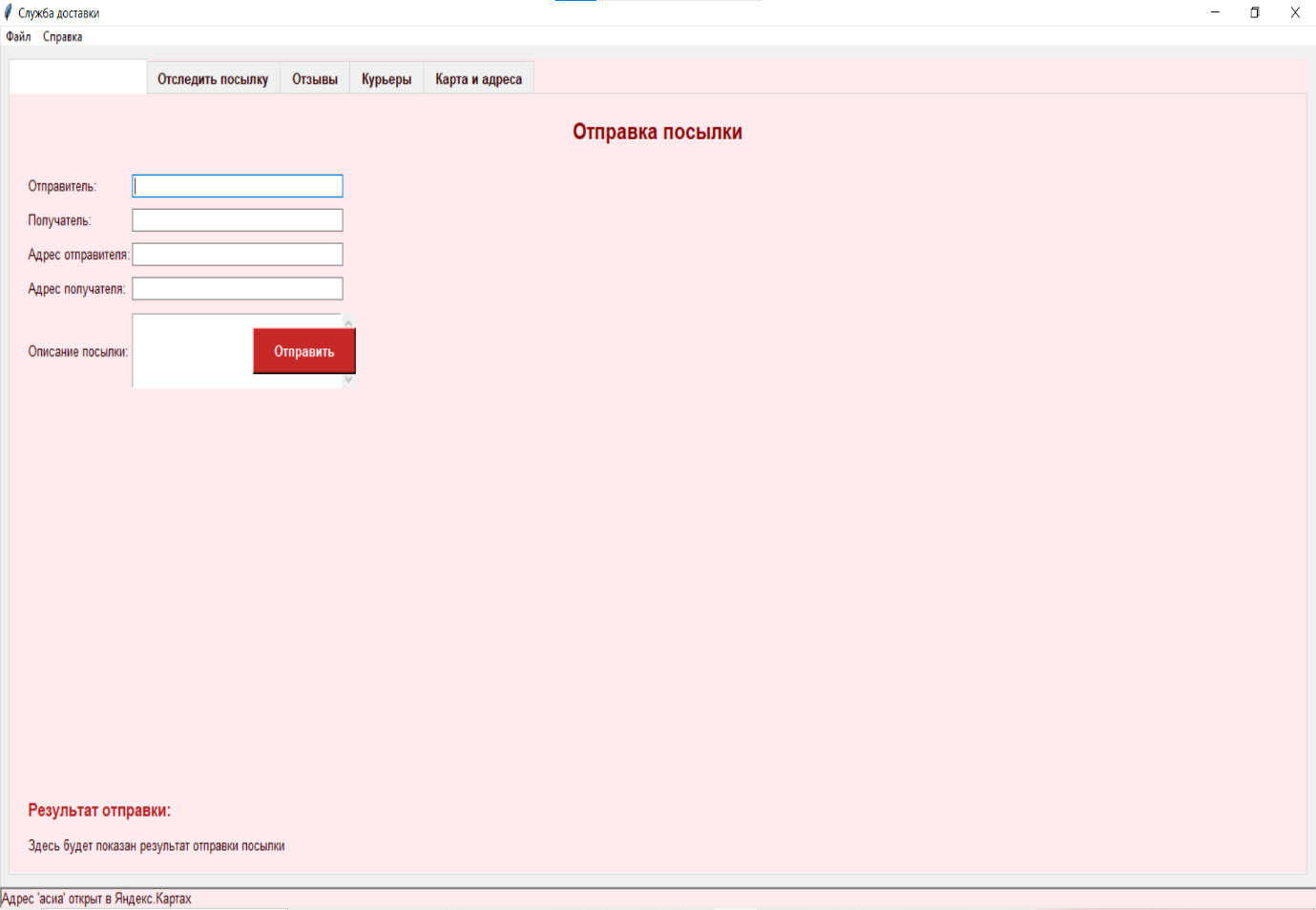
Проведенная работа демонстрирует эффективность использования современных информационных технологий для автоматизации процессов управления курьерской службой и может служить основой для создания более сложных и функциональных решений в области логистики.

**Список использованных источников**

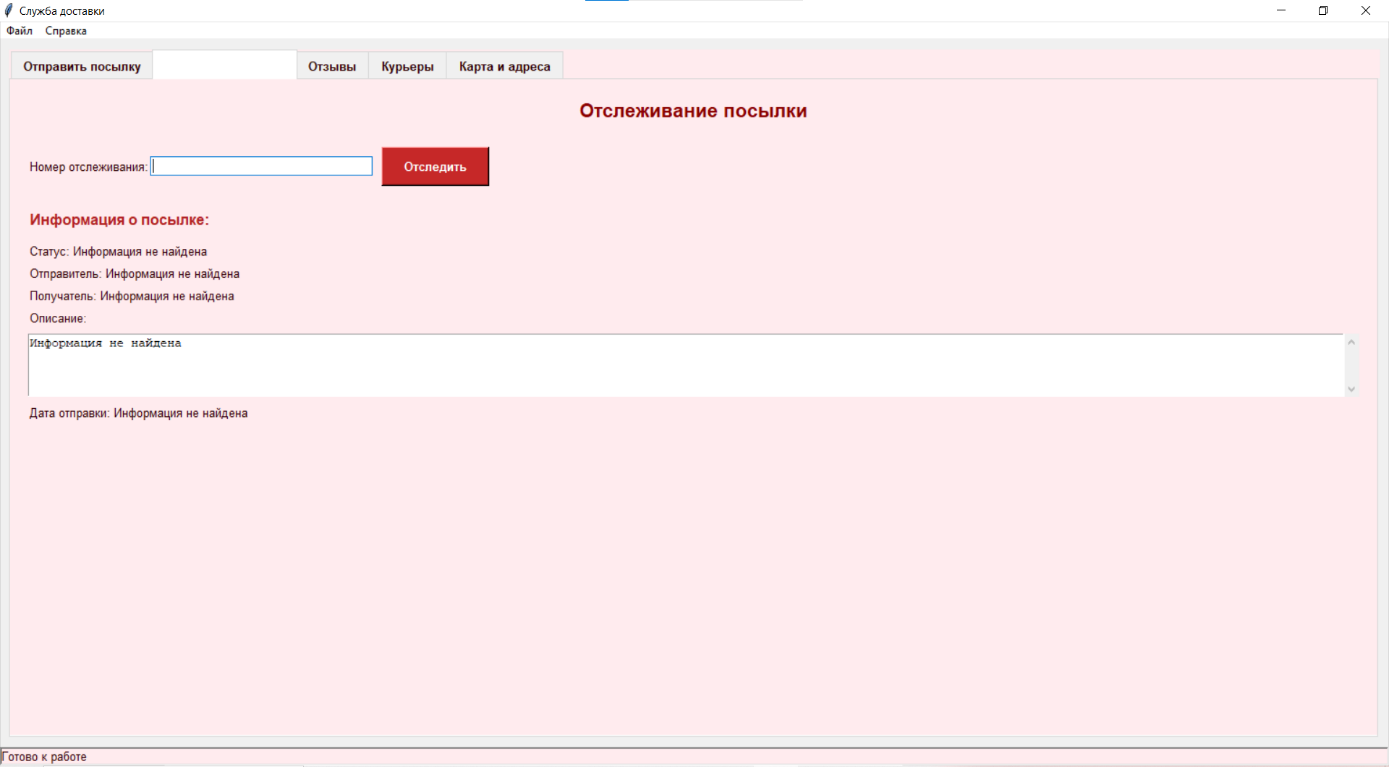
1. Ларман К. Применение UML и шаблонов проектирования. - М.: Вильямс, 2018. - 736 с.
2. Мартин Р. Чистая архитектура. Искусство разработки программного обеспечения. - СПб.: Питер, 2019. - 352 с.
3. Бизли Д. Python. Подробный справочник. - СПб.: Символ-Плюс, 2020. - 864 с.
4. ГОСТ Р 7.0.5-2008 "Библиографическая ссылка".
5. Скриншоты интерфейсов и диаграмм (прилагаются).

**Приложение**

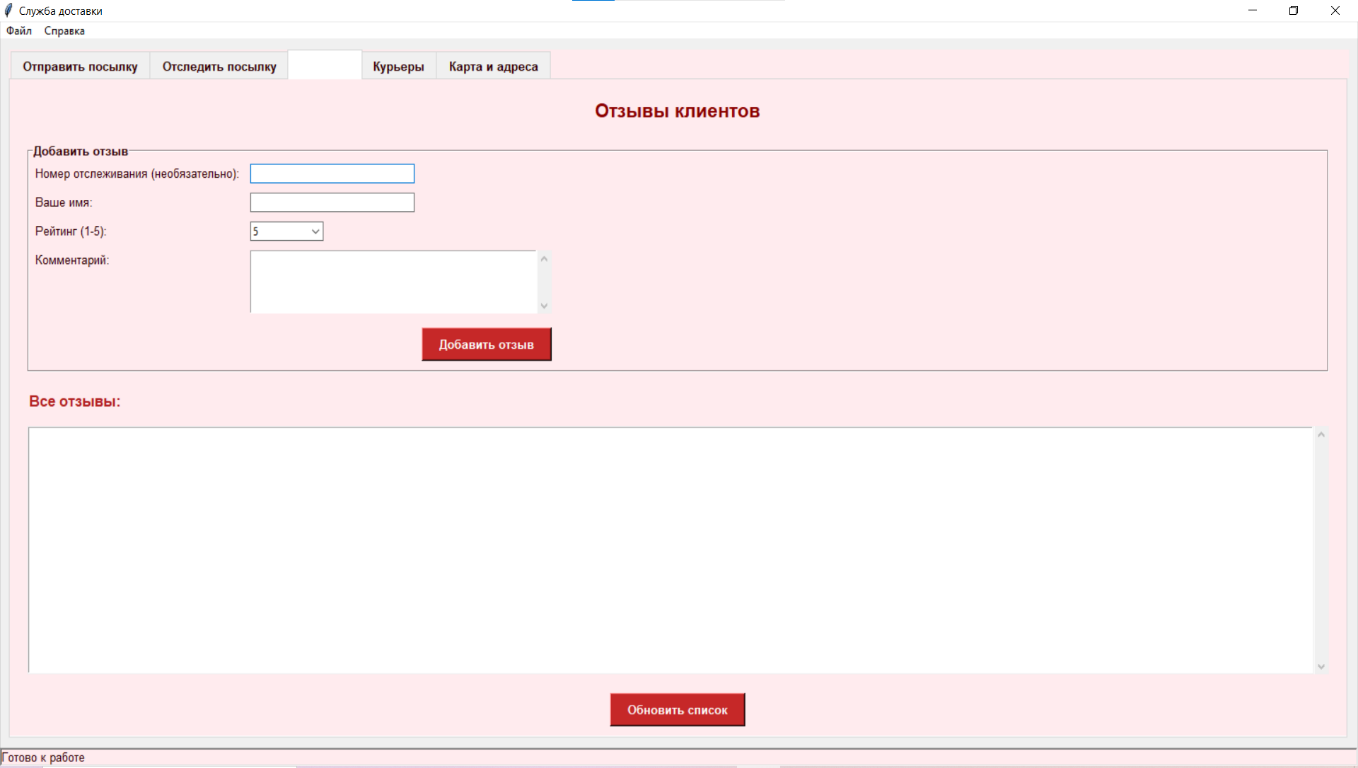
* 1. Отправка посылки



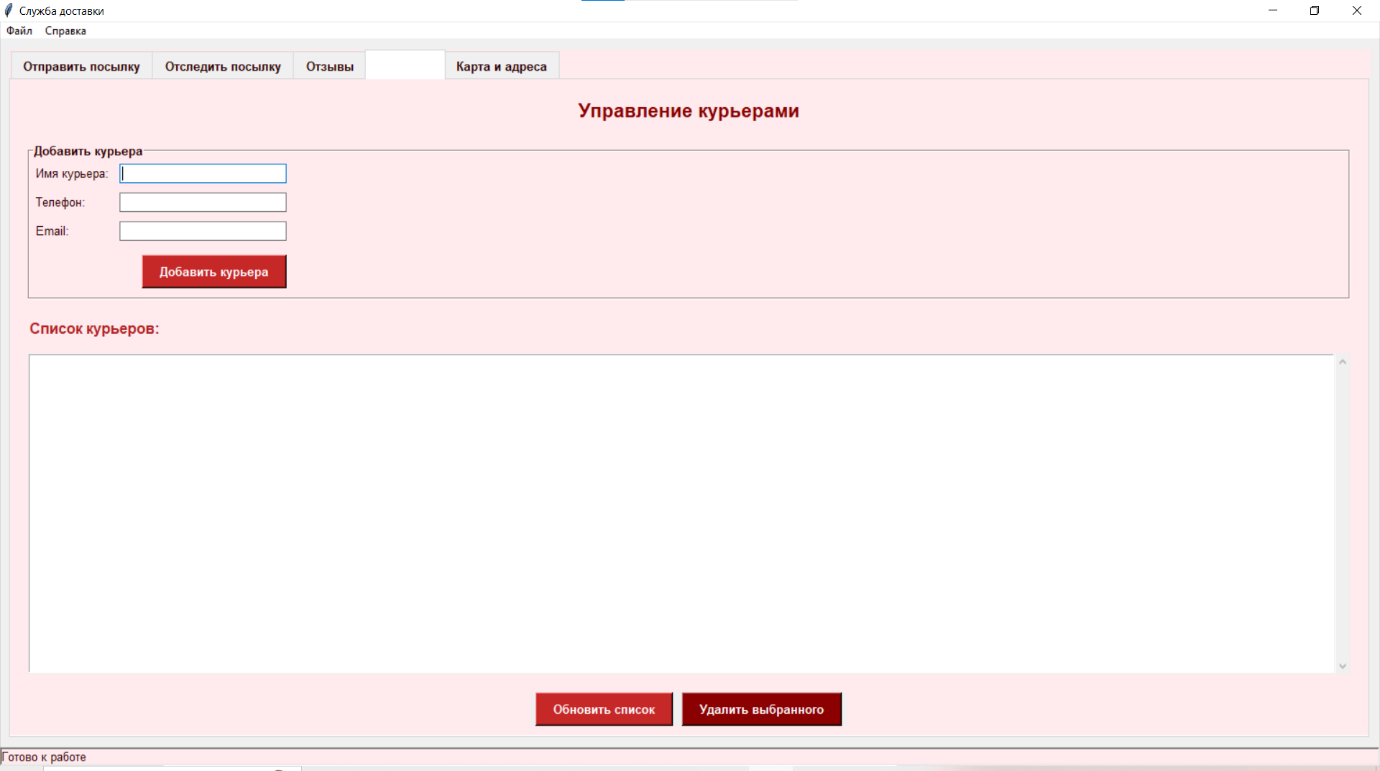
* 1. Отслеживание посылки



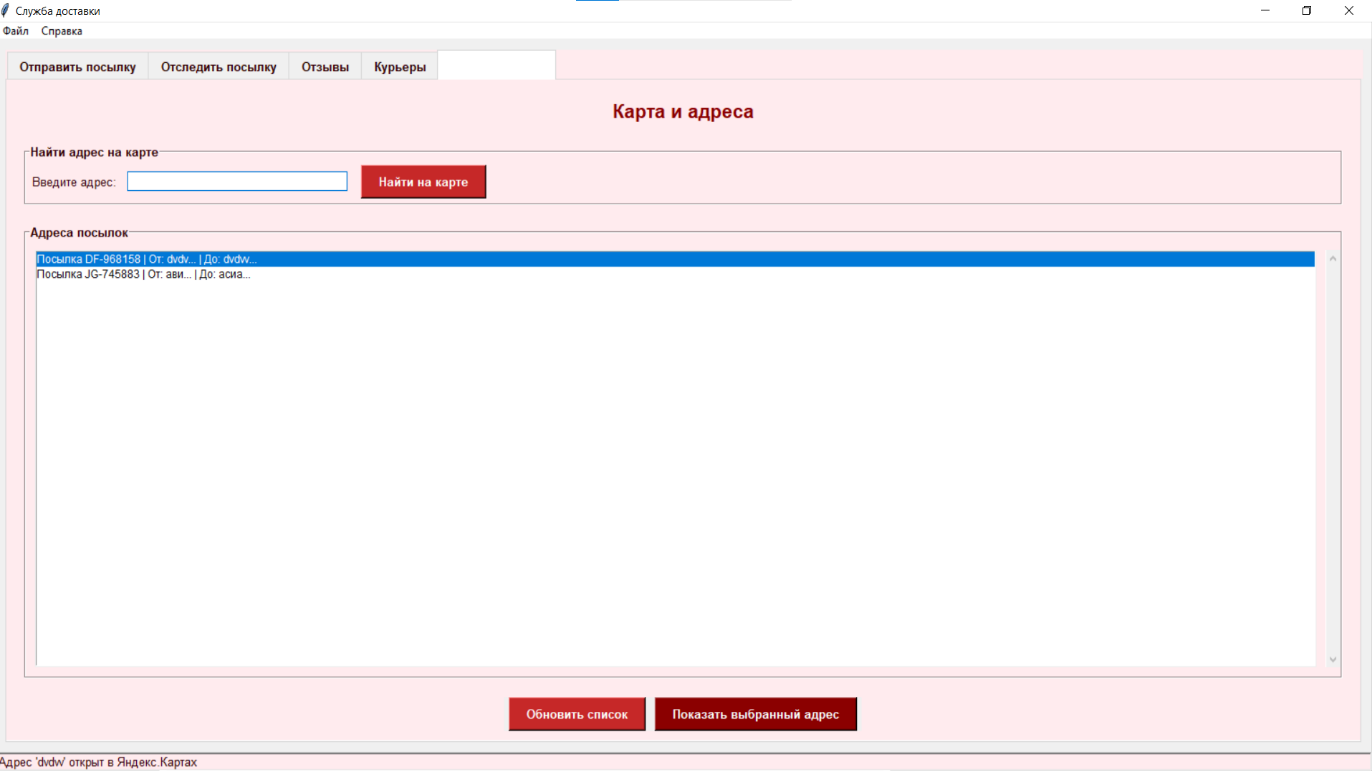
* 1. Отзывы



* 1. Управления курьерами

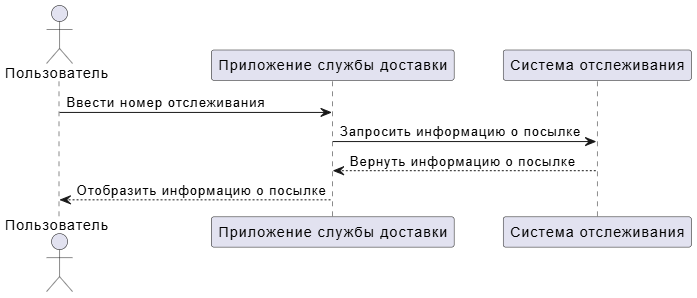


* 1. Карты

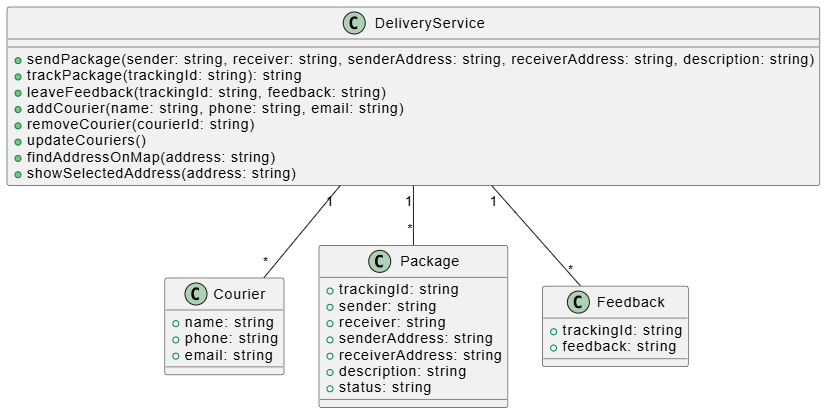


**Диаграммы**

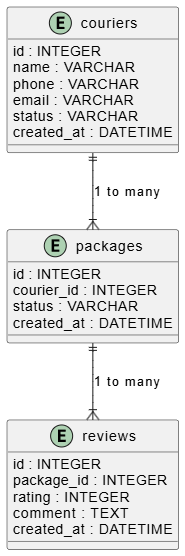
* 1. Диаграмма последовательностей

****

* 1. Диаграмма классов



* 1. ER диаграмма



* 1. Диаграмма прецедентов