ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ

ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ

«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»

Факультет компьютерных наук

Департамент программной инженерии

|  |  |
| --- | --- |
|  | УТВЕРЖДАЮ  Академический руководитель образовательной программы «Программная инженерия»,  профессор департамента программной инженерии, канд. техн. наук  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ В.В. Шилов  «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_ 2021 г. |

Выпускная квалификационная работа

Система мониторинга транспортных средств на базе мобильных устройств

по направлению подготовки 09.03.04 «Программная инженерия»

|  |  |
| --- | --- |
| Научный руководитель:  Кандидат технических наук, доцент департамента программной инженерии,  Салех Хади Мухаммед  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Оценка  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Подпись, Дата | Выполнил:  студент группы БПИ173  4 курса бакалавриата  образовательной программы «Программная инженерия»,  С.И. Ройтман  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Подпись, Дата |

# Аннотация

Работа посвящена системе мониторинга транспортных средства, использующей мобильные устройства в качестве передатчика данных.

В работе объясняется эксплуатационное назначение систем данного типа, проводится сравнение разработанной системы с популярными существующими решениями, а также рассматриваются ее конкурентные преимущества.

Работа содержит описание программной реализации и функциональные характеристики клиентского приложение для Android, используемого водителями транспортных средств, клиентского веб-приложение на React, используемого операторами и администраторами системы, серверной части на ASP .NET Core.

Статья содержит //TODO страниц, главы, иллюстраций, таблицы, источника и приложения.

Ключевые слова – система мониторинга транспортных средств; клиент-серверное приложение; Android; GNSS-трекер; OBD-|| порт; Bluetooth

# Abstract

This work is dedicated to vehicle monitoring system, which use mobile device as a data transmitter.

The paper explains the operation purpose of such systems, compares the developed system with popular existing solutions, and also considers its competitive advantages.

The work contains a description of the software implementation and functional characteristics of an Android client application used by vehicle drivers, a React web-client application used by system operators and administrators and an ASP .NET Core server side.

The paper contains //TODO pages, 3 chapters, 36 illustrations, 3 tables, 33 bibliography items and 4 appendices.

Keywords – vehicle monitoring system; client-server application; Android; GNSS-tracker; OBD-|| port; Bluetooth

# Определения

1. *GNSS –* Спутниковая система навигации (англ. Global Navigation Satellite System, GNSS, ГНСС) — система, предназначенная для определения местоположения (географических координат) наземных, водных и воздушных объектов, а также низкоорбитальных космических аппаратов. Спутниковые системы навигации также позволяют получить скорость и направление движения приёмника сигнала. В настоящее время только две спутниковые системы обеспечивают полное покрытие и бесперебойную работу для всего земного шара — GPS и ГЛОНАСС.
2. *OBD-****||*** *порт* – 16-пиновый порт, позволяющий подключиться к системе On-board Diagnostict-**||**, установленной на бортовом компьютере всех автомобилей, выпущенных после 1996 года, для получения данных о состоянии автомобиля.
3. *Автомобильный GNSS-трекер, трекер, терминал* – устройство приёма-передачи данных для спутникового контроля автомобилей, использующее спутниковую систему навигации для точного определения местонахождения объекта и OBD-**||** порт для получения данных о состоянии автомобиля с бортового компьютера.
4. *Bluetooth -* производственная спецификация беспроводных персональных сетей. Bluetooth обеспечивает обмен информацией между устройствами на надёжной, бесплатной, повсеместно доступной радиочастоте для ближней связи. Bluetooth позволяет устройствам сообщаться, когда они находятся друг от друга в радиусе около 100 м в старых версиях протокола и до 1500 м начиная с версии Bluetooth

Содержание

[Аннотация 2](#_Toc72706925)

[Abstract 3](#_Toc72706926)

[Определения 4](#_Toc72706927)

[Введение 6](#_Toc72706928)

[Глава 1. Обзор существующих систем мониторинга транспортных средств 8](#_Toc72706929)

[1.1. Сравнительный анализ систем мониторинга транспортных средств 8](#_Toc72706930)

[1.2. Выводы по главе 11](#_Toc72706931)

[Глава 2. Проектирование системы 12](#_Toc72706932)

[3.1. Высокоуровневая архитектура систем 12](#_Toc72706933)

[3.2. Функциональные характеристики 13](#_Toc72706934)

[3.2.1. Android приложение 13](#_Toc72706935)

[3.2.2. Сервис обработки данных 15](#_Toc72706936)

[3.2.3. Клиентская часть web-приложения 15](#_Toc72706937)

[3.2.4. Серверная часть 16](#_Toc72706938)

[3.3. Информационная модель и базы данных 16](#_Toc72706939)

[3.4. Проектирование сервиса обработка данных 19](#_Toc72706940)

[3.5. Проектирование мобильного приложения 19](#_Toc72706941)

[3.5.1. Общая архитектура 19](#_Toc72706942)

[3.5.2. Взаимодействие с API 21](#_Toc72706943)

[3.5.3. Навигация 22](#_Toc72706944)

[3.5.4. Запись данных автомобиля 23](#_Toc72706945)

[3.6. Bluetooth OBD трекер 24](#_Toc72706946)

[3.7. Проектирование клиентского веб-приложения 26](#_Toc72706947)

[3.7.1. Общая архитектура 26](#_Toc72706948)

[3.7.2. Навигация 27](#_Toc72706949)

[3.8. Проектирование серверной части 29](#_Toc72706950)

[3.9. Выводы по главе 29](#_Toc72706951)

[Глава 3. Разработка системы 30](#_Toc72706952)

[3.1. Сервис обработки данных 30](#_Toc72706953)

[3.2. Android-приложение 30](#_Toc72706954)

[3.3. Интеграция с Bluetooth-трекером 30](#_Toc72706955)

[3.4. Клиентское Веб-приложение 30](#_Toc72706956)

[3.5. Серверная часть 30](#_Toc72706957)

[3.6. Выводы по главе 30](#_Toc72706958)

[Заключение 31](#_Toc72706959)

[Приложение А (Техническое задание) 32](#_Toc72706960)

[Источники 33](#_Toc72706961)

# Введение

Первая система мониторинга транспортных средств была разработана в 1982 году, а в 1994 году был запущен последний из первых двадцати четырех спутников, и система GPS (первая из спутниковых систем навигации) была признана полностью работоспособной. С тех пор системы мониторинга транспортных средств прошли через значительные улучшения. Современные решения предлагают онлайн-системы управления автопарком, которые включают в себя широкий функционал мониторинга транспортных средств.

Как правило, такие системы предоставляют функционал по отслеживанию местоположения транспортных средств в режиме реального времени; позволяют собирать, агрегировать и анализировать технические данные о состоянии машины с их бортовых компьютеров; используются для планирования технического обслуживания транспортных средств; управления задачами для водителей и мониторинга их производительности; фиксирования внештатных ситуаций таких как: превышение скорости [1], вождения в нетрезвом виде [2], аварий, угона и так далее.

Системы такого рода широко используются для управления автопарками в государственном и частном секторах: службы доставки, такси, общественный транспорт, контроль командировок.

Мировой объем рынка систем мониторинга транспортных средств в 2020 году составлял 21.1 миллиард долларов США. Исследование аналитического агентства «Grand View Search» [3] прогнозирует рост рынка в среднем на 14% в год в период с 2020 по 2027 год, т.е. к 2027 году, по их прогнозам, объем мирового рынка составит 52.8 миллиардов долларов США.

Российский рынок также активно растет. По оценкам некоммерческого партнерства «ГЛОНАСС», опубликованных в 2020 году [4], российский рынок ИТ-услуг для транспорта растет на 24% в год. Одними из основных драйверов роста являются системы «ГЛОНАСС» и «ПЛАТОН», количество автомобилей, подключенных хотя бы к одной из этих систем, достигло 30% от общего числа автомобилей в РФ в середине 2020 года.



Рисунок . Динамика европейский рынок систем мониторинга транспортных средств по прогнозам аналитического агентства «Grand View Search»

Большинство современных систем мониторинга транспортных средств используют аппаратный подход для решения проблемы сбора и передачи данных с бортовых компьютеров машин: через OBD-**||** порт к бортовому компьютеру машины подключается физический GNSS-трекер с встроенной сим-картой, он собирает необходимую информацию о состоянии автомобиля и через мобильную сеть передает ее на сервер. Как правило, для синхронизации с серверами системы мониторинга необходимы устройства, выпущенные той же компанией, а покупать и устанавливать их надо за отдельные деньги.

Цель данной работы – разработать систему мониторинга транспортных средств, использующую Android-приложение вместо GNSS-трекеров. Такой подход позволит сэкономить деньги на покупке и установке дорогостоящих трекеров, заменив их широко распространенными среди водителей автомобилей Android-устройствами.

Приложение, установленное на Android, уже имеет встроенные механизмы для определения своего местоположения, а с использованием дешевых Bluetooth-терминалов, подключаемых к OBD-**||** порту автомобиля, оно также получит доступ к данным с бортового компьютера: оборотам двигателя, скорости автомобиля, пробегу, температуре охлаждающей жидкости и многим другим данным, что позволит полностью покрыть функциональность современных систем мониторинга транспортных средств.

# Глава 1. Обзор существующих систем мониторинга транспортных средств

В данной главе приводится анализ существующих систем мониторинга транспортных средств и проводиться их сравнение с разрабатываемым решением.

## Сравнительный анализ систем мониторинга транспортных средств

Анализируемые системы представлены четырьмя наиболее популярными на российском рынке решениями в области систем мониторинга транспортных средств, по данным исследования компании Omnicomm 2018 года [5]:

* Gurtam (система Wialon [6])
* ТекноКом (система АвтоГраф [7])
* Omnicomm (система Omnicomm Online [8])
* СКАУТ (система Scout Online 365 [9]).

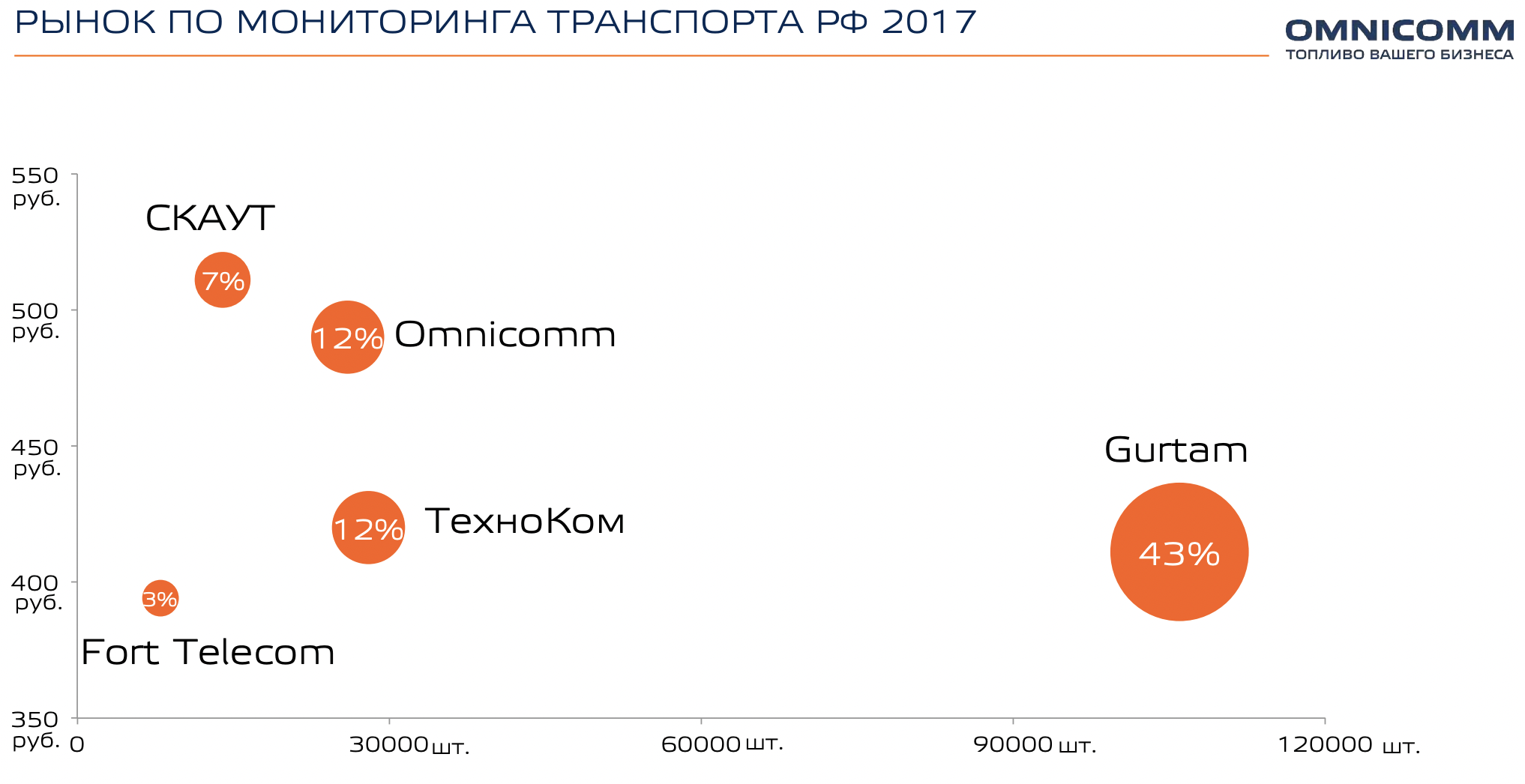


Рисунок 2. Российский рынок систем мониторинга транспортных средств в 2017 году, по данным исследования компании Omnicomm [4]

Данные системы покрывают три четверти российского сегмента рынка. Анализ их функциональности позволит получить объективные функциональные характеристики систем данного типа и определить функциональность разрабатываемого решения.

Все перечисленные аналоги в своей работе используют GNSS-трекеры, подключающиеся к бортовому компьютеру автомобиля и передающие информацию о его состоянии на сервер.

Функциональность данных систем практически идентична функциональности лидера рынка – системе Wialon, которая включает в себя:

* Сбор данных о состоянии автомобиля и передача их на сервер, реализовано с помощью GNSS-трекера, имеющего доступ к мобильной сети
* Механизм синхронизации данных с сервером при отсутствии мобильной сети
* CRUD операции (создание / чтение / редактирование / удаление) над объектами транспортных средств и водителей и их связями
* Визуализация местоположения транспортных средств и данных о их передвижениях на карте
* Визуализация данных о состоянии автомобилей: как уровень топлива, пробег, скорость транспортного средства, уровень масла и другие показатели, настраиваемые при наличии дополнительных устройств, подключенных к компонентам транспортного средства
* Управление планами технического обслуживания транспортных средств
* Управление задачами для водителей
* Настраиваемые аналитические отчеты: выполнение задач, превышение скорости
* Информационная панель с статистикой по автопарку: графики и числа, отражающие активность автопарка в течение настраиваемого периода времени

В основной функциональности предлагаемое решение повторяет аналоги, так что сравнительный анализ будет содержать в себе критерии по функциональности, различающейся между обозреваемыми системами:

* Необходимость наличия трекера для работы системы
* Поддерживаемые трекеры
* Стоимость поддерживаемых трекеров, рубли
* Наличие мобильного приложения для водителей
* Управление задачами водителей
* Коммуникация с водителем
* Система уведомлений

Таблица 1

Сравнительный анализ конкурентов

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Критерий/Система | Wialon | АвтоГраф | Omnicomm Online | Scout Online 365 | Vehicle Monitoring System |
| Необходимость наличия трекера для работы системы | Необходим | Необходим | Необходим | Необходим | Данные о местоположении система будет получать от Android-устройства – это необходимый минимум для работы системы. Остальные показатели состояния автомобиля можно получить через Bluetooth-трекер |
| Поддерживаемые трекеры | 1600 видов различных трекеры с встроенной SIM-картой | Только собственные трекеры | Только собственные трекеры | Только собственные трекеры | Любые трекеры с поддержкой Bluetooth |
| Стоимость поддерживаемых трекеров, рубли | от 2000 | В среднем 9100 | В среднем 9600 | В среднем 8600 | от 300 |
| Наличие мобильного приложения для водителей | +, но приложение ориентировано не на водителей, а на операторов – не позволяет управлять задачами или общаться с оператором | +, демоверсия, но приложение ориентировано не на водителей, а на операторов – не позволяет управлять задачами или общаться с оператором | +, но приложение ориентировано не на водителей, а на операторов – не позволяет управлять задачами или общаться с оператором | - | +, ориентированно на водителей, позволяет управлять задачами и общаться с оператором |
| Управления задачами водителей | + | - | - | - | + |
| Коммуникация с водителями | - | Посредством звонка на SIM-карту, установленную в трекере | - | - | Посредством задач и комментариев к ним и чата |
| Система уведомлений | + | + | + | - | + |

Из сравнительного анализа конкурентов можно сделать выводы об экономической привлекательности данного проекта: его минимальная конфигурация способна работать даже при отсутствии прибора, подключаемого к бортовому компьютеру автомобиля, как такового – в системе не будет данных с бортового компьютера (скорости, оборотов и температуры двигателя и т.д.), но будут данные о местоположении Android-устройства водителя, что достаточно для работы всего остального функционала. А если компании-заказчику необходимы дополнительные данные, то вместо того, чтобы покупать трекеры с встроенной SIM-картой (также называемые «терминалами»), средняя стоимость которых, согласно данным исследования компания Omnicomm [4], составляла 6472 рубля в 2017 году, компания может подобрать различные доступные OBD-трекеры, не имеющие встроенной SIM-карты, но способные передавать данные о состоянии автомобиля с помощью технологии Bluetooth, стоимостью от 400 рублей [10] (при этом функциональность они предоставляют одинаковую, так что можно купить самую дешевую версию).



Рисунок 3. Средняя стоимость терминала (GNSS-трекера), по данным исследования компании Omnicomm [4]

Кроме этого, абонентская плата, взымаемая за использование мобильной сети терминалами, в среднем составляет 408 рублей в месяц, по данным того же исследования [4], в то время как трафик мобильной сети Android-устройства входит в тарифный план водителя и оплачивается самим водителем, который соответственно может подобрать более выгодный тарифный план.

## Выводы по главе

В первой главе был проведен сравнительный анализ наиболее популярных на российском рынке систем мониторинга транспортных средств: была описана их функциональность, рассмотрены их сходства и различия. Кроме этого, было проведено сравнение существующих решений с решением, описанным в этой работе, были указаны его основные конкурентные преимущества

В следующей главе будет описана функциональность разрабатываемого решения, архитектурный подход к его построению, причины для его выбора, а также особенности применения в данной работе.

# Глава 2. Проектирование системы

В этой главе описаны функциональные характеристики данного решения, его архитектура, как высокоуровневая, так и архитектура каждой части сервиса по отдельности. // TODO

## Высокоуровневая архитектура систем

Проектирование высокоуровневой архитектуры системы исходило из потребностей и принципов организации работы её целевых клиентов – компаний, чей бизнес так или иначе связан с управлением большим количеством транспортных средств. Для такого рода компаний свойственна следующая структура распределения ролей:

* Водители – сотрудники, управляющие транспортным средством и выполняющие задания, поступающие к ним от операторов системы.
* Операторы – сотрудники, которые отслеживают и координируют работу водителей
* Администраторы – операторы, с расширенными правами по настройке и управлению системой

Такое распределение ролей и реализовано в данной системе.

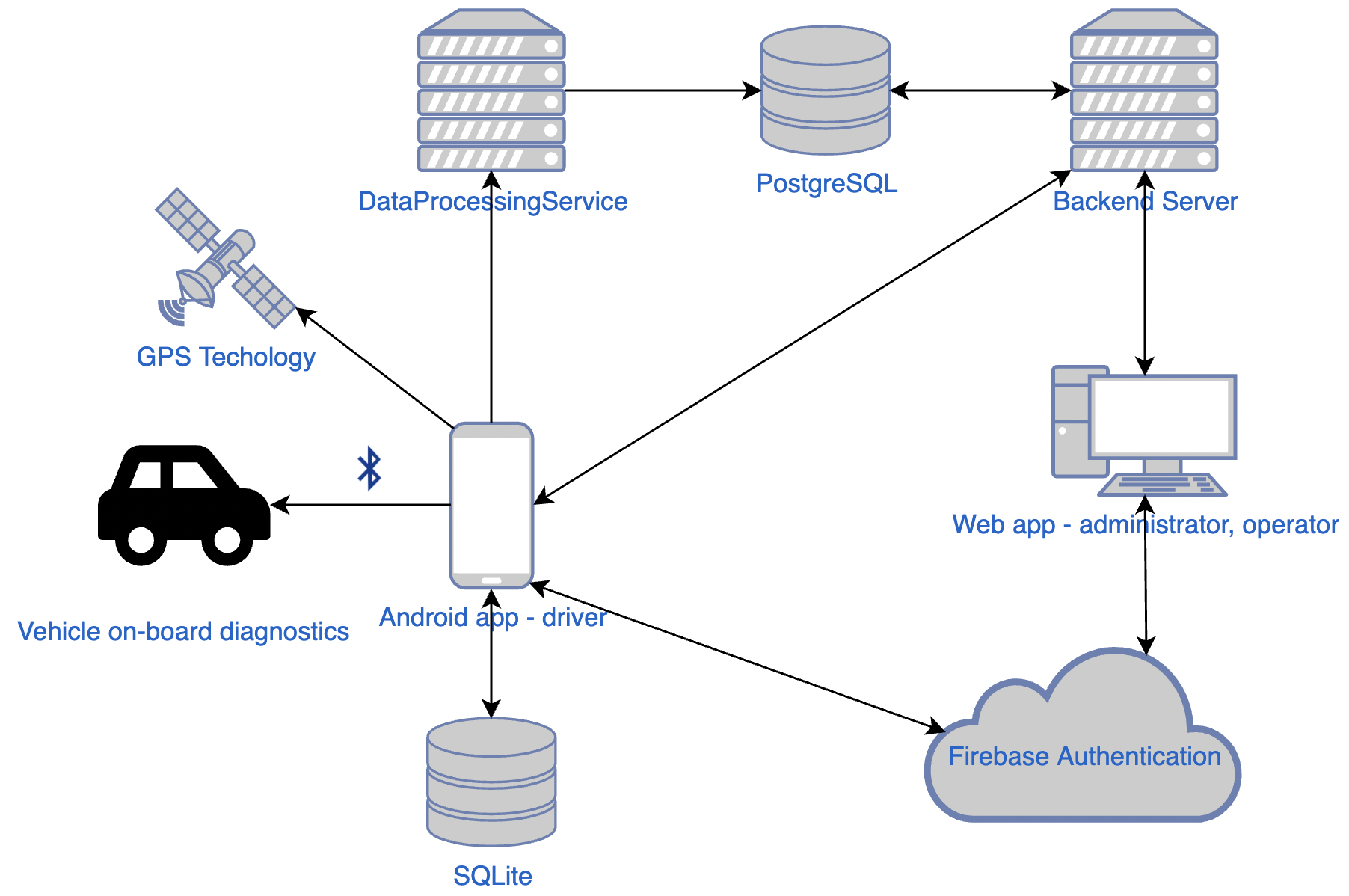


Рисунок 4. Высокоуровневая архитектура системы

Функционал системы разделен на 4 основные части:

* Android приложение
  + Предназначено для пользователей с ролью «Водитель»
  + Позволяет получать данные автомобиля, записывать их в локальную базу данных и передавать в Сервис обработки данных
  + Позволяет управлять задачами, назначенными на водителя
  + Позволяет общаться с операторами и администраторами системы по конкретным задачам или вне рамок задач
* Сервис обработки данных
  + Предоставляет функционал по записи данных, полученных с мобильных устройств, в базу данных
  + Отделен от бэкенд сервера в целях уменьшения нагрузки на него
* Клиентская часть web-приложения
  + Предназначено для пользователей с ролями «Оператор» и «Администратор»
  + Позволяет управлять сущностями пользователей, транспортных средств и задач
  + Позволяет отслеживать транспортные средства и траекторию их движения на карте
  + Позволяет анализировать работу автопарка и его сотрудников с помощью аналитических отчетов
  + Позволяет общаться с водителями по конкретным задачам или вне рамок задач
* Серверная часть
  + Предоставляет API для работы с базой данных: получение, управление, агрегирование сущностями БД
  + Управляет схемой БД с помощью механизма миграций
  + Хранит загруженные пользователями изображения в файловой системе

// TODO

Авторизация во всех частях системы реализована с помощью сервиса Firebase Authenthication. Сервис «из коробки» предоставляет широкие возможности для авторизации: различные варианты авторизации (телефон, почта, Google аккаунт, …), готовый пользовательский интерфейс, подтверждение регистрации по (телефону, почте)

## Функциональные характеристики

### Android приложение

* Авторизация по почте для пользователей с ролью «Водитель»
* Получение данных о местоположении автомобиля:
  + Географические координаты устройства. Координаты определяются либо с помощью системы GPS, при включенной функции «GPS» на мобильном устройстве, либо с помощью вышек мобильной связи при отключенной функции «GPS»
* Получение данных от Bluetooth-трекера, подключенного к бортовому компьютеру автомобиля через OBD-**||** порт:
  + Общий пробег, км
  + Текущий пробег с последнего сброса пробега, км
  + Количество ошибок двигателя системы CheckEngine
  + Коды непостоянных ошибок двигателя системы CheckEngine
  + Коды постоянных ошибок двигателя системы CheckEngine
  + Коды ошибок двигателя системы CheckEngine
  + Количество оборотов двигателя
  + Абсолютная загрузка двигателя
  + Загрузка двигателя
  + Уровень топлива, %
  + Отношение воздух/топливо
  + Температура охлаждающей двигатель жидкости
  + Температура забора воздуха
  + Температура окружающего воздуха
  + Текущая скорость автомобиля, км/ч
* Запись данных о состоянии автомобиля (местоположение, данные с Bluetooth-трекера) в локальную СУБД *(см. терминологию)*
  + Интервал записи данных в локальную БД определяется глобальными для компании настройками
* Механизм передачи данных в Сервис обработки данных из локальной СУБД при наличии интернет-соединения
  + Интервал передачи данных в Сервис обработки данных определяется глобальными для компании настройками
* Чат с пользователями приложения (текстовые сообщения и изображения) в режиме реального времени и push-уведомлениями при получении сообщения
* Управление задачами:
  + Получение назначенных на водителя задач
  + Фильтрация задач по статусам
  + Смена статуса задачи
  + Комментирование задачи (текстовые сообщения и изображения)

### Сервис обработки данных

* Реализация API для получения данных от Android приложения и записи их в БД

### Клиентская часть web-приложения

* Авторизация по почте для пользователей с ролями «Администратор» и «Оператор»
* Визуализация данных о местоположении и передвижениях транспортных средств на карте с возможностью фильтрации по временному промежутку и транспортному средству
* Построение аналитических отчетов с возможностью фильтрации по временному промежутку и транспортному средству:
  + Отчет по всем данным полученным с автомобиля
  + Отчет по времени работы автомобиля и конкретных водителей
  + Реализовано в виде таблицы на web-странице, каждую колонку можно сортировать по убыванию и возрастанию значения
* Чат с пользователями приложения (текстовые сообщения и изображения) с обновлениями в режиме реального времени и push-уведомлениями при получении сообщения
* CRUD операции *(см. терминологию)* над сущностями задач с возможностью комментирования задач (текстовые сообщения и изображения) и назначения исполнителя задачи – пользователя с ролью «Водитель»
* Функционал по смене пароля для текущего пользователя
* Для роли «Администратор»:
  + CRUD операции над сущностями пользователей
  + CRUD операции над сущностями транспортных средств с возможностью привязки водителей к транспортному средству
  + Настройка глобальных для компании параметров системы (частота записи данных в мобильную СУБД, частота их синхронизации с сервисом обработки данных)
* Для роли «Оператор»:
  + Просмотр сущностей пользователей
  + Просмотр сущностей транспортных средств с возможностью привязки водителей к транспортному средству

### Серверная часть

* Предоставляет API для работы с базой данных: получение, управление, агрегирование сущностями БД
* Управляет схемой БД с помощью механизма миграций
* Хранит загруженные пользователями изображения в файловой системе

## Информационная модель и базы данных

Рисунок 5. Схема базы данных

Для работы системы используется реляционная SQL база данных. Схема базы данных представлена рисунком 5, описание ее таблиц и полей таблицей 2.

Кроме основной базы данных система использует промежуточные – базы данных, расположенных локально на Android-устройствах. Схема базы данных идентична таблице vehicle\_data основной БД. Локальная БД предназначена для записи данных о состоянии автомобиля, перед их отправкой в Сервис обработки данных – такой подход используется для того, чтобы избежать потери данных при отсутствии интернет-соединения у мобильного приложения.

Система также использует хранение изображений в файловой системе серверной части. Такой подход связан с тем, что хранение файлов в базе данных сильно нагружает все операции внутри нее. Поэтому в базе данных хранятся только названия файлов, расположенных в файловой системе сервера.

Таблица 2

Описание таблиц базы данных

|  |  |  |
| --- | --- | --- |
| **Таблица** | **Поле** | **Описание** |
| **company** | **–** | **Компании** |
|  | id | Уникальный идентификатор |
|  | name | Название |
| **company\_settings** | **–** | **Настройки системы в рамках компании** |
|  | company\_id | Связь с таблицей company |
|  | android\_interval\_recording | Частота записи данных Android-приложения в локальную базу данных, мс |
|  | android\_interval\_synchronization | Частота передачи данных Android-приложения в Сервис обработки данных, мс |
| **employee** | **–** | **Сотрудники** |
|  | id | Уникальный идентификатор – связан с уникальным идентификатором сервиса Firebase Authenthication |
|  | company\_id | Связь с таблицей company |
|  | role\_id | Связь с таблицей role |
|  | first\_name | Имя |
|  | last\_name | Фамилия |
|  | email | Почтовый адрес |
| **role** | **–** | **Справочник ролей: оператор, администратор, водитель** |
|  | id | Уникальный идентификатор |
|  | name | Название роли |
| **vehicle\_driver\_link** | **–** | **Связь водителей и транспортных средств** |
|  | id | Уникальный идентификатор |
|  | vehicle\_id | Связь с таблицей vehicle |
|  | driver\_id | Связь с таблицей employee – водитель |
|  | start\_date | Дата привязки водителя к автомобилю |
|  | end\_date | Дата отвязки водителя от автомобиля |
| **vehicle** | **–** | **Транспортные средства** |
|  | id | Уникальный идентификатор |
|  | company\_id | Связь с таблицей company |
|  | name | Название |
|  | number | Номер |
|  | model | Модель |
|  | production\_year | Год производства |
| **vehicle\_data** | **–** | **Данные транспортного средства** |
|  | id | Уникальный идентификатор |
|  | vehicle\_id | Связь с таблицей vehicle |
|  | driver\_id | Связь с таблицей employee – водитель |
|  | datetime | Дата и время записи данных |
|  | latitude | Геокоордината – широта |
|  | longitude | Геокоордината – долгота |
|  | distance\_mil\_control | Общий пробег, км |
|  | distance\_since\_cc\_control | Текущий пробег с последнего сброса пробега, км |
|  | dtc\_number | Количество ошибок двигателя системы CheckEngine |
|  | pending\_trouble\_codes | Коды непостоянных ошибок двигателя системы CheckEngine |
|  | permanent\_trouble\_codes | Коды постоянных ошибок двигателя системы CheckEngine |
|  | trouble\_codes | Коды ошибок двигателя системы CheckEngine |
|  | rpm\_engine | Количество оборотов двигателя |
|  | absolute\_load | Абсолютная загрузка двигателя |
|  | load | Загрузка двигателя |
|  | level\_fuel | Уровень топлива, % |
|  | air\_fuel\_ratio | Отношение воздух/топливо |
|  | engine\_coolant\_temperature | Температура охлаждающей двигатель жидкости |
|  | air\_intake\_temperature | Температура забора воздуха |
|  | ambient\_air\_temperature | Температура окружающего воздуха |
|  | speed | Текущая скорость автомобиля, км/ч |
| **work\_task** | **–** | **Задачи** |
|  | id | Уникальный идентификатор |
|  | driver\_id | Связь с таблицей employee – водитель |
|  | operator\_id | Связь с таблицей employee – оператор |
|  | name | Название задачи |
|  | create\_date | Дата создания |
|  | due\_date | Срок окончания |
|  | status\_id | Связь с таблицей work\_task\_status |
| **work\_task\_status** | **–** | **Справочник статусов задач: создано, в работе, завершено, закрыто** |
|  | id | Уникальный идентификатор |
|  | name | Название статуса |
| **work\_task\_comment** | **–** | **Комментарии к задаче** |
|  | id | Уникальный идентификатор |
|  | company\_id | Связь с таблицей company |
|  | author\_id | Связь с таблицей employee – автор |
|  | task\_id | Связь с таблицей work\_task |
|  | text | Текст сообщения |
|  | datetime | Дата отправки |
|  | type | Тип сообщения: text, photo |
|  | attachment\_name | Названия файла с прикрепленным изображением, расположенного в файловой системе серверной части |
| **chat\_message** | **–** | **Сообщения, отправленные в чате** |
|  | id | Уникальный идентификатор |
|  | company\_id | Связь с таблицей company |
|  | sender\_id | Связь с таблицей employee – отправитель |
|  | receiver\_id | Связь с таблицей employee – получатель |
|  | text | Текст сообщения |
|  | datetime | Дата отправки |
|  | unread | Статус сообщения: прочитано, не прочитано |
|  | type | Тип сообщения: text, photo |
|  | attachment\_name | Названия файла с прикрепленным изображением, расположенного в файловой системе серверной части |
| **\_\_EFMigrationsHistory** | **–** | **Техническая таблица, управляемая Entity Framework, хранит миграции схемы базы данных** |
|  | MigrationId | Уникальный идентификатор миграции |
|  | ProductVersion | Версия схемы |

## Проектирование сервиса обработка данных

Сервис обработки данных предоставляет только один API метод для вызова снаружи – метод, который принимает и записывает в БД массив данных о состоянии автомобиля (таблица vehicle\_data).

## Проектирование мобильного приложения

### Общая архитектура

При проектировании мобильного приложения использовалась стандартная Model View Controller (MVC) архитектура.

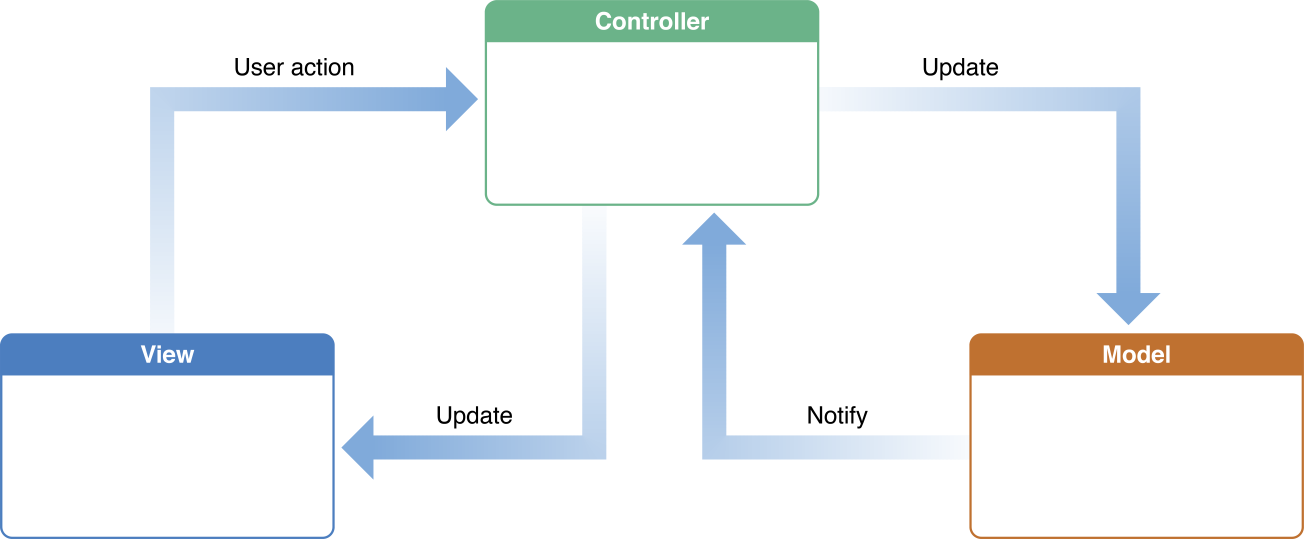


Рисунок 6. Архитектура MVC

В MVC существуют 3 слоя:

1. Model – представляет сущности данных.
2. View – отвечает за отображение на экране и обработку действий пользователя
3. Controller – посредник между Model и View. С помощью него View узнает об изменениях в Model и наоборот.

В Android-приложении в качестве моделей выступают классы, каждый из который представляет таблицу в базе данных и имеет аналогичные поля. Единственная логика, которая содержится в моделях связана с парсингом модели из JSON формата и наоборот – приведения модели в JSON формат. View представляют из себя XML файлы разметки, где отражена логика отображения моделей в UI. Контроллеры представляют из себя Activity и FragmentActivity, в них реализована вся основная бизнес-логика и взаимодействие между View и Model.

Отдельные вспомогательные утилиты были вынесены в пакет util и переиспользовались в различных контроллерах.

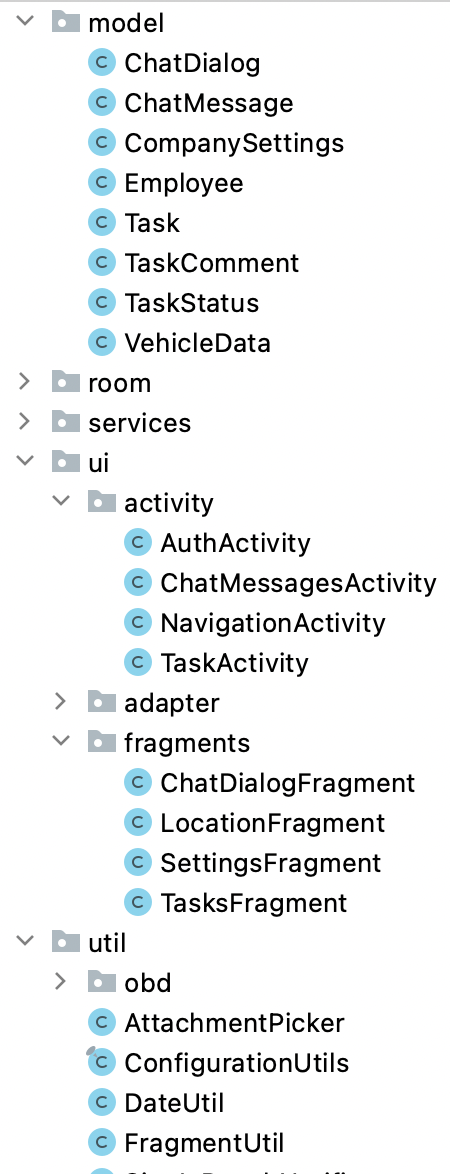


Рисунок 7. Часть распределенных по соответствующим пакетам классов Android-приложения

### Взаимодействие с API

Для взаимодействия с API серверной части были выделены Api контроллеры. В пакете API находятся классы, реализующие базовые методы по работе с API, а также интерфейсы callback (функций выполняемых при получении ответа от API). В пакете api.controller находятся классы, реализующие методы работы с API методами серверной части, разбитые по сущностям, с которыми они работают.  
Все перечисленные классы реализуют паттерн проектирования «Одиночка» [], что позволяет не создавать их для каждой отдельной Activity, а переиспользовать уже созданный экземпляр класса.

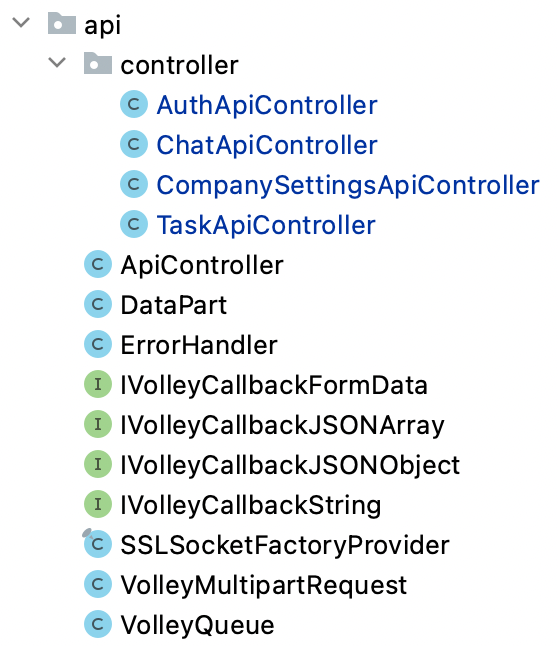


Рисунок 8. Классы для взаимодействия с API серверной части

### Навигация

Основная навигация по страницам происходит по нажатию ссылок в нижнем меню навигации.



Рисунок 9. Нижнее меню навигации

В приложении существуют следующие отдельные страницы:

Таблица 3

Описание страниц клиентского веб приложения

|  |  |
| --- | --- |
| **Страница** | **Описание** |
| Авторизации | Страница для входа в приложение |
| Сбор данных | Отображает данные привязанного к текущему сотруднику автомобиля, кнопку для старта механизма сбора данных, время с начала старта сбора данных |
| Задачи | Отображает список задач, назначенных на сотрудника, с возможностью переключения фильтра между статусами задач |
| Чат | Отображает переписки с сотрудниками компании. Позволяет получать и отправлять текстовые сообщения и изображения, обновляется в режиме реального времени |
| Настройки | Отображает данные текущего пользователя приложения. Позволяет настроить параметры «с OBD» или «без OBD» и указать имя Bluetooth трекера, к которому требуется подключиться. |

### Запись данных автомобиля

Для реализации функционала получения данных о состоянии автомобиля, который должен выполняться в фоновом режиме были использованы механизмы ForegroundServices [] и TimerTask [].

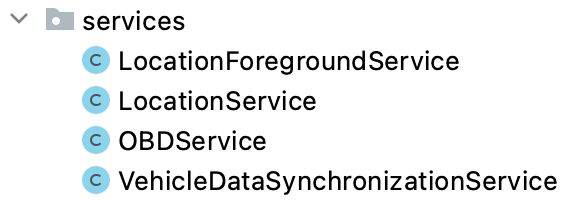


Рисунок 10. Сервисы

При старте сбора данных о состоянии автомобиля в фоновом «липком» [] режиме запускается LocationForegroundService. Основные задачи сервиса: реализация логики запуска сервиса получения геоданных (LocationService), сервиса получения данных с бортового компьютера автомобиля (OBDService) и сервиса для синхронизации записанных данных с Сервисом обработки данных (VehicleDataSynchronizationService); получение данных о геолокации и данных с Obd, их запись в локальную базу данных.

В приложении предусмотрены два режима работы, настраиваемые пользователем:

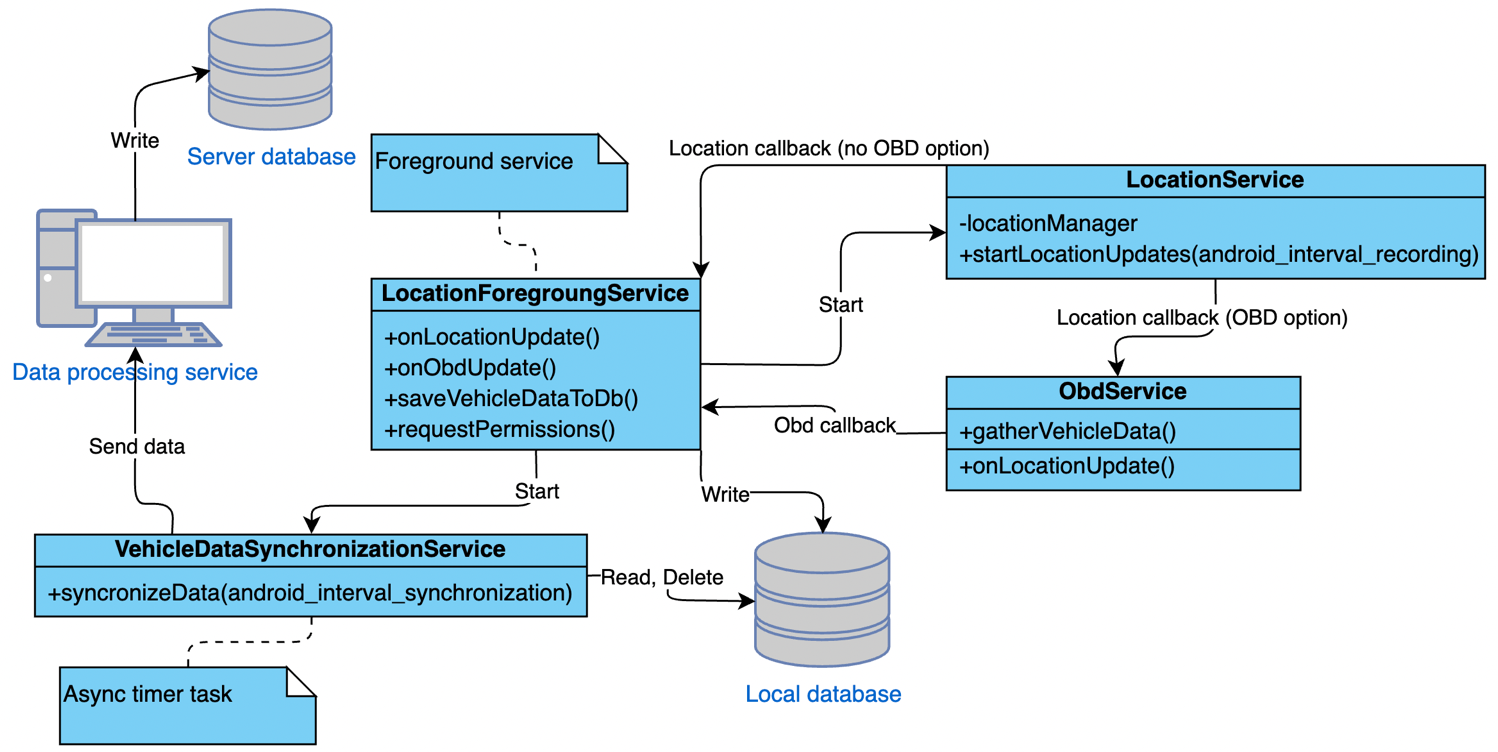
* «без OBD» – в систему поступают только данные, касающиеся геолокации Android-устройства с привязкой ко времени, транспортному средству и водителю
* «c OBD» – в систему поступают данные, касающиеся геолокации Android-устройства с привязкой ко времени, транспортному средству и водителю + данные с бортового компьютера автомобиля

Рисунок 11. Схема работы компонентов, отвечающих за запись и передачу данных автомобиля

LocationService – сервис, который раз в время, регулируемое глобальным для компании параметров android\_interval\_recording, собирает данные о текущей геолокации и с помощью callback функции передает их своему подписчику – locationManager

При настроенном режиме «без OBD» подписчиком LocationService выступает сам LocationForegroundService (применяется паттерн проектирования «Обозреватель» [] и «Шаблонный метод» []). При получении данных о геолокации он соединяет их с данными по текущему времени, транспортному средству и пользователю и записывает их в локальную базу данных.

При настроенном режиме «с OBD» подписчиком LocationService является OBD Service. При получении данных о геолокации он запрашивает данные с бортового компьютера автомобиля, соединяет их и передает в подписанный на него LocationForegroundService, где они соединяются с данными по текущему времени, транспортному средству и пользователю и записываются в локальную базу данных.

Данные из локальной базы данных собираются VehicleDataSynchronizationService с ограничением в 50 записей раз в время регулируемое глобальным для компании параметров android\_interval\_synchronization и передаются в Сервис обработки данных. При успешной передаче данных, соответствующие записи очищаются из локальной базы данных.

## Bluetooth OBD трекер

Для получения сведений с бортового компьютера система использует Bluetooth OBD трекер *(см. терминологию)* ELM327.

ELM327 — микросхема, преобразующая ряд протоколов, используемых в диагностических шинах автомобилей, в протокол RS-232, разработанная компанией Elm Electronics.



Рисунок 12. Bluetooth OBD трекер ELM327

  
Рисунок 13. OBD-|| порт

OBD трекер ELM327 совместим с автомобилями, которые оборудованы специальным диагностическим разъемом OBD2 16 пин. Такие авто впервые стали выпускаться в США в 1996 году, в остальных странах оснащение диагностическим разъемом произошло намного позднее. Автомобильные концерны на территории Японии комплексно оборудовали все типы производимых машин начиная с 2000 г. В европейских государствах машины, работающие на бензине, были оборудованы в 2001 г., дизельные с 2004 г. Китайские авто оснастили в 2008, 2009 годах соответственно. Отечественные автозаводы начали укомплектовывать 16 контактными разъемами модели выпускаемые с 2010 года. До 2010 на Российских моделях устанавливали 12 pin GM (VAG) разъем, для подключения к таким интерфейсам потребуются переходники.

Получение данных с OBD трекера происходит по следующей процедуре:

* Сервис Android-приложения (ObdService) включает Bluetooth, предварительно запрашивая на это разрешение у пользователя
* По заданному в настройках имени, находит Obd трекер и подключается к нему
* ObdService передает Obd трекеру конфигурационные команды (протокол общения, таймаут ожидания ответа и т.п.)
* ObdService получает местоположение девайса через callback и передает Obd трекеру команды для получения данных автомобиля
* ObdService считывает ответы и приводит их к нужным типам
* ObdService передает собранные данные в callback-функцию LocationForegroundService

## Проектирование клиентского веб-приложения

### Общая архитектура

Клиентское веб-приложение представляет из себя отдельное приложение, взаимодействующее с серверной частью через API.

Архитектура и принцип взаимодействия с API аналогичен реализованному в Android-приложении. В качестве моделей выступают классы, каждый из который представляет таблицу в базе данных и имеет аналогичные поля. Единственная логика, которая содержится в моделях связана с парсингом модели из JSON формата и наоборот – приведения модели в JSON формат. View представляют из себя наборы jsx тегов, в которых отражена логика отображения моделей в UI. Контроллерами являются страницы и отдельные компоненты, в них реализована вся основная бизнес-логика и взаимодействие между View и Model.

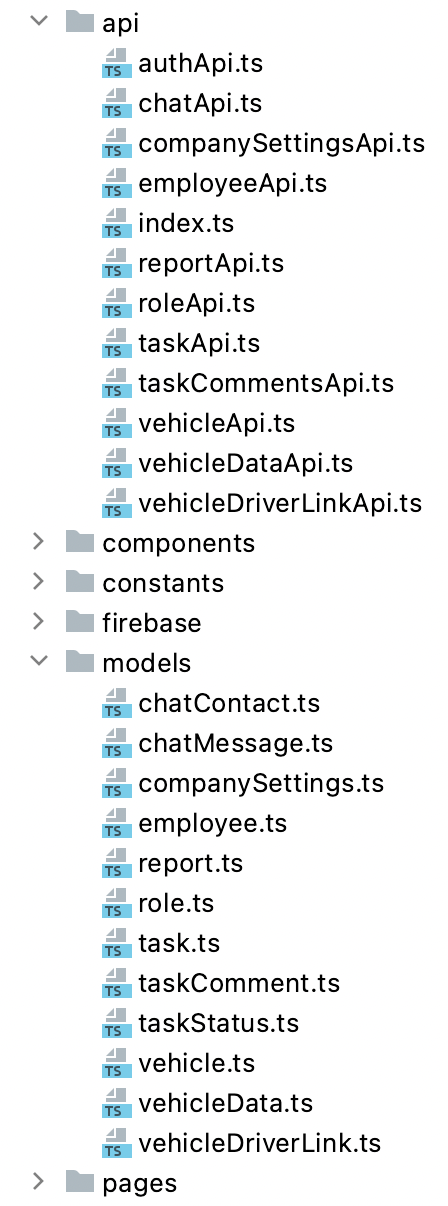


Рисунок 14. Часть распределенных по соответствующим пакетам файлов клиентского веб-приложения

### Навигация

Основная навигация по страницам происходит по нажатию ссылок в верхнем меню навигации.



Рисунок 15. Верхнее меню навигации клиентского веб-приложения

В приложении существуют следующие отдельные страницы:

Таблица 4

Описание страниц клиентского веб приложения

|  |  |  |
| --- | --- | --- |
| **Страница** | **Путь** | **Описание** |
| Авторизации | /signIn | Страница для входа в приложение |
| Главная | /home | Отображает карту с местоположением транспортных средств, траекторией их движения. Позволяет настраивать фильтра по транспортному средству и временному интервалу |
| Доска задач | /taskboard | Отображает доску, подобную доске Trello []. Позволяет создавать задачи, редактировать их, комментировать и удалять. |
| Чат | /chat | Отображает переписки с сотрудниками компании. Позволяет получать и отправлять текстовые сообщения и изображения,, обновляется в режиме реального времени |
| Отчеты | /reports | Позволяет настраивать фильтр (название отчета, временной интервал, транспортное средство) для формирования отчетов в виде таблиц |
| Настройки компании | /company\_settings | Доступна только для роли «Администратор», отображает и позволяет устанавливать глобальные для компании параметры |
| Аккаунт | /account | Отображает текущего пользователя, позволяет сменить пароль для входа в систему |

Кроме этого, для главной страницы реализовано переключение бокового меню. При переключении бокового меню Главная страница не переключается и на ней остается карта с транспортными средствами и соответствующие фильтры.

Таблица 5

Описание боковых меню Главной страницы

|  |  |
| --- | --- |
| **Боковое меню** | **Описание** |
| Сотрудники | Отображает список сотрудников, отфильтрованных по ролям. Пользователи с ролью «Администратор» могут проводить CRUD операции над сущностями сотрудников |
| Транспортные средства | Отображает список транспортных средств. Позволяет привязывать к автомобилю сотрудников с ролью «Водитель». Пользователи с ролью «Администратор» могут проводить CRUD операции над сущностями транспортных средств |
| Задачи | Отображает список задач, отфильтрованных по статусу. Позволяет создавать задачи, редактировать их, комментировать и удалять. |

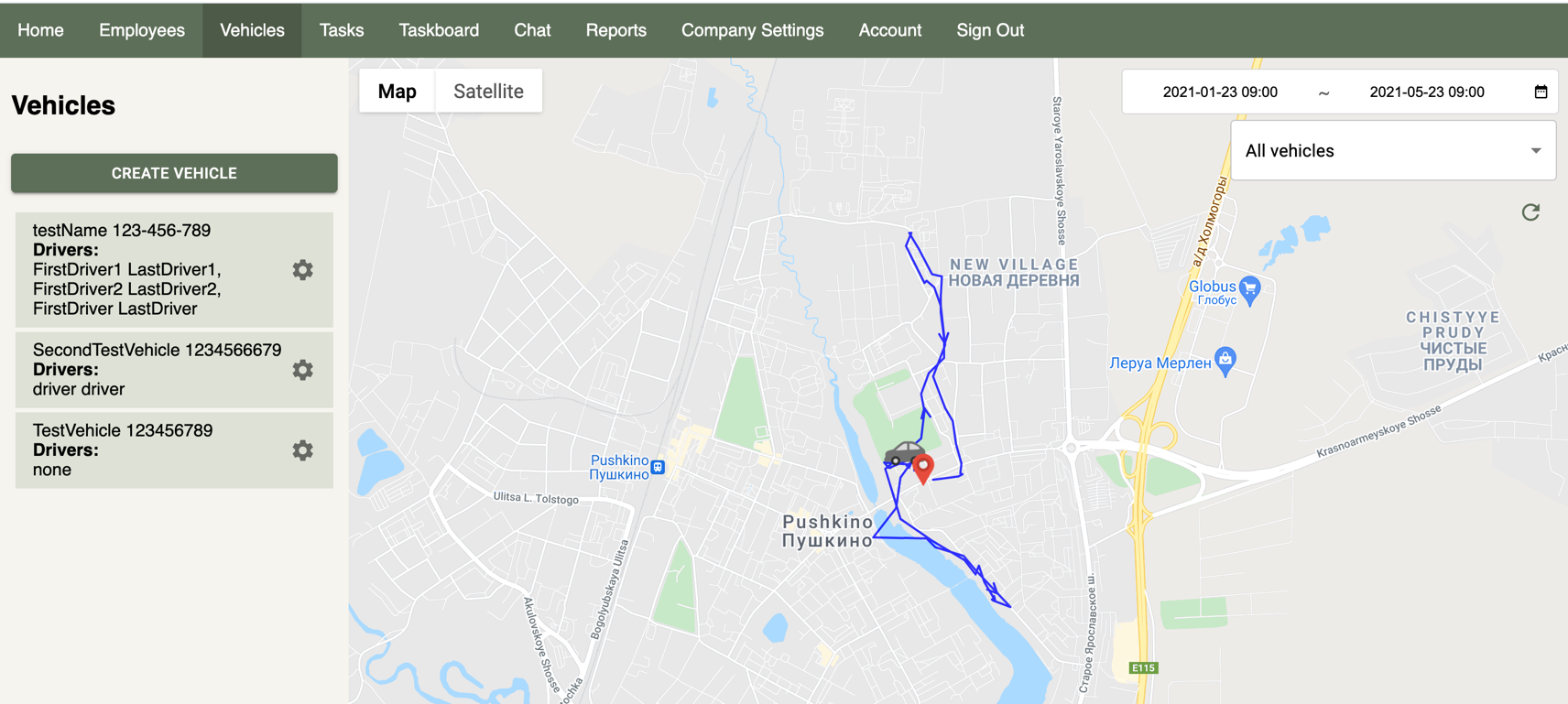


Рисунок 16. Главная страница с включенным боковым меню «Транспортные средства»

## Проектирование серверной части

### Общая архитектура

Серверная часть представляет из себя web-API, построенная по принципам RESTful API []. Реализация принципов RESTful API означает, что сервер функционирует по принципу закрытого API; взаимодействие с клиентами происходит независимо от технологий, на которых реализованы клиенты; сервер реализует обработку стандартных REST методов: GET, POST, PUT, DELETE и возвращает в ответ стандартные REST коды (200, 400, 404, 409 и другие); сервер не запоминает состояния, т.е. предыдущих запросов пользователя, но способен кэшировать и отдавать готовые ответы на GET запросы; все запросы клиента и ответы передаются в формате JSON [], за исключением изображений, которые передаются в формате image в атрибуте multipart/form-data, передача отдельных файлов не с помощью JSON запросов не противоречит положениям RESTful архитектуры; наличие промежуточного слоя, который отвечает за перенаправление запросов и безопасность. Кроме этого, сервер реализует обратный канал для связи с клиентами, заинтересованными в обновлении чата в режиме реально времени – использование отдельного функционала, которые не соответствует принципам REST, также не противоречит положениям RESTful архитектуры.

Главными задачами, которые выполняет серверная часть, являются

* Работа с базой данных
  + Поддержка механизма миграций для обеспечения сохранения изменений в схеме базы данных
  + Получение, агрегация и изменение данных базы данных
* Работа с файловым хранилищем:
  + Сохранение изображений (с их предварительным сжатием)
  + API для получения сохраненных изображений по их имени
* Обеспечение обратного канала связи для обновления чатов на клиентах в режиме реального времени

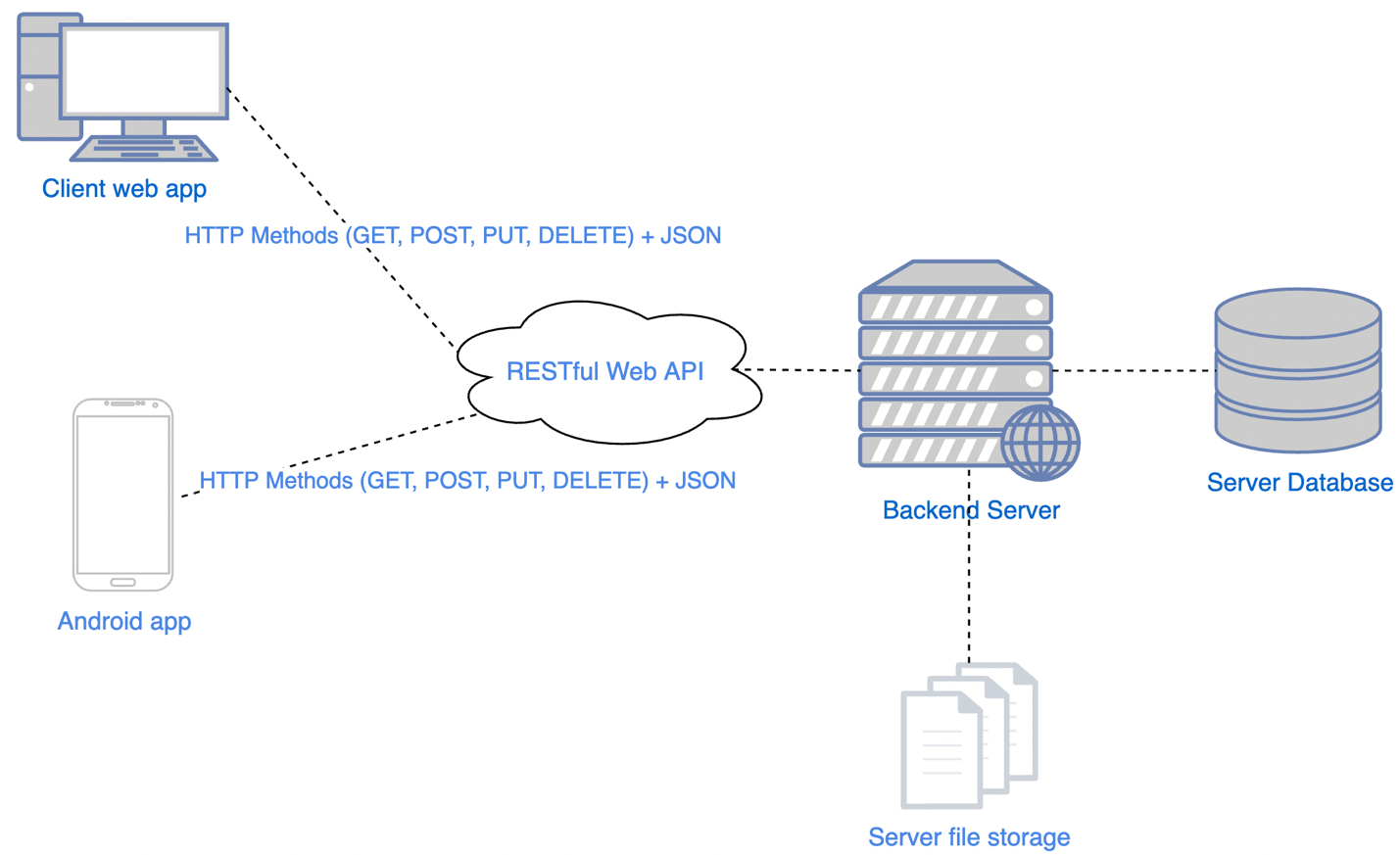


Рисунок 17. Принцип взаимодействия серверной части с клиентами

Рисунок 18. Часть классов серверной части распределенных по соответствующим пакетам

API методы реализованы в контролерах и разбиты по сущностям, с которыми они оперируют.

### Взаимодействие с базой данных

Для непосредственной работы с базой данных контролеры используют сервисы для работы с базой данных, которые расположены в пакете Services.Database. Сервисы также разбиты по сущностям, с которыми они оперируют, в них реализована логика работы с базой данных.

Все сервисы для работы с базой данных наследуются от базового generic [] сервиса с указанием класса сущности объектов базы данных, с которыми они работают. Это позволяет избежать лишнего дублирования кода, так как базовый сервис уже реализует стандартные CRUD операции для работы с базой данных.

Все сервисы для работы с базой данных внедряются через механизм «Внедрения зависимостей» (Dependency Injection) [], реализуя принцип «Инверсии зависимостей» (Inversion of Control) []. Это позволяет упростить код и облегчить замену сервисов. Явно указав в контролере необходимые для его работы сервисы, можно отдать управление жизненным циклом сервисов в «руки» Dependency Injection.

### Сохранение изображений в файловой системе

Хранение изображений напрямую в базе данных сильно увеличивает нагрузку на нее, поэтому было принято решение сохранять изображения в файловой системе серверной части в папке «wwwroot», а в базе данных хранить имя файла в указанной папке.

Кроме этого, чтобы не слишком занимать место в файловой системе серверной части все загруженные изображения проходят процесс сжатия.  
Для сжатия изображения используется следующий алгоритм

* Если расширение изображения по высоте или широте больше 600 пикселей
  + Делим максимальную по числу пикселей сторону на 600, получаем делитель
  + Создаем новое изображение с высотой = высота / делитель и широтой = широта / делитель, т.е. уменьшаем число пикселей обоих сторон в число раз, в которое максимальная сторона превышает ограничение в 600 пикселей
* Иначе – загружаем как есть

Такой алгоритм позволяет в разы уменьшить вес картинки, не изменяя ее изначальных пропорций

## Выводы по главе

В данной главе были разобраны и обоснованы принципы проектирования архитектуры проекта в целом и его отдельных частей в частности. // TODO

# Глава 3. Разработка системы

В этой главе представлены и описаны инструменты и методы, использованные в процессе этапа реализации всех компонент системы. // TODO

## Сервис обработки данных

Сервис обработки данных реализован на языке C# 8.0 и фреймворке ASP .NET Core 3.1.

Сервис использует Dapper ORM [] для работы с базой данных.

## Android-приложение

Android приложение реализовано на языке Java 8

## Интеграция с Bluetooth-трекером

// TODO

## Клиентское Веб-приложение

// TODO

## Серверная часть

// TODO

## Выводы по главе

В данной главе был описан процесс разработки программного продукта на основе результатов проектирования полученных в Главе 2 настоящей работы. // TODO

# Заключение

// TODO

# Приложение А (Техническое задание)

Представлено отдельным документом «Техническое задание».

# Источники

1. Monika Jain, Praveen Kumar, Priya Singh, Chhavi Narayan Arora, Ankita Sharma. «Detection of Over Speeding Vehicles on Highways». [Электронный ресурс] / Режим доступа: <https://ijcsmc.com/docs/papers/April2015/V4I4201599a39.pdf> , свободный (дата обращения: 18.02.2021).
2. Pratiksha Bhuta, Karan Desai, Archita Keni “Alcohol Detection and Vehicle Controlling”. Апрель 2015. [Электронный ресурс] / Режим доступа: <http://www.ijetajournal.org/volume-2/issue-2/IJETA-V2I2P14.pdf> , свободный (дата обращения: 18.02.2021).
3. Vehicle Tracking Systems Market Size, Share & Trends Analysis Report By Vehicle Type, By End-use, By Technology, By Type, By Component, By Region, And Segment Forecasts, 2020 – 2027. Июль 2020. [Электронный ресурс] / Режим доступа: <https://www.grandviewresearch.com/industry-analysis/vehicle-tracking-systems-market>, свободный. (дата обращения: 17.11.20).
4. Мониторинг транспорта и навигация (рынок России). [Электронный ресурс] / Режим доступа: <https://www.tadviser.ru/index.php/%D0%A1%D1%82%D0%B0%D1%82%D1%8C%D1%8F:%D0%9C%D0%BE%D0%BD%D0%B8%D1%82%D0%BE%D1%80%D0%B8%D0%BD%D0%B3_%D1%82%D1%80%D0%B0%D0%BD%D1%81%D0%BF%D0%BE%D1%80%D1%82%D0%B0_%D0%B8_%D0%BD%D0%B0%D0%B2%D0%B8%D0%B3%D0%B0%D1%86%D0%B8%D1%8F_(%D1%80%D1%8B%D0%BD%D0%BE%D0%BA_%D0%A0%D0%BE%D1%81%D1%81%D0%B8%D0%B8)>, свободный. (дата обращения: 17.11.20).
5. 8-е ежегодное исследование рынка мониторинга транспорта РФ. [Электронный ресурс] / Режим доступа: <https://gurtam.com/docs/Omnicomm_Navitoring_2018.pdf>, свободный (дата обращения: 17.11.20)
6. Wialon. [Электронный ресурс] / Режим доступа: <https://gurtam.com/ru/wialon>, свободный (дата обращения 17.11.20)
7. АвтоГраф. [Электронный ресурс] / Режим доступа: <https://glonassgps.com/>, свободный (дата обращения 17.11.20)
8. Omnicomm. [Электронный ресурс] / Режим доступа: <https://www.omnicomm.ru/>, свободный (дата обращения 17.11.20)
9. СКАУТ 365. [Электронный ресурс] / Режим доступа: <https://scout-gps.ru/programmnoe-obespechenie/scout365/>, свободный (дата обращения 17.11.20)
10. OBD-**||** Bluetooth adapter [Электронный ресурс] / Режим доступа: <https://market.yandex.ru/search?text=obd%202%20%20bluetooth&cpa=0&cvredirect=2&how=aprice&onstock=0&local-offers-first=0>, свободный (дата обращения 22.04.21)