|  |  |  |
| --- | --- | --- |
| **МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ** | | |
|  | **федеральное государственное бюджетное образовательное учреждение высшего образования «Российский экономический университет имени Г.В. Плеханова»**  **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**  Московский приборостроительный техникум |

**ОТЧЕТ**

по производственной практике

ПП.04.01 «Сопровождение и обслуживание программного обеспечения компьютерных

систем»

Профессионального модуля ПМ.04 «Сопровождение и обслуживание программного

обеспечения компьютерных систем»

Специальность 09.02.07 Информационные системы и программирование

Квалификация: Программист

Студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Игошев Ростислав Вадимович

*подпись фамилия, имя, отчество*

Группа П50-4-21

Руководитель практической подготовки от профильной организации

Инженер, АО «Корпорация космических систем специального назначения «Комета»

*Должность, название профильной организации*

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Солдатов Иван Александрович

*подпись фамилия, имя, отчество*

М.П. «\_\_\_\_» \_\_\_\_\_\_\_\_\_ 2025 года

Руководитель практической подготовки от

Московского приборостроительного техникума

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Ермашенко Екатерина Антоновна

*подпись фамилия, имя, отчество*

«\_\_\_\_» \_\_\_\_\_\_\_\_\_ 2025 года

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 3](#_Toc194858516)

[1. ЦЕЛИ И ЗАДАЧИ ПРАКТИКИ 4](#_Toc194858517)

[2. БАЗА ПРАКТИКИ 6](#_Toc194858518)

[3. СОДЕРЖАНИЕ ПРАКТИКИ 7](#_Toc194858519)

[ЗАКЛЮЧЕНИЕ 8](#_Toc194858520)

[ПРИЛОЖЕНИЕ А. ТЕКСТ ПРОГРАММЫ 10](#_Toc194858521)

ВВЕДЕНИЕ

Производственная практика производилась в Акционерном обществе «Корпорация космических систем специального назначения «Комета», основная деятельность корпорации заключается в создании больших информационно-управляющих систем различного назначения. Практика проводилась очно по адресу Москва, ул. Велозаводская, д. 5.

Приоритетными направлениями деятельности АО «Корпорация «Комета» являются:

- Прикладные и фундаментальные исследования в области создания космических информационно-управляющих систем;

- Научно-исследовательские работы по изучению фоновых обстановок, обнаружению малоконтрастных объектов и обработке больших потоков информации;

- Создание глобальной космической системы всепогодного наблюдения Земли.

В должностные обязанности входит: разработка базы данных и различных программных модулей.

1. ЦЕЛИ И ЗАДАЧИ ПРАКТИКИ

Цель практики - освоение студентами всех видов профессиональной деятельности по специальности 09.02.07 «Информационные системы и программирование», формирование общих и профессиональных компетенций, а также приобретение необходимых умений и опыта практической работы по специальности.

Производственная практика включает в себя следующие этапы: практика по профилю специальности и преддипломная практика.

Практика по профилю специальности ПП.04.01 «Сопровождение и обслуживание программного обеспечения компьютерных систем» направлена на формирование у студентов общих и профессиональных компетенций, приобретение практического опыта и реализуется в рамках профессионального модуля.

Общие компетенции:

ОК 01. Выбирать способы решения задач профессиональной деятельности, применительно к различным контекстам;

ОК 02. Использовать современные средства поиска, анализа и интерпретации информации и информационные технологии для выполнения задач профессиональной деятельности;

ОК 03. Планировать и реализовывать собственное профессиональное и личностное развитие, предпринимательскую деятельность в профессиональной сфере, использовать знания по правовой и финансовой грамотности в различных жизненных ситуациях;

ОК 04. Эффективно взаимодействовать и работать в коллективе и команде;

ОК 05. Осуществлять устную и письменную коммуникацию на государственном языке с учетом особенностей социального и культурного контекста;

ОК 06. Проявлять гражданско-патриотическую позицию, демонстрировать осознанное поведение на основе традиционных российских духовно-нравственных ценностей, в том числе с учетом гармонизации межнациональных и межрелигиозных отношений, применять стандарты антикоррупционного поведения;

ОК 07. Содействовать сохранению окружающей среды, ресурсосбережению, применять знания об изменении климата, принципы бережливого производства, эффективно действовать в чрезвычайных ситуациях;

ОК 08. Использовать средства физической культуры для сохранения и укрепления здоровья в процессе профессиональной деятельности и поддержания необходимого уровня физической подготовленности;

ОК 09. Пользоваться профессиональной документацией на государственном и иностранном языках.

Профессиональные компетенции:

ПК 4.1. Осуществлять инсталляцию, настройку и обслуживание программного обеспечения компьютерных систем.

ПК 4.2. Осуществлять измерения эксплуатационных характеристик программного обеспечения компьютерных систем

ПК 4.3. Выполнять работы по модификации отдельных компонент программного обеспечения в соответствии с потребностями заказчика

ПК 4.4 Обеспечивать защиту программного обеспечения компьютерных систем программными средствами.

2. БАЗА ПРАКТИКИ

Полное название организации: Акционерное общество «Корпорация космических систем специального назначения «Комета».

Краткое название: АО «Корпорация «Комета».

Подразделение: ОКБ-42 Отдел программного обеспечения

Руководитель от организации: Солдатов Иван Александрович

Должность руководителя: Инженер-программист

Адрес организации: Москва, ул. Велозаводская, д. 5.

В качестве средств вычислительной техники использовался персональный компьютер. Его характеристики представлены в таблице 1.

Таблица 1 – Программные средства

|  |  |  |  |
| --- | --- | --- | --- |
| № | Тип средства | Название средства | Назначение |
| 1 | 2 | 3 | 4 |
| 1 | Система разработки программного обеспечения | QT Creator | Разработка программного обеспечения на языке C++ |
| 2 | Текстовый редактор | LibreOffice Writer | Разработка документации, формирование отчётных документов по шаблонам |

Таблица 2 – Технические средства

|  |  |  |
| --- | --- | --- |
| № | Тип оборудование | Наименование оборудование |
| 1 | 2 | 3 |
| 1 | Размер экрана: | 17” |
| 2 | Разрешение экрана: | 1920x1080 пикс. |
| 3 | Линейка процессора: | 2 x Intel Pentium 4 CPU 3GHz |
| 4 | Количество ядер процессора: | 2 |
| 5 | Оперативная память: | 2 ГБ |
| 6 | Тип видеокарты: | Интегрированная |
| 7 | Видеокарта: | - |
| 8 | Конфигурация накопителей: | HDD |
| 9 | Общий объем всех накопителей: | 120 ГБ |
| 10 | Операционная система | Astra Linux |

3. СОДЕРЖАНИЕ ПРАКТИКИ

При сопровождении программного продукта выявились дополнительные требования к разработанной программе, не входящие в изначальное техническое задание. Дополнительно в программный продукт необходимо ввести: модуль подключения к серверу по вводимому адресу.

Виды повседневных работ, выполнимых на практике представлены в таблице 3.

Таблица 3 – Виды работ

|  |  |
| --- | --- |
| Дата или период выполнения  работ | Краткое содержание  выполняемых работ |
| 10.02.2025 | Анализ проектной документации |
| 13.02.2025 | Анализ технической документации |
| 17.02.2025 | Разработка требований к программным модулям |
| 20.02.2025 | Разработка программного модуля |
| 24.02.2025 | Первичная отладка программного модуля с использованием встроенных инструментов отладки рабочей среды |
| 28.02.2025 | Доработка программного модуля |
| 03.03.2025 | Конечная отладка программного модуля с использованием встроенных инструментов отладки рабочей среды |
| 07.03.2025 | Интеграция программного модуля в программное обеспечение |
| 10.03.2025 | Реализация сообщений об ошибках |
| 14.03.2025 | Отладка программного обеспечения после интеграции программного модуля |
| 17.03.2025 | Выявление уязвимых частей программы |
| 21.03.2025 | Разработка тестовых сценариев |
| 24.03.2025 | Первичное тестирование |
| 28.03.2025 | Исправление ошибок программного кода |
| 01.04.2025 | Финальное тестирование |
| 05.04.2025 | Инспектирование программного обеспечения на предмет соответствия стандартам кодирования |
| 08.04.2025 | Оформление необходимой документации |
| 12.04.2025 | Печать и подпись готовой документации |

Ниже, на рисунках 1-3, представлены изображения, на которых продемонстрирована работа разработанного в ходе производственной практики программного модуля.

При запуске модуля, появляется окно ввода IP-адреса и порта для дальнейшего подключения к серверу, которое показано на рисунке 1.

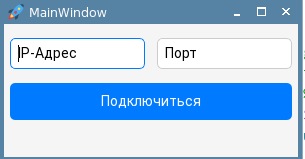


Рисунок 1 – Начальное окно модуля

В начальном окне для подключения необходимо заполнить два поля: IP-адрес и порт. Пример вводимых данных изображен на рисунке 2.

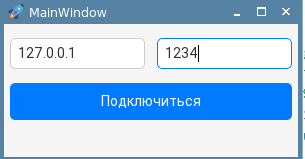


Рисунок 2 – Пример вводимых данных для подключения

Введя верный адрес и порт при запущенном сервере после нажатия кнопки «Подключиться» произойдет успешное подключение, в результате чего откроется основное приложение (рисунок 3), где пользователь сможет работать с установленным клиент-серверным подключением.

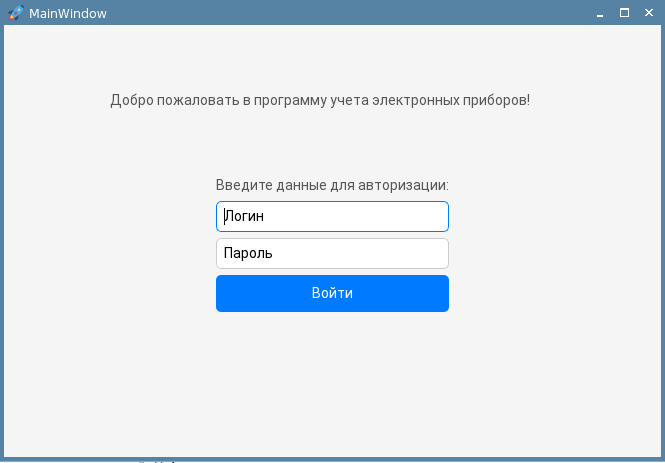


Рисунок 3 – Открывшееся окно авторизации после успешного подключения

ЗАКЛЮЧЕНИЕ

В ходе производственной практики были успешно освоены навыки сопровождения и поддержки ПО, командной работы. Научился работать с техническим заданием, правильно выполнять отладку программных модулей и интегрировать их в программное обеспечение.

Были выполнены следующие компетенции:

ОК 01. Выбирать способы решения задач профессиональной деятельности, применительно к различным контекстам;

ОК 02. Использовать современные средства поиска, анализа и интерпретации информации и информационные технологии для выполнения задач профессиональной деятельности;

ОК 03. Планировать и реализовывать собственное профессиональное и личностное развитие, предпринимательскую деятельность в профессиональной сфере, использовать знания по правовой и финансовой грамотности в различных жизненных ситуациях;

ОК 04. Эффективно взаимодействовать и работать в коллективе и команде;

ОК 05. Осуществлять устную и письменную коммуникацию на государственном языке с учетом особенностей социального и культурного контекста;

ОК 07. Содействовать сохранению окружающей среды, ресурсосбережению, применять знания об изменении климата, принципы бережливого производства, эффективно действовать в чрезвычайных ситуациях;

ОК 09. Пользоваться профессиональной документацией на государственном и иностранном языках.

ПК 4.1. Осуществлять инсталляцию, настройку и обслуживание программного обеспечения компьютерных систем.

ПК 4.2. Осуществлять измерения эксплуатационных характеристик программного обеспечения компьютерных систем

ПК 4.3. Выполнять работы по модификации отдельных компонент программного обеспечения в соответствии с потребностями заказчика

ПК 4.4 Обеспечивать защиту программного обеспечения компьютерных систем программными средствами.

ПРИЛОЖЕНИЕ А. ТЕКСТ ПРОГРАММЫ

АННОТАЦИЯ

В данном программном документе приведён текстовые данные модуля подключения к серверу ИС отслеживания данных электронных устройств.

В разделе «Текст программы» указано наименование модуля, область применения программы и код.

ТЕКСТ ПРОГРАММЫ

1. Наименование объекта

Наименование - «EnterIp».

1. Область применения объекта

Модуль «EnterIp» предоставляет функционал для подключения к серверной части ИС отслеживания данных электронных устройств.

1. Код программы

Ниже представлен код модуля, который позволяет устанавливать подключение:  
——————— (cpp) ———————

#include "enteripwindow.h"

#include "ui\_enteripwindow.h"

#include "main.h"

#include "mainwindow.h"

bool double\_click\_ip = false;

EnterIpWindow::EnterIpWindow(QWidget \*parent) :

QMainWindow(parent),

ui(new Ui::EnterIpWindow)

{

ui->setupUi(this);

connect(ui->connectPushButton, &QPushButton::clicked, this,

&EnterIpWindow::on\_connectPushButton\_clicked);

setWindowFlags(Qt::Dialog);

}

EnterIpWindow::~EnterIpWindow()

{

delete ui;

}

bool EnterIpWindow::isNumber(const QString& str){

bool ok;

str.toInt(&ok);

return ok;

}

void EnterIpWindow::on\_connectPushButton\_clicked()

{

if (double\_click\_ip){

double\_click\_ip = false;

}

else {

double\_click\_ip = true;

QString ip = ui->ipLineEdit->text();

int port = 0;

if(isNumber(ui->portLineEdit->text())){

port = ui->portLineEdit->text().toInt();

}

else {

port = 1234;

}

//client->error = false;

try{

client = new TcpClient(ip, port);

client->checkConnection();

}

catch (...){

client->error = true;

}

if (client->error){

QMessageBox::critical(this, "Не удалось подключиться", "Введите корректный адрес и порт.");

client->error = false;

}

else{

MainWindow \* w = new MainWindow();

w->show();

this->close();

}

}

}

——————— (h) ———————

#ifndef ENTERIPWINDOW\_H

#define ENTERIPWINDOW\_H

#include <QMainWindow>

#include <QMessageBox>

namespace Ui {

class EnterIpWindow;

}

class EnterIpWindow : public QMainWindow

{

Q\_OBJECT

public:

explicit EnterIpWindow(QWidget \*parent = nullptr);

~EnterIpWindow();

private slots:

void on\_connectPushButton\_clicked();

bool isNumber(const QString& str);

private:

Ui::EnterIpWindow \*ui;

};

#endif // ENTERIPWINDOW\_H

——————— (ui) ———————

<?xml version="1.0" encoding="UTF-8"?>

<ui version="4.0">

<class>EnterIpWindow</class>

<widget class="QMainWindow" name="EnterIpWindow">

<property name="geometry">

<rect>

<x>0</x>

<y>0</y>

<width>294</width>

<height>133</height>

</rect>

</property>

<property name="windowTitle">

<string>MainWindow</string>

</property>

<widget class="QWidget" name="centralwidget">

<layout class="QGridLayout" name="gridLayout">

<item row="0" column="0">

<widget class="QLineEdit" name="ipLineEdit">

<property name="placeholderText">

<string>IP-Адрес</string>

</property>

</widget>

</item>

<item row="0" column="2">

<widget class="QLineEdit" name="portLineEdit">

<property name="placeholderText">

<string>Порт</string>

</property>

</widget>

</item>

<item row="1" column="0" colspan="3">

<widget class="QPushButton" name="connectPushButton">

<property name="text">

<string>Подключиться</string>

</property>

</widget>

</item>

</layout>

</widget>

<widget class="QMenuBar" name="menubar">

<property name="geometry">

<rect>

<x>0</x>

<y>0</y>

<width>294</width>

<height>29</height>

</rect>

</property>

</widget>

<widget class="QStatusBar" name="statusbar"/>

</widget>

<resources/>

<connections/>

</ui>

——————— (cpp) ———————

#include "tcpclient.h"

#include <QDebug>

QTcpSocket socket;

bool error;

TcpClient::TcpClient(const QString &host, int port, QObject \*parent) : QObject(parent)

{

error = false;

socket = new QTcpSocket(this);

connect(socket, &QTcpSocket::connected, this, &TcpClient::onConnected);

connect(socket, &QTcpSocket::readyRead, this, &TcpClient::onReadyRead);

connect(socket, &QTcpSocket::errorOccurred, this, &TcpClient::onError);

socket->connectToHost(host, port);

}

QList<QJsonObject> TcpClient::sendDataList(const QByteArray &msg){

QList<QJsonObject> none;

socket->waitForConnected(5000);

if (socket->isOpen()){

socket->write(msg);

socket->waitForBytesWritten(5000);

}

else{

qDebug() << "Не удалось подключиться";

return none;

}

socket->waitForReadyRead(5000);

QByteArray data = socket->readAll();

QJsonDocument jsonDoc = QJsonDocument::fromJson(data);

QJsonObject jsonResponce = jsonDoc.object();

for (int i = 0; i < jsonResponce.keys().size(); i++){

QString val = jsonResponce[jsonResponce.keys().at(i)].toString();

if (val != ""){

qDebug() << jsonResponce.keys().at(i) << " - "<< val;

}

else {

qDebug() << jsonResponce.keys().at(i) << " - "<< jsonResponce[jsonResponce.keys().at(i)].toDouble();

}

}

QString recievedCheckSum = jsonResponce["checksum"].toString();

jsonResponce.remove("checksum");

QByteArray jsonData = QJsonDocument(jsonResponce).toJson();

QString str = jsonResponce["data"].toString();

QByteArray jsonValue = str.toUtf8();

QByteArray calculatedChecksum = QCryptographicHash::hash(jsonValue, QCryptographicHash::Sha256).toHex();

QJsonDocument jsonDoc2 = QJsonDocument::fromJson(jsonValue);

if (recievedCheckSum != QString(calculatedChecksum)){

qDebug() << "Несовпадение контрольных сумм.";

return none;

}

else if (/\*jsonResponce["data"].isArray()\*/ true){

QJsonArray jsonArray = /\*jsonResponce["data"].toArray()\*/jsonDoc2.array();

QList<QJsonObject> arr;

for (const QJsonValue &value : jsonArray){

if (value.isObject()){

QJsonObject jsonObject = value.toObject();

qDebug() << "Объект: " << jsonObject;

arr.append(jsonObject);

}

}

return arr;

}

else{

qDebug() << "Получен не массив";

}

return none;

}

bool TcpClient::checkConnection(){

return socket->waitForConnected(5000);

}

QString TcpClient::sendDataString(const QByteArray &msg){

QList<QJsonObject> none;

socket->waitForConnected(5000);

if (error){

qDebug() << "Не удалось подключиться";

return "ERROR\_Не удалось подключиться к серверу.";

}

if (socket->isOpen()){

socket->write(msg);

socket->waitForBytesWritten(5000);

}

else{

qDebug() << "Не удалось подключиться2";

return "ERROR\_Не удалось подключиться к серверу.";

}

socket->waitForReadyRead(5000);

QByteArray data = socket->readAll();

QJsonDocument jsonDoc = QJsonDocument::fromJson(data);

QJsonObject jsonResponce = jsonDoc.object();

for (int i = 0; i < jsonResponce.keys().size(); i++){

QString val = jsonResponce[jsonResponce.keys().at(i)].toString();

if (val != ""){

qDebug() << jsonResponce.keys().at(i) << " - "<< val;

}

else {

qDebug() << jsonResponce.keys().at(i) << " - "<< jsonResponce[jsonResponce.keys().at(i)].toDouble();

}

}

QString recievedCheckSum = jsonResponce["checksum"].toString();

jsonResponce.remove("checksum");

QByteArray jsonData = QJsonDocument(jsonResponce).toJson();

QString str = jsonResponce["data"].toString();

QByteArray jsonValue = str.toUtf8();

QByteArray calculatedChecksum = QCryptographicHash::hash(jsonValue, QCryptographicHash::Sha256).toHex();

if (recievedCheckSum != QString(calculatedChecksum)){

qDebug() << "Несовпадение контрольных сумм.";

return "ERROR\_Несовпадение контрольных сумм.";

}

else if (jsonResponce["type"].toString() == "error"){

return "ERROR\_" + jsonResponce["data"].toString();

}

else {

return "DATA\_" + jsonResponce["data"].toString();

}

}

void TcpClient::onConnected(){

qDebug() << "Подключено к серверу";

}

void TcpClient::onError(QAbstractSocket::SocketError err){

qDebug() << "Ошибка: " << err;

error = true;

}

void TcpClient::onReadyRead(){

// QByteArray data = socket->readAll();

// QJsonDocument jsonDoc = QJsonDocument::fromJson(data);

// if (jsonDoc.isArray()){

// QJsonArray jsonArray = jsonDoc.array();

// for (const QJsonValue &value : jsonArray){

// if (value.isObject()){

// QJsonObject jsonObject = value.toObject();

// qDebug() << "Устройство: " << jsonObject;

// }

// }

// }

// else{

// qDebug() << "Получен не json-массив";

// }

}

——————— (h) ———————

#ifndef TCPCLIENT\_H

#define TCPCLIENT\_H

#include <QTcpSocket>

#include <QJsonArray>

#include <QJsonObject>

#include <QJsonDocument>

#include <QCryptographicHash>

class TcpClient : public QObject

{

Q\_OBJECT

public:

TcpClient(const QString &host, int port, QObject \*parent = nullptr);

QList<QJsonObject> sendDataList(const QByteArray &data);

QString sendDataString(const QByteArray &data);

bool error;

bool checkConnection();

private slots:

void onConnected();

void onError(QAbstractSocket::SocketError);

void onReadyRead();

private:

QTcpSocket \*socket;

};

#endif // TCPCLIENT\_H