**ООО «АМ-ИНТЕЛЛЕКТУАЛЬНЫЕ ТЕХНОЛОГИИ»**

|  |
| --- |
|  |

|  |  |
| --- | --- |
|  |  |

**Система МОНИТОРИНГА Парковочных мест**

**АРТИКУЛ**

**Описание ПРОГРАММЫ**

0000000

Москва 2021

АНнотация

Настоящий документ содержит сведения о функциональном назначении, структуре программного обеспечения «Система мониторинга парковочных мест», артикул , условиях его функционирования, способах доступа к интерфейсу и API системы, описание входных и выходных данных.

Настоящий документ составлен в соответствии с требованиями ГОСТ 19.402‑78 и   
ГОСТ 19.106‑78.

Содержание

[1. Общие сведения 4](#_Toc138769630)

[1.1. Обозначение и наименование программы 4](#_Toc138769631)

[1.2. Краткое наименование 4](#_Toc138769632)

[1.3. Языки программирования, на которых написана программа 4](#_Toc138769633)

[2. Функциональное назначение 5](#_Toc138769634)

[3. Описание логической структуры 7](#_Toc138769635)

[3.1. Алгоритм программы 7](#_Toc138769636)

[3.1.1. Распознавания статуса занятости парковочных мест 7](#_Toc138769637)

[3.1.2. Расчет точности распознавания статуса занятости парковочных мест 9](#_Toc138769638)

[3.1.3. Алгоритм поиска парковочных мест, внутри заданной географическими координатами области 9](#_Toc138769639)

# Общие сведения

## Обозначение и наименование программы

Полное наименование программы: Система мониторинга парковочных мест, артикул.

## Краткое наименование

Краткое наименование программы: ПАРКОВКИ.

## Языки программирования, на которых написана программа

Основным языком программирования является GoLang.

# Функциональное назначение

Программа предназначена для автоматизации процессов мониторинга парковочного пространства общего пользования г. Москвы, обеспечивающих реализацию следующих функций:

выполнение аутентификации субъектов доступа (посредством механизма ввода комбинации идентификатора учетной записи (логина) и пароля пользователя) через СУДИР;

выполнение авторизации субъектов доступа;

управление субъектами доступа;

управление правами субъектов доступа;

графический интерфейс пользователя Технологического портала, который состоит из набора интерактивных веб-страниц;

интерактивная карта с нанесенными на нее значками, обозначающими все доступные для ПО видеокамеры и зарегистрированные в ПО парковочные пространства;

интерфейс детализированного представления каждого парковочного пространства, где отображен последний снимок, полученный с выбранной Видеокамеры, с автоматически нанесенными на него областями проекций распознанных ПО парковочных мест и их текущим статусом в виде особой цветовой окраски для каждого и статусов (свободно/занято/статус неизвестен);

интерфейс интерактивного проектирования размещения парковочных мест на карте с привязкой к камере и регистрацией географических координат каждого парковочного места;

интерфейс реестра парковочных пространств и парковочных мест;

интерфейс интерактивного проектирования и отображения границ и географических координат парковочных мест каждого парковочного пространства, обеспечивающего возможность корректировки парковочных мест:

* + - * добавление парковочных мест;
      * задание границ парковочных мест;
      * удаления парковочных мест;
      * редактирования географических координат парковочных мест.

интерфейс создания задач для наблюдения за парковками с привязкой к одной из камер дворового наблюдения с настройкой позиции камеры для сбора данных и выбора периода анализа занятости мест;

интерфейс просмотра списка камер, изображений с камер для настройки мониторинга парковки;

взаимодействие со шлюзом ЕЦХД для получения информации окамерах и изображений с камер;

взаимодействие с Модулем распознавания регистрационных знаков и определения характеристик транспортных средств, для получения bbox запаркованных транспортных средств;

REST API, возвращающий по запросу данные по парковочным местам, доступным для Видеокамер, установленных по УНОМ;

REST API, возвращающий по запросу данные по парковочным местам, внутри заданной географическими координатами области;

формирование еженедельных отчетов о состоянии парковочного пространства;

логирование внутренних событий и результатов операций ПО в консоль.

# Описание логической структуры

## Алгоритм программы

Основной алгоритм выполнения программы состоит из следующих этапов:

### Алгоритмы создания/сохранения разметки парковочных мест



Разово-ручное полигональное выделение мест поверх трансляции с камеры.

Под каким бы углом не было парковочное место, его можно описать четырехугольником.

Полигональная область парковки усложнит алгоритмы вычисления занятости и увеличит затраты, поэтому область парковки задается четырехугольником, скорее всего даже параллелограммом.

Возможные сценарии разметки парковочного пространства:

1. Парковочные места являются нумерованными, для каждого парковочного места с номером передаем номер и состояние "свободно/занято" во Внешнюю систему.
2. Существует территория, представляющая область интереса, для которой задается количество допустимых парковочных мест. Оператор не задает точное расположение парковочных мест. Определяется количество припаркованных автомобилей внутри области интереса, передается информация, что на определенной территории присутствуют 8 авто из 10 возможных для этой территории.

### Распознавания статуса занятости парковочных мест

 



### Методика определения точности распознавания статуса занятости парковочных мест

Должна быть описана в Регламенте оказания услуг

### Алгоритм поиска парковочных мест по УНОМ

### Алгоритм поиска парковочных мест, внутри заданной географическими координатами области

Время отклика Решения на запросы внешних ИСиР не должно превышать 1 (одной) секунды.

### Алгоритм опроса камер ЕЦХД для получения изображений с камер, наблюдаемых парковочных пространств

Частота опроса Видеокамер — один раз в 30 (тридцать) секунд.

### Алгоритм формирования еженедельных отчетов

### Алгоритм отображения на карте парковочных мест и камер в виде точек с поддержкой кластеризации при масштабировании

## Программный интерфейс для внешних систем (API ПО)

Поддерживаются представленные ниже методы API (таблица 3).

Таблица 3 – Методы API

| МЕТОД | ВЫПОЛНЯЕМОЕ ДЕЙСТВИЕ | ПОДСИСТЕМА / МОДУЛЬ |
| --- | --- | --- |
| **Получение данных по УНОМ** | | |
| GET/ | запрос с параметрами УНОМ | Подсистема сбора и обработки данных |
| **Получение данных по географическим координатам** | | |
| POST / | запрос с параметрами прямоугольной ограничивающей области с координатами | Подсистема сбора и обработки данных |

### Коды ответов

ПО возвращает корректные коды ответа HTTP в зависимости от обработки запроса (таблица 4).

Таблица 4 – Коды ответов

| КОД ОТВЕТА HTTP | НАЗНАЧЕНИЕ |
| --- | --- |
| 200 OK | Запрос успешно обработан.  Возвращается для запросов на получение информации: поиск, получение списков и отдельных записей |
| 201 Created | Объект успешно создан.  Возвращается для запросов на создание объектов |
| 400 Bad Request | Запрос не соответствует формату.  Возвращается для запросов, которые не могут быть обработаны из-за несоответствия формату взаимодействия |
| 401 Unauthorized | Запрос не авторизован.  Возвращается при попытке выполнения запроса без аутентификации или с неверной аутентификационной информацией |
| 403 Forbidden | Объект, на который ссылается запрос, не существует. |
| 404 Not Found | Объект, на который ссылается запрос, не существует.  Возвращается для запросов, имеющих ссылку на объект по идентификатору в случае отсутствия объекта с указанным идентификатором |
| 409 Conflict | Создаваемый объект существует.  Возвращается для запроса на создание объекта, который уже был создан ранее |
| 500 Internal Server Error | Ошибка ПО  Возвращается во всех случаях ошибки при выполнении запроса, если ошибка не позволила обработать запрос |

### Объекты

#### Bounding Box

Bounding Box, координаты– прямоугольник, который задаётся координатами левого верхнего угла и правого нижнего.

Пример JSON:

{

"top": 100,

"left": 20,

"bottom": 340,

"right": 260

}

### Методы программного интерфейса

##### Получение данных по УНОМ

Запрос:

GET /:id

id – идентификатор **УНОМ**.

Ответ: возвращает объект <ххх>

Массив Парковочных мест, содержащий:

* Статус парковочного места:
  + место свободно
  + место занято
  + статус неизвестен
* Место для инвалидов (да/нет)
* Место спецтехники (да/нет)
* Координаты парковочного места

##### Получение данных по географическим координатам

Запрос:

POST /

запрос с параметрами прямоугольной ограничивающей области с координатами,

Тело запроса представляет собой JSON вида:

{

• Южная широта ограничивающей области;

• Западная долгота ограничивающей области;

• Северная широта ограничивающей области;

• Восточная долгота ограничивающей области.

}

Ответ: возвращает объект <ххх>

Массив Парковочных мест, содержащий:

* Статус парковочного места
  + место свободно
  + место занято
  + статус неизвестен
* Место для инвалидов (да/нет)
* Место спецтехники (да/нет)
* Координаты парковочного места

## Используемые методы

### Объектно-ориентированный подход при описании задач

Решение задач опирается на применение объектно-ориентированной технологии их описания и реализации. Основой этой технологии служит объектная модель ПО.

Объектная модель ПО описывает структуру объектов, составляющих ПО, их свойства, операции и взаимосвязи с другими объектами.

В объектной модели отражены те понятия и объекты реального мира, которые важны для разработки. В ней отражается прежде всего прагматика разрабатываемой системы. Прагматика выражается в использовании терминологии прикладной области, связанной с использованием разрабатываемой системы.

Объектом называется понятие, абстракция или любая другая вещь с четко очерченными границами, имеющая смысл в контексте рассматриваемой прикладной проблемы.

Введение объектов преследует две цели:

* понимание прикладной задачи (проблемы);
* введение основы для реализации ее на компьютере.

Целью разработки объектной модели является выделение и описание объектов, составляющих в совокупности проектируемую систему, а также выявление и указание различных зависимостей между объектами.

### Объектно-ориентированные языки разработки

При разработке использовался современный объектно-ориентированный язык программирования GoLang.

GoLang — это многопоточный компилируемый язык (интерпретатор в экосистеме также предусмотрен, но на практике необходимости в нем нет: компиляция происходит мгновенно). По производительности Go почти не уступает C++ и в десятки раз превосходит скриптовые языки — такие, как JavaScript, Python, Ruby, PHP.. Язык имеет статическую типизацию, поддерживает полиморфизм, перегрузку операторов (в том числе операторов явного и неявного приведения типа), делегаты, атрибуты, события, переменные, свойства, обобщённые типы и методы, итераторы.

Низкая требовательность к памяти. В GoLang есть «сборщик мусора» — периодически те объекты, потребность в которых уже не возникнет, удаляются. Таким образом происходит автоматическая очистка памяти.

Простота параллельных вычислений. Язык идеально подходит для создания программ, рассчитанных на многоядерные процессоры. Многопоточность, реализуемая через так называемые горутины (go routines), которые взаимодействуют посредством каналов, позволяет вызывать несколько функций практически одновременно. Эта возможность очень актуальна при создании больших и сложных программ. Средства параллельного программирования экономят время разработчика и помогают равномерно распределять ресурсы процессора.

### Объектно-реляционная база данных

База данных выступает в качестве информационной модели предметной области, в рамках которой функционирует система.

В БД отражается специфика предметной области, особенности решаемых на ее основе задач.

Базы данных предназначены для того, чтобы обеспечивать:

* учет потребностей всех решаемых задач, моделей, всех пользователей на данном уровне управления;
* однозначность представления и понимания используемой информации на разных уровнях и объектах (за счет использования единых и согласованных словарей и классификаторов);
* адекватное отображение предметной области, но, вместе с тем, состав информации на каждом уровне управления не должен превышать его потребностей, а быть достаточным и обобщенным в необходимой степени;
* хранение (в случае необходимости):

1. картографической информации;
2. фактографической информации;
3. документальной информации;
4. мультимедийной информации;

* обслуживание пользователей за время не более допустимого с возможными дополнительными требованиями на обслуживание заданного количества пользователей;
* принцип открытости, т. е. БД должна обладать информационной полнотой и, в то же время, быть способной к структурному информационному расширению при появлении новых категорий пользователей или прикладных задач;
* по возможности соответствие реально существующим связям и объектам предметной области для уменьшения вероятности программных ошибок, связанных с неправильной интерпретацией данных, а также облегчения разработки средств ведения БД.

В качестве СУБД, отвечающей всем перечисленным выше требованиям выбрана СУБД PostgreSQL.

PostgreSQL – свободная объектно-реляционная система управления базами данных.

Работа СУБД реализована в архитектуре клиент-сервер. Рабочий сеанс Postgres включает следующие взаимодействующие процессы (программы):

* главный серверный процесс, управляющий файлами баз данных, принимающий подключения клиентских приложений и выполняющий различные запросы клиентов к базам данных. Эта программа сервера БД называется postgres;
* клиентское приложение пользователя, выполняет операции в базе данных. Поддерживается широкий спектр клиентских приложений - это может быть текстовая утилита, графическое приложение, веб-сервер, использующий базу данных для отображения веб-страниц, или специализированный инструмент для обслуживания БД.

Сервер Postgres может обслуживать одновременно несколько подключений клиентов. Для этого он запускает («порождает») отдельный процесс для каждого подключения. Можно сказать, что клиент и серверный процесс общаются, не затрагивая главный процесс postgres. Таким образом, главный серверный процесс всегда работает и ожидает подключения клиентов, принимая которые, он организует взаимодействие клиента и отдельного серверного процесса.

PostgreSQL поддерживает язык SQL и отвечает основным требованиям стандарта SQL:2011.

PostgreSQL работает с локализацией, установленной в операционной системе и отвечающей стандарту POSIX. На практике это означает возможность работы с несколькими десятками языков, в том числе и с русским языком во всех возможных кодировках: koi8-r, cp1251, iso8859-5 и UTF-8. Возможность корректной работы PostgreSQL с конкретной кодировкой зависит от корректной поддержки этой кодировки средствами самой операционной системы.

Работа с данными, хранящимися в БД, ведется на основе языка структурированных запросов SQL.

SQL (англ. Structured Query Language – «язык структурированных запросов») – декларативный язык программирования, применяемый для создания, модификации и управления данными в реляционной базе данных, управляемой соответствующей системой управления базами данных.

SQL является информационно-логическим языком, предназначенным для описания, изменения и извлечения данных, хранимых в реляционных базах данных.

При всех изменениях SQL остаётся единственным механизмом связи между прикладным программным обеспечением и базой данных.

### Стандартизация и унификация

Методы программных интерфейсов Системы (API) спроектированы с учетом требований унификации:

* для выполнения аналогичных операций должны использоваться аналогичные методы;
* должны быть унифицированы термины, используемые для описания идентичных понятий, операций и действий пользователя;
* реакция системы на действия пользователя (посредством API-ответов) должна быть типовой для каждого действия.

Использованное при создании Системы программное обеспечение и библиотеки программных кодов являются общедоступными и используются в промышленных масштабах.

### Использованные при разработке библиотеки

При разработке использованы следующие библиотеки (приведены со ссылками на их страницы в сети Интернет):

* «PostgreSQL 11.9 (Debian 11.9-1.pgdg90+1) on x86\_64-pc-linux-gnu, compiled by gcc (Debian 6.3.0-18+deb9u1) 6.3.0 20170516, 64-bit" https://www.postgresql.org/»;

## Структура программы с описанием функций составных частей и связи между ними

Общая архитектура ПО представлена ниже (рисунок 8).

Рисунок 8 – Общая архитектура ПО

## Функциональная структура ПО

Структурно ПО состоит из следующих подсистем:

## Связи ПО с другими программами

Информационное взаимодействие на сетевом уровне строится с использованием протоколов стека TCP/IP.

# Используемые технические средства

Технические средства (далее – ТС), обеспечивающие функционирование ПО, предоставляются Заказчиком и размещены на базе вычислительных ресурсов Заказчика.

Программное обеспечение должно корректно функционировать и выполнять показатели назначения на серверных электронно-вычислительных машинах со следующими минимальными характеристиками:

– процессор (CPU): не менее 24 ядер;

– оперативная память (RAM): не менее 64 ГБ;

– накопитель на магнитном диске либо твердотельный: не менее 6 ТБ полезной емкости;

– архитектура процессора и операционной системы: 64 разряда,

Программное обеспечение функционирует под управлением операционной системы Ubuntu версии 18.04 и выше.

АРМ пользователя должен размещаться на устройствах, обеспечивающих его индивидуальное использование авторизуемым пользователем (в рамках обеспечения безопасности в части доступа к информационной системе и ее данным), а также поддержку браузеров Chrome 85+.

В качестве периферийных ТС рекомендуется использование периферийных устройств сбора, подготовки и отображения данных:

– стандартная клавиатура и манипулятор типа «мышь»;

– монитор LCD с разрешением не менее 1440х900 и диагональю не менее 17”.

Для функционирования ПО на автоматизированных рабочих местах (далее – АРМ) пользователей рекомендуется использование ТС с характеристиками не ниже указанных в таблице «Технические характеристики АРМ пользователя» (таблица 6).

Таблица 6 – Технические характеристики АРМ пользователя

| ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ | ЗНАЧЕНИЕ, НЕ МЕНЕЕ |
| --- | --- |
| Системный блок: | |
| Частота процессора, ГГц | 3 |
| Тип оперативной памяти | «DDR 4» |
| Объем оперативной памяти, Гбайт | 8 |
| Частота шины оперативной памяти, МГц | 2666 |
| Тип запоминающего устройства | «HDD» или «SDD» |
| Объем запоминающего устройства, Гбайт | 100 |
| Мощность блока питания, Вт | 400 |

# Вызов и загрузка

## Способ вызова программы с соответствующего носителя данных

Пользователь может обращаться к программе, работая как на ПЭВМ под управлением Linux, так и на ПЭВМ под управлением Windows. Программа вызывается пользователем через браузер обращением по ее адресу посредством HTTP-запросов с аутентификацией.

Подготовка программы к работе и ее настройка администратором описаны в документе «Руководство администратора».

## Входные точки в программу

Адрес доступа сообщается пользователям администратором системы.

# Входные данные

Программа поддерживает графический интерфейс работы с пользователем, поэтому все произведенные пользователем воздействия на программу можно рассматривать как входные данные в нее (рисунок 9, поток 3). Описание работы с программой через графический интерфейс приведено в документах:

«Руководство пользователя»;

«Руководство администратора».

Рисунок 9 – Схема информационных потоков

Описание информационных потоков представлено ниже (таблица 7).

Таблица 7 – Описание информационных потоков

| № ПОТОКА НА СХЕМЕ | ПЕРЕДАВАЕМЫЕ ДАННЫЕ |
| --- | --- |
| 1 | Получение изображений с камер |
| 2 | Отправка изображения на детектор |
| 3 | Настройки заданий анализа парковочных мест, через интерфейс пользователя |

# Выходные данные

Программа поддерживает графический интерфейс работы с пользователем, поэтому все высвеченные для пользователя формы и их информационное наполнение можно рассматривать как выходные данные программы. Описание работы с программой через графический интерфейс приведено в документах:

«Руководство пользователя»;

«Руководство администратора».

### Структура данных

Данные содержат:

### Формат данных

Данные представляют набор значений в формате значений полей SQL-таблицы.

### Описание данных

Данные представляют количественную оценку

# ПЕРЕЧЕНЬ ТЕРМИНОВ

Термины и их определения представлены ниже (таблица 8).

Таблица 8 – Термины и их определения

|  |  |
| --- | --- |
| ТЕРМИН | ОПРЕДЕЛЕНИЕ |
| [Модуль распознавания регистрационных знаков и определения характеристик транспортных средств](https://confluence.amtech.ru/pages/viewpage.action?pageId=1066144072) | Программное обеспечения, предназначенное для распознавания свойств ТС и получения bbox ТС на исходном изображении |
| Парковочное место |  |
| Парковочное пространство |  |
| Шлюз ЕЦХД |  |
| СУДИР |  |
| bbox |  |
| HTTP | Протокол прикладного уровня передачи данных, в соответствии с [RFC 2616] |
| JSON | (JavaScript Object Notation) –текстовый формат обмена данными, в соответствии с [RFC 8259] |

# ПЕРЕЧЕНЬ СОКРАЩЕНИЙ

Сокращения и их расшифровки представлены ниже (таблица 9).

Таблица 9 – Сокращения и их расшифровки

| СОКРАЩЕНИЕ | РАСШИФРОВКА |
| --- | --- |
| АРМ | Автоматизированное рабочее место |
| БД | База данных |
| ГОСТ | Государственный стандарт |
| ОС | Операционная система |
| ПО | Программное обеспечение |
| СУБД | Система управления базами данных |
| ТС | Технические средства |
| API | (Application Programming Interface) — описание способов (набор классов, процедур, функций, структур или констант), которыми одна компьютерная программа может взаимодействовать с другой программой |
| DDR | (Double Data Rate) – динамическая память с удвоенной скоростью передачи |
| HDD | (Hard Disk Drive) – накопитель на жестком диске |
| HTTP | (HyperText Transfer Protocol) — протокол прикладного уровня передачи данных. |
| IP | (Internet Protocol) – протокол Интернета |
| SQL | (Structured Query Language) – декларативный язык программирования, применяемый для создания, модификации и управления данными в реляционной базе данных, управляемой соответствующей системой управления базами данных |
| SSD | (Solid-State Drive) – компьютерное энергонезависимое немеханическое запоминающее устройство на основе микросхем памяти |
| TCP | (Transmission Control Protocol) – один из основных протоколов передачи данных Интернета, предназначенный для управления передачей данных |
| TCP/IP | (Transmission Control Protocol/Internet Protocol) – сетевая модель передачи данных, представленных в цифровом виде. Модель описывает способ передачи данных от источника информации к получателю. В модели предполагается прохождение информации через четыре уровня, каждый из которых описывается правилом (протоколом передачи). Наборы правил, решающих задачу по передаче данных, составляют стек протоколов передачи данных, на которых базируется Интернет |
| XML | (eXtensible Markup Language) – расширяемый язык разметки |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Лист регистрации изменений** | | | | | | | | | |
| Изм | Номера листов (страниц) | | | | Всего листов (страниц) в документе | №  документа | Входящий № сопроводит. документа и дата | Подп. | Дата |
| Измененных | Замененных | Новых | Аннулированных |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |