ПРОГРАММНО-АППАРАТНЫЙ КОМПЛЕКС

ИНДИВИДУАЛЬНОЙ ВИДЕОРЕГИСТРАЦИИ, КАТАЛОГИЗАЦИИ И ДОЛГОВРЕМЕННОГО ХРАНЕНИЯ ВИДЕОАРХИВОВ

СПО «WEB-интерфейс»

ОПИСАНИЕ ПРОГРАММЫ

РАМГ.50125-05 13 01

2021

содержание

[1 Общие сведения 3](#_Toc43727765)

[1.1 Обозначение и наименование программы 3](#_Toc43727766)

[1.2 Программное обеспечение, необходимое для функционирования программы 3](#_Toc43727767)

[1.3 Языки программирования, на которых написана программа 4](#_Toc43727768)

[2 Функциональное назначение 5](#_Toc43727769)

[3 Описание логической структуры 6](#_Toc43727770)

[3.1 Алгоритм программы 6](#_Toc43727771)

[3.2 Используемые методы 8](#_Toc43727772)

[3.3 Структура программы с описанием функций составных частей и связи между ними 8](#_Toc43727773)

[3.4 Связи программы с другими программами 9](#_Toc43727774)

[4 Используемые технические средства 10](#_Toc43727775)

[5 Вызов и загрузка 11](#_Toc43727776)

[6 Входные и выходные данные 12](#_Toc43727777)

[6.1 Характер входных и выходных данных 12](#_Toc43727778)

[6.2 Формат, описание и способ кодирования входных и выходных данных 12](#_Toc43727779)

# Общие сведения

## Обозначение и наименование программы

Наименование: Специальное программное обеспечение «WEB-интерфейс» (далее по тексту – Программа) из состава программно-аппаратного комплекса индивидуальной видеорегистрации, каталогизации и долговременного хранения видеоархивов (ПАК «Видеоархив») РАМГ.466535.016.

Обозначение: РАМГ.50125-05.

## Программное обеспечение, необходимое для функционирования программы

Для обеспечения функционирования Программы необходим компьютер с предварительно установленной операционной системой специального назначения «AstraLinux SpecialEdition» – версия Смоленск 1.6.

Для запуска Программы на ЭВМ должны быть установлены программные пакеты.

* php;
* apache2;
* ffmpeg.

Для наполнения Программы контентом необходимо установленное на той же ЭВМ СПО «Сервер доступа» РАМГ.50125-06 или СПО «Терминал» РАМГ.50125-02 с соответствующим набором программных пакетов:

* gcc g++
* qtmultimedia5-dev
* libqt5multimediawidgets5
* libqt5multimedia5-plugins
* libqt5multimedia5
* qt5-qmake
* qt5-default
* rsync
* mercurial
* make
* libqt5serialport5-dev
* sqlite3
* qt5-qmake qt5-default
* rsync

из комплекта диска для разработчиков для ОС «Astra Linux Special Edition» – версия Смоленск 1.6. devel-smolensk-1.6-20.06.2018\_15.

Программа использует скрипты JavaSccriptc открытым исходным кодом:

* query-3.5.1.min.js;
* jquery-ui.min.js;
* jquery-ui-timepicker-addon.min.js.

## Языки программирования, на которых написана программа

Программа разработана на языках программирования HTML 5, JavaScript, PHP7.0, CSS, которые находятся в составе диска разработчика Astra Linux 1.6.

# Функциональное назначение

Программа позволяет получить пользовательский интерфейс для доступа к видеоархивам, размещенным на Терминалах и системах хранения данных, посредством протокола HTTP и веб-браузера.

Программа предназначена для:

* отображения списка сохранных в видеоархиве фото и видеофайлов;
* выбора фото и видеофайлов из списка для просмотра;
* просмотра выбранных видеофайлов в формате H.264 и Н.265 в оконном и полноэкранном режимах;
* просмотра выбранных фото-файлов в формате JPEG, в полноэкранном и оконном режимах;
* поиска фото и видеофайлов по автору записи;
* поиска фото и видеофайлов по диапазону даты записи;
* поиска фото и видеофайлов месту записи (номер Терминала или географическая область);
* отображения списка событий (журнал событий).

# Описание логической структуры

## Алгоритм программы

Алгоритмы реализации основных функций:

* авторизация клиента сервера;
* поддержание сессии;
* получение списков, данных без обработки;
* получение выборочных данных;
* описание запроса на изменение данных;
* просмотр видеозаписей.

3.1.1 Описание алгоритма авторизации

Модуль php Программы устанавливает соединение с API СПО «Сервер доступа» или СПО «Терминал», как клиент. Клиент посылает на сервер строку данных с меткой «логин».

Сервер принимает строку и сверяет наличие такого логина в БД. Если логин верный, то сервер передает связанное с паролем число клиенту. Если логин не верный, то также возвращается случайное число. Клиент сервера соединяет полученное число с введённым пользователем паролем и производит набор математических преобразований над полученным числом. Результат преобразований направляется на сервер. Сервер производит аналогичные действия над паролем и сравнивает полученные и расчетные значения. В случае успеха клиенту сервера направляется номер текущей сессии. Все последующие запросы клиент сервера направляет с номером сессии. В случае если логин или пароль были не верны, сервер направляет клиенту сообщение об ошибке.

3.1.2 Описание поддержания сессии

После успешной авторизации сервер начинает сессию. Каждое обращение к API клиент маркирует меткой сессии. Если пользователь работает в графическом интерфейсе, но необходимости формировать запросы к API нет, то такие запросы формируются автоматически для поддержания сессии.

В случае физического разрыва сессии или отсутствия запросов от клиента сервера происходит разрыв сессии. С этого момента запросы с меткой закрытой сессией игнорируются. В этом случае Клиент производит повторную авторизацию.

3.1.3 Описание получения списков данных без обработки

Во время действия сессии клиент может запросить следующие списки, указав количество данных в странице:

* список пользователей;
* список записанных событий;
* список карт доступа;
* список событий;
* список загруженных файлов;
* список регистраторов;
* список настроек сервера;
* список настроек регистратора.

Для получения полного списка выполняйся несколько запросов страниц.

3.1.4 Описание получения выборочных данных

Для получения выборочных данных со стороны клиента сервера формируется специально подготовленные данные условия:

* временной интервал;
* тип события;
* Id пользователя;
* Id системного события;
* Id регистратора;
* тип файла;
* шаблон имени файла.

Затем формируется запрос, содержащий «имя списка», который надо получить, условия формирования списка и метку сессии. Сервер, получив эти данные, формирует запросы к БД, получает от БД ответ и формирует страницы списка с ответами.

3.1.5 Описание запроса на изменения данных

В результате действий пользователя Программе требуется изменить данные.

Программа, выступая в роли клиента, формирует запрос к API сервера на изменение данных. При этом указывается тип изменяемых данных, Id экземпляра и само изменение.

Сервер принимает такой запрос по номеру сессии, определят Id пользователя. Если пользователь имеет право производить такие изменения, то сервер производит изменения данных в БД.

После завершения операции клиенту сервера направляется изменённый экземпляр данных или сообщение об ошибке.

3.1.6 Описание просмотра видеофайлов

Пользователь Программы из списка доступных для просмотра файлов выбирает один для просмотра. ID этого файла передается серверной части Программы, где начинает работу PHP скрипт.

Скрипт по ID находит расположение файла в хранилище и запускает ряд утилит, которые начинают потоковую трансляцию этого файла. Затем PHP передает метку трансляции в браузер где макрос JS перенаправляет трансляцию в графический интерфейс.

При необходимости пользователь может изменить параметры трансляции, изменив скорость и направление воспроизведения.

## Используемые методы

Программа разработана с использованием методов в соответствии со стандартом языка HTML5, и PHP версии 7.0.

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

Программа – клиент-серверное приложение, в котором клиент взаимодействует с WEB-сервером при помощи браузера.

Логика WEB-приложения распределена между сервером и клиентом, хранение данных осуществляется на сервере, обмен информацией происходит по локальной сети. Одним из преимуществ такого подхода является тот факт, что клиенты не зависят от конкретной операционной системы пользователя, поэтому веб-приложения являются межплатформенными службами.

WEB-интерфейс позволяет эффективно одновременно работать с данными многим пользователям.

Структура Программы приведена на рисунке 3.1 и включает следующие элементы:

Модуль HTML

Модуль CSS

Модуль JS

Модуль PHP

API СПО сервер

Рисунок 3.1 – Логическая структура Программы

* Модули HТML, JS и CSS загружаются в браузер пользователя и формируют интерактивный графический интерфейс.
* Модуль JS обеспечивает обработку действий пользователя в графическом интерфейсе и формирует запросы на серверную часть, где выполняются модули PHP.
* Модули PHP через командную оболочку операционной системы взаимодействуют с API СПО «Сервер доступа» или API СПО «Терминал».

## Связи программы с другими программами

Программа взаимодействует с СПО «Терминал» и СПО «Сервер доступа».

Программа взаимодействует с браузерами, расположенными на компьютере пользователей.

# Используемые технические средства

Программа разработана для функционирования на Терминалах в вариантах исполнения или на сервере предоставления доступа к файловым ресурсам из состава СХД в вариантах исполнения, входящих в состав ПАК «Видеоархив» РАМГ.466535.016. WEB-браузер, на котором осуществляется работа, размещается на АРМ доступа к видеоархиву или на любом компьютере пользователя, связанном с СХД или Терминалом.

Минимальные системные требования к компьютеру пользователя6

* монитор с разрешением не менее 1980 х1080;
* наличие одного из браузеров: Chrome, Opera, Firefox (новее 60 версии) или Internet Explorer (новее 9 версии).

Остановить работу Программы в случае необходимости можно в настройках СПО «Терминал» или настройках СПО «Сервер доступа».

# Вызов и загрузка

Программа устанавливается на предприятии-изготовителе во время настройки сервера предоставления доступа к файловым ресурсам или Терминала. Запуск производится автоматически во время включения.

Программа готова для подключения браузеров пользователей. Браузеры пользователи Программы запускают по мере необходимости.

# Входные и выходные данные

* 1. Характер входных и выходных данных

Входные данные – данные, вводимые в графическом интерфейсе пользователя и получаемые от API СПО «Терминал» или от API СПО «Сервер доступа».

Выходные данные – являются результатом работы ПО и отображаются в графическом интерфейсе пользователя или передаются в API СПО «Сервер доступа».

* 1. Формат, описание и способ кодирования входных и выходных данных

Форматы, описания и способ кодирования данных, вводимых пользователем, задаются графической средой.

Входные данные, с которыми работает Программа приведены в таблице 6.1.

Таблица 6.1 – Входные данные Программы

| Название | Источник | Тип |
| --- | --- | --- |
| Логин | Браузер | Строка |
| Пароль | Браузер | Строка |
| Ответ на запрос от сервера доступа | СПО «Сервер доступа» | Массив файлов |
| Ответ на запрос Терминала | СПО «Терминал | Массив файлов |
| Настройки соединения с удаленным СХД | Браузер | Структура |
| Данные пользователя | Браузер | Структура |
| Настройки регистратора | Браузер | Структура |
| Отбор имени файла в отчет | Браузер | Структура |
| Запрос пользователя на поиск в БД | Браузер | Структура |
| Настройки сервера | Браузер | Структура |

Выходные данные, с которыми работает Программа приведены в таблице 6.2.

Таблица 6.2 – Выходные данные Программы

| Название | Получатель | Тип |
| --- | --- | --- |
| Запрос к СХД | СПО «Сервер доступа» | Структура |
| Запрос к  Терминалу | СПО «Терминал» | Структура |
| Отчет | Съемный накопитель | Файл |
| Видео | Съемный накопитель | Файл |
| Видео | Браузер | Видеопоток |
| Фото | Браузер | Файл |
| Данные о регистраторах | Браузер | Экранная форма |
| Данные о файле | Браузер | Экранная форма |
| Список пользователей | Браузер | Экранная форма |
| Список файлов | Браузер | Экранная форма |
| Список настроек Терминала | Браузер | Экранная форма |
| Список настроек регистратора | Браузер | Экранная форма |
| Список настроек пользователя | браузер | Экранная форма |

Обмен данными между Программой и клиентами производиться в виде информационных объектов (структур). В текстовом виде такие структуры могут быть записаны в формате JSON. Входные и выходные данные имеют одинаковую структуру.

Пример входных-выходных данных, которые описывают системные события, произошедшие на Терминале.

{

"log": {

"pageCount": 1,

"pageNum": 1,

"rowsOnPage": 10,

"row": [

{

"id": "{5944cae3-44de-49d6-b82a-584530df2d76}",

"dt": 0,

"type": 11,

"user": "{9f377c68-e080-11e6-af86-7305b500b54c}",

"info": "Доступ по логину \'user5\'.",

"dev": "",

"file": "users/key/{9f377c68-e080-11e6-af86-7305b500b54c}/cam\_20200601-142027.jpg",

"ucomm": ""

},

{

"id": "{927455a0-df50-4628-adda-f0774361ab0a}",

"dt": 0,

"type": 12,

"user": "{9f377c68-e080-11e6-af86-7305b500b54c}",

"info": "Выход",

"dev": "",

"file": "",

"ucomm": ""

},

{

"id": "{3c68a185-9f8e-4a05-ad79-88770b74493f}",

"dt": 0,

"type": 11,

"user": "{7f4835cc-e080-11e6-9925-43c60cf49054}",

"info": "Доступ по логину \'user4\'.",

"dev": "",

"file": "users/key/{7f4835cc-e080-11e6-9925-43c60cf49054}/cam\_20200601-142034.jpg",

"ucomm": ""

},

Описание полей структуры системные события приведены в таблице 6.3.

Таблица 6.3 – Описание полей структуры системные события

| Параметр | Описание | Тип |
| --- | --- | --- |
| log | Имя структуры | структура |
| pageCount | Номер страницы | число |
| pageNum | Число страниц | число |
| rowsOnPage | Строк в странице | число |
| row | сообщение | структура |
| id | Номер сообщения | строка |
| dt | Дата события | число |
| type | Тип события | строка |
| user | ID пользователя автора | строка |
| info | Примечание | строка |
| dev | устройство | строка |
| file | Принятый файл | строка |
| ucomm | комментарий |  |

Пример входных-выходных данных, которые содержат данные о пользователях Терминала или сервера.

{

"user": [

{

"id": "{00000000-0000-0000-0000-000000000000}",

"login": "admin",

"salt": "6S5hMp38",

"f": "administartor",

"i": "",

"o": "",

"card": "348FBEB9397566",

"rule": 9,

"blocked": 0,

"fields": ["", "", "", "", "", "", "", "", "", ""]

},

{

"id": "{3f4835cc-e080-11e6-9925-43c60cf49054}",

"login": "user2",

"salt": "iF306IwG",

"f": "Шевелёва",

"i": "Надежда",

"o": "Васильевна",

"card": "34FEED612A2B77",

"rule": 5,

"blocked": 0,

"fields": ["", "", "", "", "", "", "", "", "", ""]

},

{

"id": "{4f377c68-e080-11e6-af86-7305b500b54c}",

"login": "user3",

"salt": "9hFAb048",

"f": "",

"i": "Ипполит",

"o": "Георгиевич",

"card": "34415581E2F786",

"rule": 7,

"blocked": 0,

"fields": ["", "", "", "", "", "", "", "", "", ""]

}

Описание полей структуры «Пользователи» приведены в таблице 6.4.

Таблица 6.4 – Описание полей структуры «Пользователи»

| Параметр | Описание | Тип |
| --- | --- | --- |
| user | Название структуры | структура |
| id | Уникальный номер пользователя | строка |
| login | Логин | строка |
| salt | Хеш пароля | строка |
| f | Фамилия | строка |
| i | Имя | строка |
| o | Отчество | строка |
| card | Данные с карты доступа | строка |
| rule | Принадлежность к группам | число |

Лист регистрации изменений

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Изм. | Номера листов (страниц) | | | | Всего листов (стр.) в документе | № документа | Входящий  № сопроводи-тельного  документа | Под-пись | Дата |
| изме­нен­ных | Заме-нен-ных | новых | Аннулиро-ванных |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |