

# 1 ОБЗОР ЛИТЕРАТУРЫ

## 1.1 Исследование предметной области

### 1.1.1 Клиент-серверная архитектура системы

«Клиент-сервер» - архитектура, которая чаще всего ложится в основу проектирования современных многопользовательских систем. Ее использование позволяет разделить обязанности и ответственности между клиентами и сервисом. Проектирование разрабатываемой в дипломном проекте системы будет осуществляться в соответствии с клиент-серверной архитектурой.

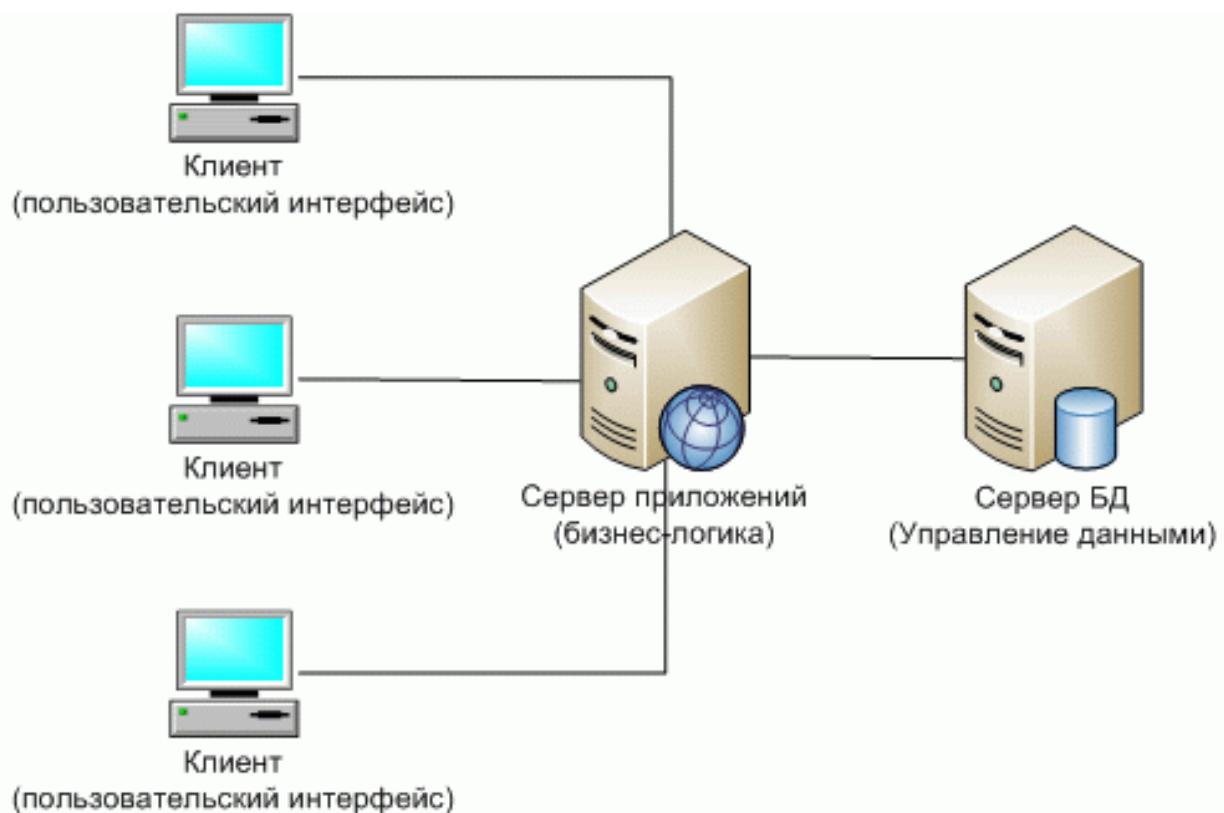


Рисунок 1.1 – Схема клиент-серверной архитектуры

Клиент-серверная архитектура заключает в себе идею использования одной или нескольких машин для создания одного продукта/системы, разделяя обязанности между частями этого продукта/системы. Разделение обязанностей достигается за счет деления системы на несколько программных модулей – клиентское ПО и серверное ПО. Обычно клиентское и серверное ПО взаимодействуют друг с другом посредством общения между устройствами системы с использованием различных сетевых протоколов, например HTTP, FTP, RPC и т.д., однако они также могут находиться и на

одном устройстве, используя для общения внутренние протоколы операционной системы, на которой они выполняются.

Клиент-серверная архитектура, в отличие от peer-to-peer, является централизованной, то есть конечные пользователи (клиенты), как и работоспособность всей системы, зависят от центральной точки – сервера, что является как преимуществом, так и недостатком.

Преимущества архитектуры:

- необходимость в наличии мощного устройства только со стороны сервера, клиентом же могут выступать современные устройства любой мощности, от персональных компьютеров до умных часов и смартфонов;
- необходимость обеспечивать защиту и хранить пользовательские данные только на стороне сервера ввиду того, что хранить их на устройстве конечного пользователя небезопасно;
- отсутствие дублирования кода между различными программными компонентами системы, что позволяет сделать клиенты легкими и быстрыми даже на самых слабых устройствах;

Недостатки архитектуры:

- отказ серверной части архитектуры вызывает неработоспособность всей системы;
- высокая стоимость серверного оборудования ввиду высоких требований к его производительности;
- поддержка серверного оборудования требует выделенных специалистов;

В настоящем проекте применение peer-to-peer архитектуры целесообразно, ввиду непостоянного количества клиентов в сети системы, а также их малой производительности.

### **1.1.2 Распространение на платформах**

На этапе исследования изучены приложения, представленные в магазинах App Store и Google Play. Аналоги разрабатываемого приложения распространяются по схеме In-App Purchases [1]. Существуют несколько типов таких приложений, самыми распространенными из них являются следующие:

- пользователь, скачивая приложение из магазина, получает доступ к его полной версии, и оплата идет непосредственно за услуги, предоставляемые сервисом. Покупки внутри приложений представляют из себя подписки или единоразовые платежи, дающие определенные преимущества в использовании, например скидки на поездки в такси определенного класса;
- пользователь, скачивая приложение из магазина, получает доступ к его ограниченной версии, покупка (подписка или единоразовый платеж) разблокируют недоступный ранее платный функционал.

Сервисы, представленные в магазинах предложений, являются представителями первого типа приложений.

## **1.2 Обзор аналогов**

### **1.2.1 Yandex.Go**

Yandex.Go [2] – это бесплатное приложение, ориентированное на рынок стран СНГ, для пользования услугами такси, перевозок, а так же доставки еды. Грамотно продуманный и реализованный графический интерфейс, позволяющий выполнять все самые популярные задачи одной рукой в несколько нажатий, обеспечивает удобство пользования. Интерфейс приложения представлен на рисунке 1.2.

В Yandex.Go реализована возможность оплаты поездки прямо из приложения. Пользователи могут привязывать карты различных банков и использовать одну из них для последующей оплаты. При списании средств не требуется одноразовый пароль (OTP), это делает оплату моментальной и исключает вмешательство пользователя в процесс.

Важным при использовании таких приложений является точность расчета времени прибытия и показа местоположения. В приложении информация о загруженности дорог и, соответственно, основанный на этих данных маршрут и расчетное время прибытия, вычисляются исходя из данных других сервисов компании Yandex. Необходимые данные вычисляются с использованием сервисов Yandex.Maps и Yandex.Navigator, которые анализируют данные со спутников, подключенных к сети устройств и различных государственных источников, с которыми сотрудничает компания Yandex.

Бесплатная версия приложения предоставляет пользователю функционал, необходимый для заказа такси и перевозки вещей. При покупке подписки пользователь получает доступ к различным бонусам в системе компании Yandex, например:

- бесплатный доступ к различным сервисам компании, таким как Kinopoisk, Yandex.Music и т.д.;
- скидки на поездки в такси определенного класса.

Плюсы сервиса Yandex.Go:

- возможность оплаты картой;
- высокая точность расчета времени прибытия;
- удобный пользовательский интерфейс.

Минусы приложения:

- разные версии приложений под каждую операционную систему, что удорожает процесс разработки;
- отсутствие возможности оплаты мобильными системами;
- функционирует только в некоторых странах СНГ.

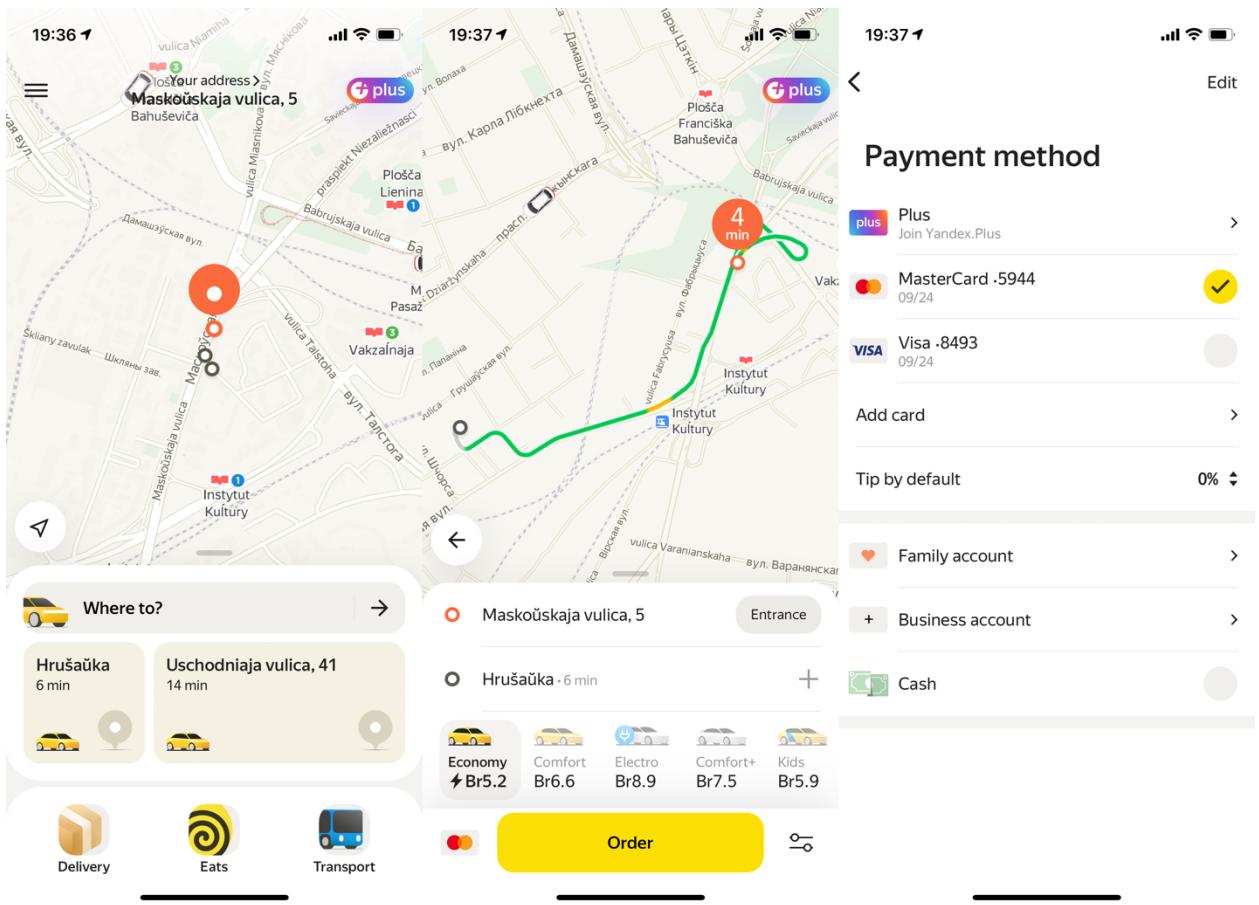


Рисунок 1.2 – Приложение Yandex.Go

## 1.2.2 Uber

Uber [3] – это мобильное приложение для пользования услугами такси международного уровня: сервис является крупнейшим игроком на многих мировых рынках, в том числе американский, европейский и рынок стран СНГ. Пользовательский интерфейс приложения представлен на рисунке 1.3. Бесплатная версия позволяет вызывать такси различных классов и заказывать доставку еды. Преимуществом приложения являются дополнительные системы оплаты.

В Uber, как и приведенном выше аналоге, пользователь может добавлять свои платежные карты в персональный аккаунт, однако помимо этого у пользователя появляется возможность оплаты платформенными сервисами, такими как Apple Pay, Google Pay и т.д., что позволяет пользователю избегать предоставления своих персональных данных третьим лицам.

Как и в большинстве аналогичных приложений, пользователь может получить доступ к дополнительному функционалу приложения с помощью подписки. Дополнительные функции, предоставляемые по подписке, связаны с получением различных скидок и бонусов. В бесплатной версии приложения вызов такси или доставки предполагает оплату по обычной цене с надбавкой в качестве платы за услуги сервиса. При покупке подписки у пользователя

появляется дополнительная скидка на такси и доставку, а также появляются дополнительные места, из которых можно заказывать еду.

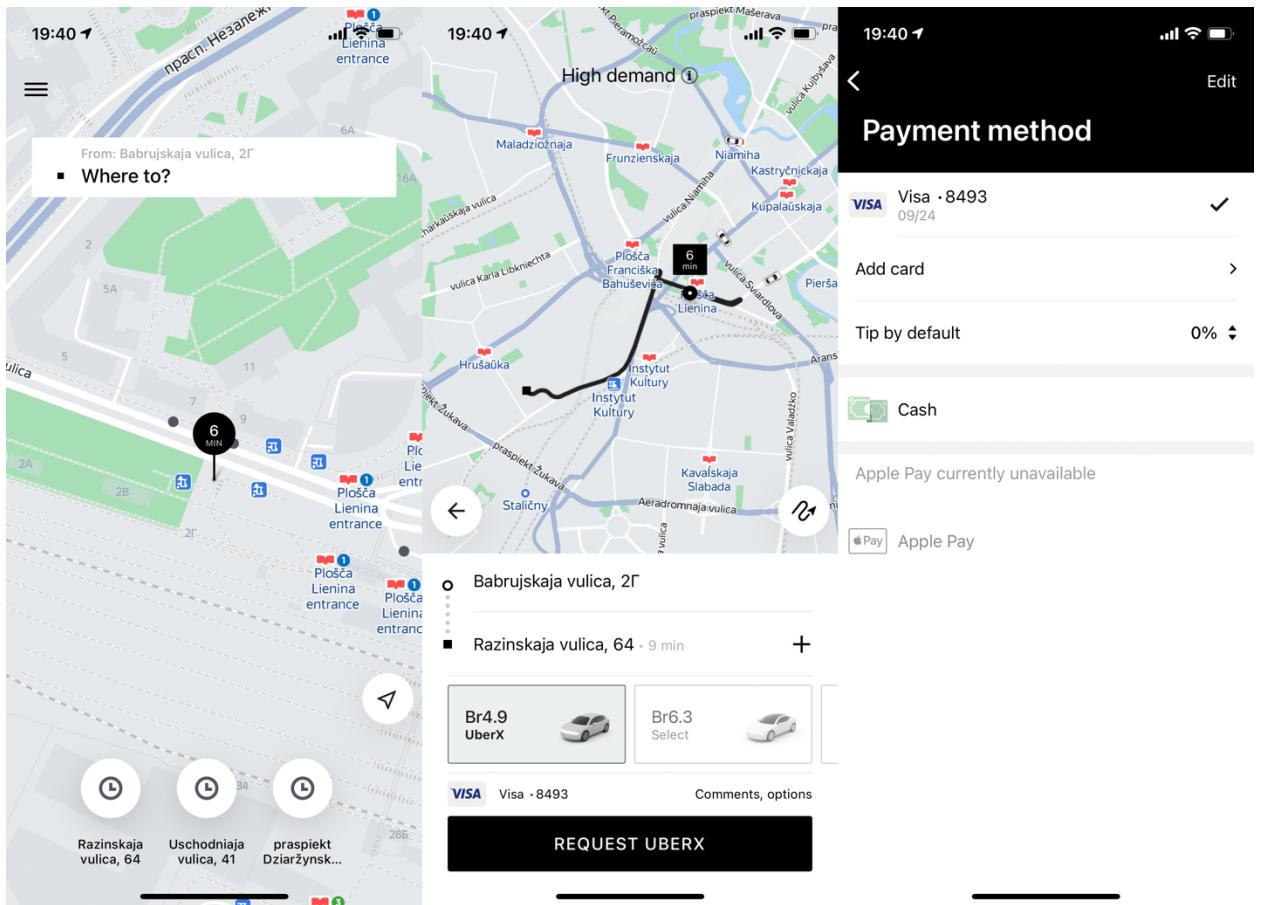


Рисунок 1.3 – Приложение Uber

#### Плюсы сервиса Uber:

- возможность оплаты картой и мобильными системами (Apple Pay, Google Pay);
- высокая точность расчета времени прибытия;
- работает в большинстве стран мира.

#### Минусы приложения:

- разные версии приложений под каждую операционную систему, что удорожает процесс разработки;
- недостатки в пользовательском интерфейсе (например расположение строки поиска сверху экрана, при использовании телефона одной рукой требуется перехватывать устройство).

### 1.2.3 Bolt

Bolt [4] дает возможность заказывать такси в различных точках мира. Пользовательский интерфейс приложения представлен на рисунке 1.4.

Особенностью сервиса является экологичность транспорта и высокая квалификация водителей, что делает его лучшим на рынке Европы. Сервис предоставляет удобный интерфейс и высокую отзывчивость как приложения, так и его серверов, что делает работу с ним плавной и быстрой. Преимуществами приложения Bolt является возможность оплаты через платформенные сервисы, а также расширенные возможности по авторизации.

Bolt предоставляет пользователям возможность заказа такси, однако, в отличие от конкурентов, оно позволяет использовать уже существующие аккаунты различных сервисов и социальных сетей для авторизации в своей системе, что избавляет пользователя от необходимости предоставления данных и заполнения форм регистрации.

Bolt является абсолютно бесплатным и не предоставляет подписок для своих пользователей, зарабатывая исключительно на перевозках и транспортировке.

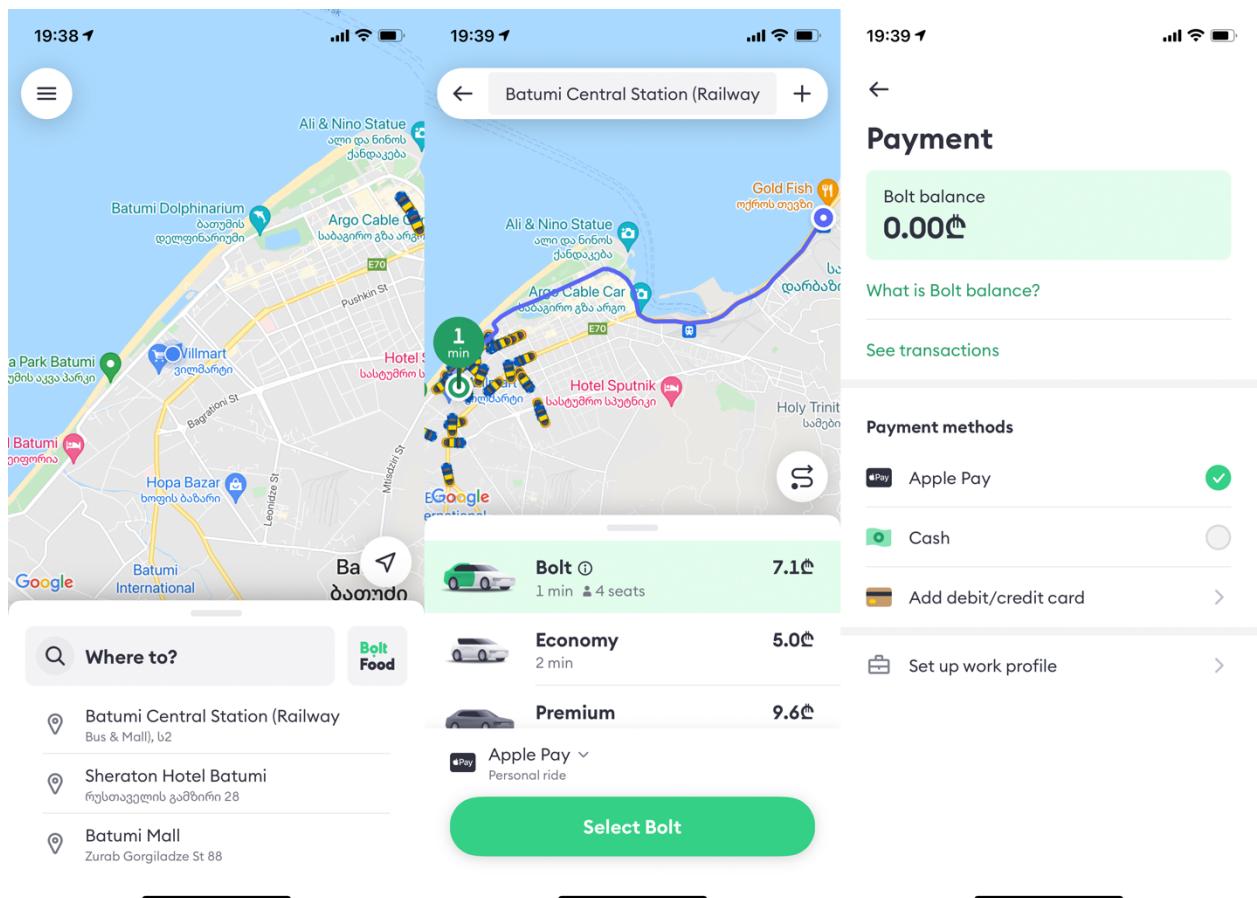


Рисунок 1.4 – Приложение Bolt

#### Плюсы сервиса Bolt:

- возможность оплаты картой и мобильными системами (Apple Pay, Google Pay);
- возможность авторизации с помощью сторонних сервисов;

– удобный пользовательский интерфейс и высокая скорость работы.

Минусы приложения:

- разные версии приложений под каждую операционную систему, что удорожает процесс разработки;
- списание средств происходит до подтверждения заказа водителем;
- функционирует в ограниченном количестве стран.

#### 1.2.4 135

135 [5] – сервис для пользования услугами такси и перевозками на территории Республики Беларусь. Он является самым крупным исключительно белорусским сервисом, а также одним из первых на рынке. Сервис предоставляет для пользователей не только мобильное приложение, но еще и сайт, что позволяет охватить наибольший круг пользователей. Внешний вид мобильного приложения представлен на рисунке 1.5.

Приложение 135, как и Bolt, является абсолютно бесплатным и не предоставляет моделей подписки.

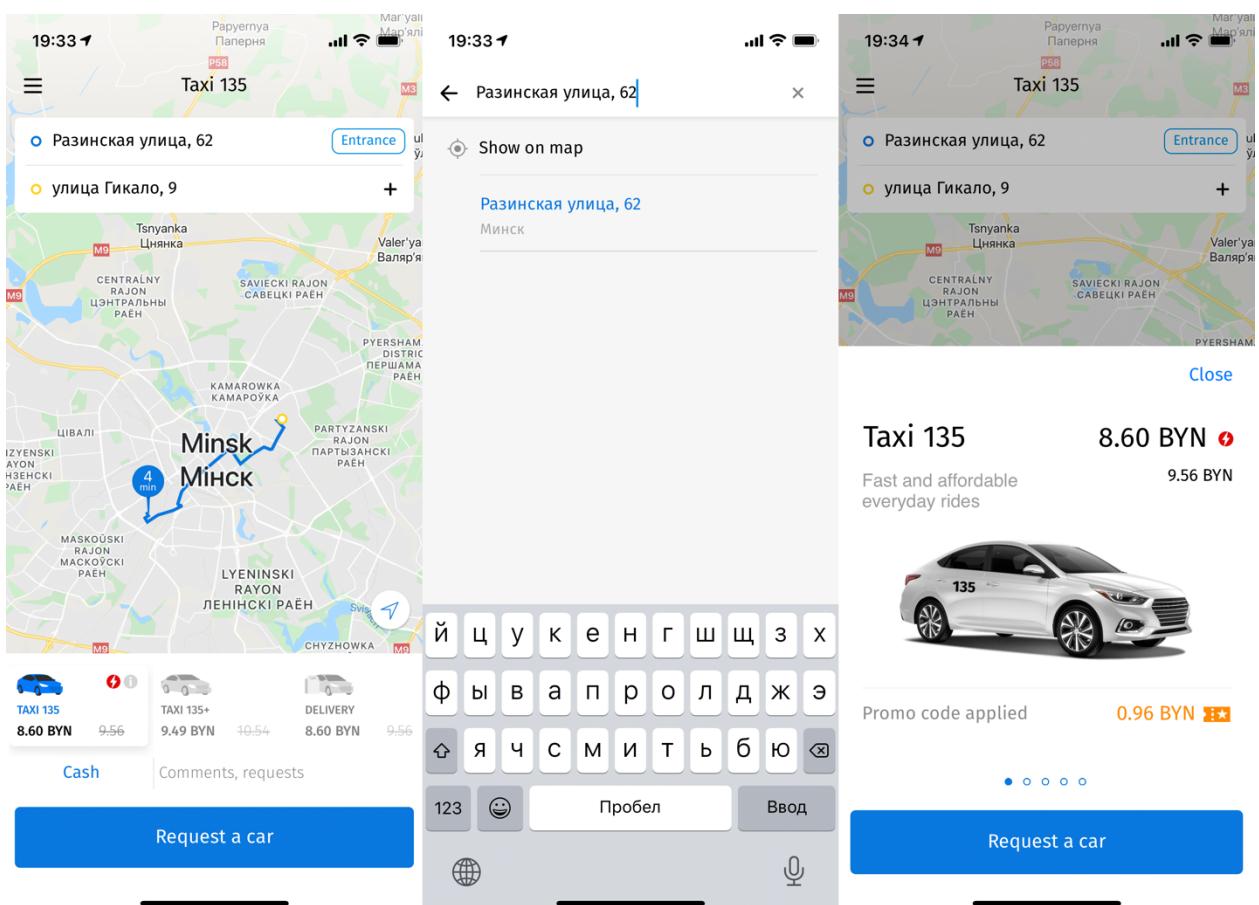


Рисунок 1.5 – Приложение 135

Плюсы сервиса 135:

- возможность оплаты картой;
- возможность вызова микроавтобусов при поездке большой компанией.

Минусы приложения:

- разные версии приложений под каждую операционную систему, что удорожает процесс разработки;
- малое количество машин;
- отсутствие возможности оплаты мобильными системами.

### 1.2.5 Maxim

Maxim [6] – сервис по вызову такси, работающий в более чем 1000 городах мира. Основанная в 2003 году, платформа начала набирать популярность в России, а позже начала распространяться в странах ближнего зарубежья. Пользовательский интерфейс представлен на рисунке 1.6. Несмотря на устаревший дизайн, приложение обладает достаточно широким функционалом, который отсутствует у аналогов.

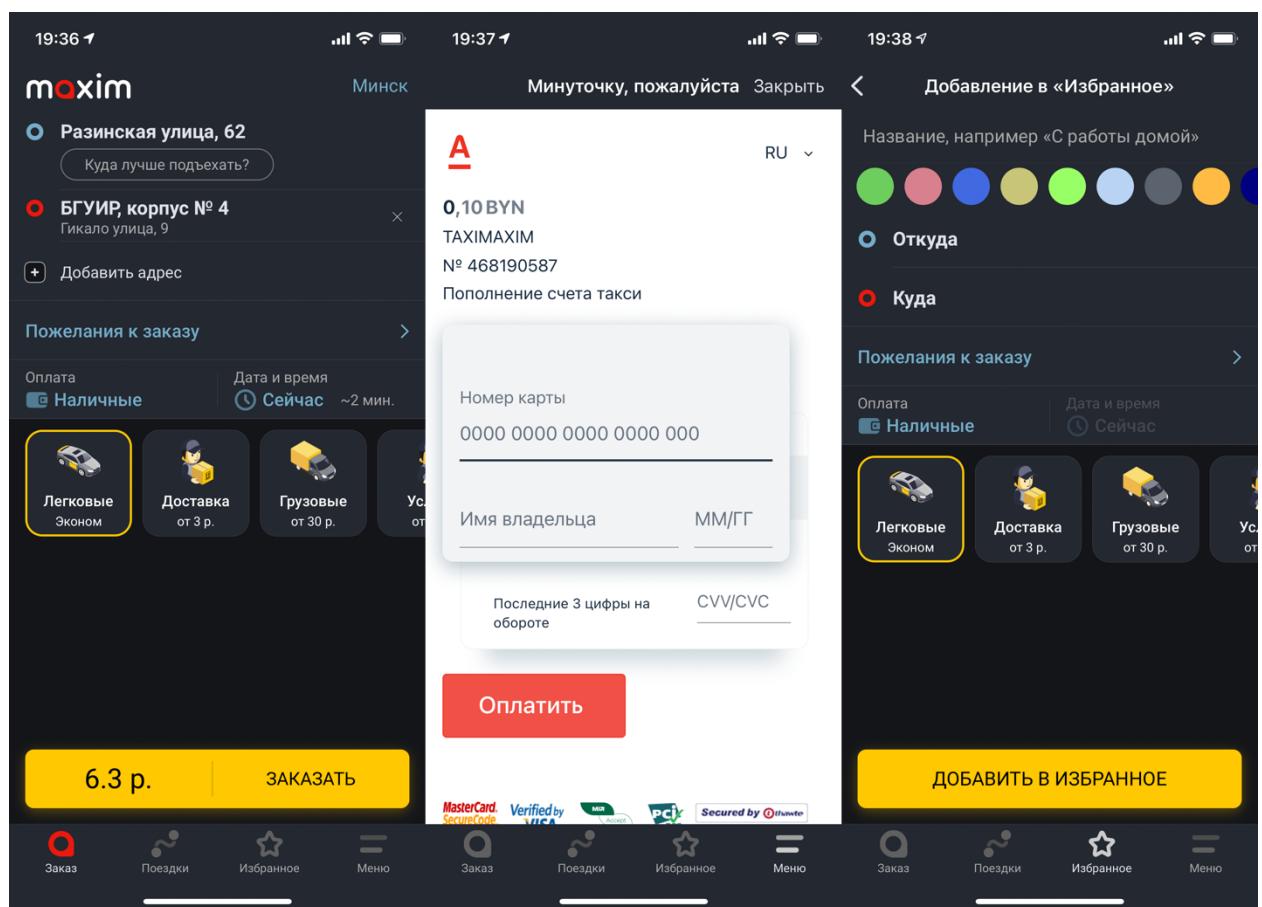


Рисунок 1.6 – Приложение Maxim

Приложение Maxim распространяется бесплатно и не предоставляет подписочную модель.

Плюсы сервиса Maxim:

- возможность оплаты картой;
- возможность добавление избранных маршрутов;
- возможность выбрать время подачи машины.

Минусы приложения:

- добавление карты происходит в веб-интерфейсе;
- устаревший дизайн;
- малое количество машин.

## 1.3 Обзор технологий и инструментов

### 1.3.1 Языки программирования

#### 1.3.1.1 Java

Java — строго типизированный объектно-ориентированный язык программирования общего назначения, разработанный компанией Sun Microsystems. Программы на Java транслируются в байт-код Java, выполняемый виртуальной машиной Java (JVM) — программой, обрабатывающей байтовый код и передающей инструкции оборудованию как интерпретатор.

#### 1.3.1.2 Swift

Swift — открытый мультипарадигмальный компилируемый язык программирования общего назначения. Создан компанией Apple в первую очередь для разработчиков iOS и macOS. Swift задумывался как более лёгкий для чтения и устойчивый к ошибкам программиста язык, нежели предшествовавший ему Objective-C. Программы на Swift компилируются при помощи LLVM. Swift может использовать рантайм Objective-C, что делает возможным использование обоих языков (а также C) в рамках одной программы.

#### 1.3.1.3 TypeScript

TypeScript — язык программирования, представленный Microsoft в 2012 году и позиционируемый как средство разработки веб-приложений, расширяющее возможности JavaScript. TypeScript отличается от JavaScript возможностью явного статического назначения типов, поддержкой использования полноценных классов (как в традиционных объектно-ориентированных языках), а также поддержкой подключения модулей, что

призвано повысить скорость разработки, облегчить читаемость, рефакторинг и повторное использование кода, помочь осуществлять поиск ошибок на этапе разработки и компиляции, и, возможно, ускорить выполнение программ.

### **1.3.1.4 Python**

Python — высокоуровневый язык программирования общего назначения с динамической строгой типизацией и автоматическим управлением памятью. Python является мультипарадигмальным языком программирования, поддерживающим императивное, процедурное, структурное, объектно-ориентированное программирование, метапрограммирование и функциональное программирование. Стандартная библиотека включает большой набор полезных переносимых функций, начиная с возможностей для работы с текстом и заканчивая средствами для написания сетевых приложений.

## **1.3.2 Клиентская часть**

Клиентская часть разрабатываемой системой будет представлена мобильным приложением. Для выбора оптимальной технологии требуется сделать обзор доступных вариантов.

### **1.3.2.1 Android SDK**

Android SDK [7] – официальный набор утилит от Google для разработки под операционную систему Android. Разработана с использованием языков Kotlin, Java и C++. Позволяет разрабатывать высокопроизводительные приложения для платформ Android, Android TV и WearOS. В состав SDK включены различные средства разработки, в том числе отладчик, набор библиотек, телефонный эмулятор на базе движка QEMU, набор документации, примеров приложений и руководств. Среда Android SDK может быть запущена на компьютерах, использующих ОС Linux, Mac OS X 10.5.8 и новее, Windows 7 и новее.

### **1.3.2.2 iOS SDK**

iOS SDK [8] – официальный набор утилит от Apple для разработки под операционную систему iOS. Разработана с использованием языков Objective-C, Swift и C++, позволяет разрабатывать оптимизированные приложения для модельного ряда устройств iPhone. Наряду с набором инструментов Xcode, SDK содержит iPhone Simulator, используемый для имитации внешнего вида iPhone на компьютере разработчика, ранее называвшийся «Aspen Simulator». Новые версии SDK сопровождают новые версии iOS. Чтобы тестировать приложения, получать техническую поддержку и распространять приложения

через App Store, разработчикам необходимо подписаться на программу Apple Developer Program.

### **1.3.2.3 React Native**

React Native [9] – фреймворк для создания кроссплатформенных приложений, разработанный компанией Facebook. Данный фреймворк позволяет разрабатывать приложение под несколько платформ (например, iOS и Android) с использованием лишь одного языка – TypeScript. Это преимущество компенсируется более низкой производительностью в сравнении с Android и iOS SDK. TypeScript-код, написанный разработчиком, выполняется в фоновом потоке, и взаимодействует с платформенными API через асинхронную систему обмена данными, называемую Bridge.

### **1.3.2.4 Flutter**

Flutter [10] - фреймворк для создания кроссплатформенных приложений, разработанный компанией Google. Как и React Native, данный фреймворк позволяет писать код на несколько платформ с использованием одного языка – Dart. Показатели производительности фреймворка также ниже, в сравнении с Android и iOS SDK. Из-за ограничений на динамическое выполнение кода в App Store, под iOS Flutter использует АОТ-компиляцию. Широко используется такая возможность платформы Dart, как «горячая перезагрузка», когда изменение исходного кода применяется сразу в работающем приложении без необходимости его перезапуска.

## **1.3.3 Серверная часть**

Серверное ПО является центром в клиент-серверной архитектуре: тут происходит обработка данных, платежей и прочая специфичная каждому приложению бизнес-логика. Рассмотрим самые популярные технологии для построения такого ПО.

### **1.3.3.1 Spring**

Spring Framework [11] — универсальный фреймворк с открытым исходным кодом для Java-платформы. Spring дает большую свободу и гибкость разработчикам, при этом предоставляя эффективные и мощные утилиты для работы с такими важными вещами, как безопасность и хранение данных. Этот фреймворк предлагает последовательную модель и делает её применимой к большинству типов приложений, которые уже созданы на основе платформы Java. Считается, что Spring реализует модель разработки, основанную на лучших стандартах индустрии, и делает её доступной во многих областях Java.

### **1.3.3.2 Nest**

Nest (NestJS) — это фреймворк для создания эффективных масштабируемых серверных приложений NodeJS. Он сочетает в себе элементы ООП (объектно-ориентированное программирования), ФП (функционального программирования) и ФРП (функционально-реактивного программирования). Nest не только обеспечивает дополнительный уровень абстракции над распространенными платформами NodeJS (Express/Fastify), но также предоставляет свой собственный API разработчику, что дает ему свободу в использовании сторонних модулей.

### **1.3.3.3 Django**

Django — это высокоуровневый фреймворк, который способствует быстрой разработке и чистому прагматичному дизайну. Является самым популярным средством разработки веб-серверов на языке Python с широким набором встроенных средств и утилит. Django используется в сайтах Instagram, Mozilla, The Washington Times, Pinterest, YouTube, Google и др. На базе Django разработан ряд готовых решений со свободной лицензией, среди которых интернет-магазины, системы управления содержимым, а также более узконаправленные проекты.

## **1.3.4 Система управления базами данных**

База данных является важной частью многопользовательских систем. При проектировании таких приложений выбор базы данных проводится путем строгого анализа и отбора по целому ряду пунктов, например скорость и отказоустойчивость. Оптимальными являются несколько СУБД, описанных ниже.

### **1.3.4.1 PostgreSQL**

PostgreSQL [13] — свободная объектно-реляционная система управления базами данных. Существует в реализациях для множества операционных систем, в том числе Windows, macOS и Linux. PostgreSQL базируется на языке SQL и поддерживает многие из возможностей стандарта SQL 2011. Сильными сторонами PostgreSQL считаются:

- высокопроизводительные и надёжные механизмы транзакций и репликации;
- наследование;

- возможность индексирования геометрических (в частности, географических) объектов и наличие базирующегося на ней расширения PostGIS.

#### **1.3.4.2 MongoDB**

MongoDB — документно-ориентированная система управления базами данных, не требующая описания схемы таблиц. На данный момент является самой популярной NoSQL системой, в качестве схемы данных используется JSON. Система масштабируется горизонтально, используя технику сегментирования объектов баз данных — распределение их частей по различным узлам кластера. Администратор выбирает ключ сегментирования, который определяет, по какому критерию данные будут разнесены по узлам (в зависимости от значений хэша ключа сегментирования). Благодаря тому, что каждый узел кластера может принимать запросы, обеспечивается балансировка нагрузки.

### **1.4 Постановка задачи**

Разрабатываемая система должна выдавать максимальную производительность при минимальных временных и денежных затратах на разработку.

Клиентская часть должна быть отзывчивой и не вызывать дискомфорт у пользователя, а также должна предоставлять следующие функции:

- функция регистрации и входа в систему;
- возможность добавления и оплаты картой;
- возможность выбора места отправки и назначения на карте или в поле для ввода адреса;
- просмотр истории поездок и профиля пользователя;
- отслеживание местоположения водителя и машины;
- принятие заказа, если пользователь приложения – водитель такси.

Серверная часть должна максимально быстро обрабатывать входящие запросы клиентов и поддерживать с ними соединение для передачи текущего местоположения. Серверное ПО должно соответствовать следующим требованиям:

- высокие показатели по пропускной способности;
- эффективное расходование ресурсов;
- минимальное время, необходимое для разработки, внесения изменений и разворачивания системы.