

Министерство образования и науки Российской Федерации
Федеральное государственное автономное образовательное
учреждение высшего профессионального образования
«Национальный исследовательский Нижегородский государственный
университет им. Н.И. Лобачевского»
Институт информационных технологий, математики и механики

Отчет по программному проекту "IT-перспектива"
**"Устройство для наблюдения за состоянием здоровья человека в
рабочее время"**

Выполнили:

студент группы 382006-1

Юнин Д.Д.

студент группы 382008-1

Булгаков Д.Э.

Преподаватель:

Карчков Д.А.

Нижегород
2022 г.

Содержание

1. Введение.	3
2. Постановка задачи и цели работы.	4
3. Методы решения задачи.	5
4. Программная реализация.	7
5. Результаты работы.	30
6. Руководство пользователя.	31
7. Заключение.	36
8. Список литературы.	36
9. Приложения.	38

Введение.

Наблюдение за состоянием здоровья необходимо для обеспечения гарантии первоначальной и последующей физической пригодности работников для выполнения поставленных перед ними профессиональных задач. На данный момент для контроля здоровья сотрудников применяется обязательный периодический медицинский осмотр с периодом проведения один раз в год. Однако, такого вида обследования может быть не достаточно в случаях, если специальность сотрудника связана с :

- вредными и(или) опасными производственными факторами.
- использованием технически сложных механизмов и устройств повышенной опасности.
- пищевой промышленностью.
- изменением условий труда.

В таких случаях необходимо проводить регулярный мониторинг здоровья сотрудника, чтобы отслеживать динамику его физического и психологического состояния и свести к минимуму вред, причиненный здоровью и трудовому потенциалу работника.

На текущий момент, существуют устройства личного пользования для мониторинга ключевых показателей жизненно важных функций. (фитнес-браслеты, умный часы и т.п.) На крупных производствах и в больших организациях наиболее распространены только устройства контроля рабочего времени сотрудника. Необходимо создать устройство, для регулярного определения состояния здоровья человека, которое в зависимости от потребности компаний может иметь разный набор датчиков измерения показателей, а также с понятной расшифровкой полученных данных в графическом приложении.

Постановка задачи и цели работы.

Задача :

Разработать устройство, для регулярного определения состояния здоровья человека, которое в зависимости от потребности компаний может иметь разный набор датчиков измерения показателей, а также с понятной расшифровкой полученных данных в приложении на ПК.

Цели :

Необходимо разработать :

1. Основные модули устройства :
 - Модуль с микроконтроллером.
 - Модуль с датчиками.
 - Модуль с батареей.
 - Модуль для датчика пульса и оксигенации.
2. ПО для микроконтроллера, которое позволяет считывать данных с датчиков, обрабатывать их и передавать их на сервер.
3. ПО для обработки полученной информации и получения диагноза.

Методы решения задачи.

Для решения задачи разобьем ее на этапы :

1. Создание модели устройства.
2. Написание ПО для считывания датчиков микропроцессором.
3. Написание ПО для обработки полученной информации на ПК.

Создание модели устройства.

Датчики и микроконтроллер сегментированы на модули. Модули напечатаны с использованием 3D принтера. Модули устройства располагаются по слоям :

1. Модуль с микроконтроллером.
 - Модуль с платой адаптера ESP32
2. Модуль со считывающими датчиками.
 - Часы реального времени RTC
 - Плата для хранения данных microSD
 - Гироскоп и акселерометр GY-521
3. Модуль с батареей.
 - Преобразователь напряжения в 3.3V
 - Адаптер питания
 - Li-ion батарея
4. Модуль со считывающим датчиков (реализован отдельно от подобного модуля, так как информация о пульсе будет получаться с пальца)
 - Датчик пульса и оксигенации MAX30102

ПО микроконтроллера для считывания информации с датчиков и передачи на сервер.

На этом этапе должна быть реализована система классов, которая отвечает за использование датчиков микроконтроллером, созданием пакета данных и последующей передачей через Bluetooth. Необходимый функционал каждого датчика описывается отдельным классом (для более удобной работы с ними). Для написания ПО используется фреймворк Arduino на языке C++.

ПО для обработки полученной информации и получения диагноза.

На этом этапе должна быть разработана программа, которая позволяет получать и анализировать полученные пакеты данных и строить на их основе вывод о текущем состоянии активности человека, используя машинное обучение. Для удобного использования программа оснащена пользовательским интерфейсом с учетом использования нескольких учетных записей для входа. Для написания интерфейса ПО используется фреймворк Tkinter. Язык программирования Python.

Программная реализация.

Структура программы для микропроцессора ESP32.

Программа микроконтроллера разделена на два модуля которые занимаются : отпавкой данных, получением данных.

Модуль с получением данных разделен на классы, каждый из которых занимается считыванием данных с одного типа устройства.

Модуль с отпавкой данных создают пакеты с записями с датчиков, которые отсылаются на ПК по технологии Bluetooth.

Программа реализована на языке программирования C++ и состоит из 6 файлов, 5 из которых являются заголовочными с реализацией классов, обеспечивающих работу датчиков и передачи данных и одного файла исходного кода.

- Файл "main.cpp"

В файле реализуется функция setup и loop, которая отвечает за запуск программы на микроконтроллере.

- Файл "ESP32_Time.h"

Файл является заголовочным, в нем находится объявление и реализация класса ESP32_Time для подключения и использования часов реального времени. Представлены следующие методы класса :

- Функция "Init"

Public метод класса. Запускает передачу данных по шине I2C для часов реального времени.

Входные данные : функция не принимает никаких параметров.

Выходные данные : функция не возвращает никаких параметров.

- Функция "GetCurrentDateString"

Public метод класса. Возвращает текущую дату и время в виде "Thu, 16 Apr 2020 18:34:56".

Входные данные : функция не принимает никаких параметров.

Выходные данные : функция возвращает объект класса String.

- Функция "GetCurrentTimeString"

Public метод класса. Возвращает текущую дату и время в виде "18:34:56"

Входные данные : функция не принимает никаких параметров.

Выходные данные : функция возвращает объект класса String.

- Функция "GetDay"

Public метод класса. Возвращает текущее значение дня.

Входные данные : функция не принимает никаких параметров.

Выходные данные : функция возвращает переменную типа int.

– Функция "GetMonth"

Public метод класса. Возвращает текущее значение месяца.

Входные данные : функция не принимает никаких параметров.

Выходные данные : функция возвращает переменную типа int.

– Функция "GetYear"

Public метод класса. Возвращает текущее значение года.

Входные данные : функция не принимает никаких параметров.

Выходные данные : функция возвращает переменную типа int.

– Функция "GetHour"

Public метод класса. Возвращает текущее значение часов.

Входные данные : функция не принимает никаких параметров.

Выходные данные : функция возвращает переменную типа int.

– Функция "GetMin"

Public метод класса. Возвращает текущее значение минут.

Входные данные : функция не принимает никаких параметров.

Выходные данные : функция возвращает переменную типа int.

– Функция "GetSec"

Public метод класса. Возвращает текущее значение секунд.

Входные данные : функция не принимает никаких параметров.

Выходные данные : функция возвращает переменную типа int.

– Функция "GetSec"

Public метод класса. Возвращает текущее дату и время в виде строки, переданной в качестве входного параметра.

Входные данные : функция принимает объект класса String.

Выходные данные : функция возвращает объект класса String.

● Файл "ESP32_SDcard.h"

Файл является заголовочным, в нем находится объявление и реализация класса ESP32_SDcard для подключения и использования SD кард-ридера. Представлены следующие методы класса :

– Функция "Init"

Public метод класса. Запускает передачу данных по шине SPI для SD кард-ридера.

Входные данные : функция не принимает никаких параметров.

Выходные данные : функция не возвращает никаких параметров.

- Функция "listDir"
Public метод класса. Выводит в SerialMonitor список директорий в указанном пути на карте до глубины, переданной входящим параметром. Используется для отладки.
Входные данные : функция принимает константный указатель на тип char и переменную типа uint8_t.
Выходные данные : функция не возвращает никаких параметров.
- Функция "createDir"
Public метод класса. Создает директорию на карте по указанному пути.
Входные данные : функция принимает константный указатель на тип char.
Выходные данные : функция не возвращает никаких параметров.
- Функция "removeDir"
Public метод класса. Удаляет директорию на карте по указанному пути.
Входные данные : функция принимает константный указатель на тип char.
Выходные данные : функция не возвращает никаких параметров.
- Функция "readFile"
Public метод класса. Читает файл в директории на карте по указанному пути и выводит в SerialMonitor. Используется для отладки.
Входные данные : функция принимает константный указатель на тип char.
Выходные данные : функция не возвращает никаких параметров.
- Функция "writeFile"
Public метод класса. Записывает сообщение в файл, находящийся в директории на карте по указанному пути.
Входные данные : функция принимает константный указатель на тип char и константный указатель на тип char.
Выходные данные : функция не возвращает никаких параметров.
- Функция "appendFile"
Public метод класса. Вставляет сообщение в конец файла, находящегося в директории на карте по указанному пути.
Входные данные : функция принимает константный указатель на тип char и константный указатель на тип char.
Выходные данные : функция не возвращает никаких параметров.

- Функция "renameFile"
Public метод класса. Изменяет название файла, находящегося в директории на карте по указанному пути.
Входные данные : функция принимает константный указатель на тип char и константный указатель на тип char.
Выходные данные : функция не возвращает никаких параметров.
- Функция "deleteFile"
Public метод класса. Удаляет файл, находящийся в директории на карте по указанному пути.
Входные данные : функция принимает константный указатель на тип char.
Выходные данные : функция не возвращает никаких параметров.
- Функция "testFileIO"
Public метод класса. Проверяет работу файловой системы в указанной директории. Выводит в SerialMonitor информацию для тестирования. Используется для отладки.
Входные данные : функция принимает константный указатель на тип char.
Выходные данные : функция не возвращает никаких параметров.
- Файл "ESP32_MAX30102.h"
Файл является заголовочным, в нем находится объявление и реализация класса ESP32_MAX30102 для подключения и использования датчика пульса и оксигенации MAX30102. Представлены следующие методы класса :
 - Функция "Init"
Public метод класса. Запускает передачу данных по шине I2C для датчика пульса и оксигенации MAX30102.
Входные данные : функция не принимает никаких параметров.
Выходные данные : функция не возвращает никаких параметров.
 - Функция "GetDataFromMAX30102"
Public метод класса. Заполняет массив с начальными значениями массивы красных и инфракрасных излучений.
Входные данные : функция не принимает никаких параметров.
Выходные данные : функция не возвращает никаких параметров.
 - Функция "UpdateArray"
Public метод класса. Обновляет массивы красных и инфракрасных излучений новыми данными с датчика.
Входные данные : функция не принимает никаких параметров.
Выходные данные : функция не возвращает никаких параметров.

- Функция "UpdateData"
Public метод класса. Вычисляет значения пульса и оксигенации.
Входные данные : функция не принимает никаких параметров.
Выходные данные : функция не возвращает никаких параметров.
- Функция "GetBPM"
Public метод класса. Возвращает значение пульса.
Входные данные : функция не принимает никаких параметров.
Выходные данные : функция возвращает объект класса String.
- Функция "bGetBPM"
Public метод класса. Возвращает значение пульса.
Входные данные : функция не принимает никаких параметров.
Выходные данные : функция возвращает переменную типа byte.
- Функция "fGetSpO2"
Public метод класса. Возвращает значение оксигенации.
Входные данные : функция не принимает никаких параметров.
Выходные данные : функция возвращает переменную типа float.

- Файл "ESP32_Gyro.h"

Файл является заголовочным, в нем находится объявление и реализация класса ESP32_Gyro для подключения и использования акселерометра и гироскопа GY-521. Представлены следующие методы класса :

- Функция "Init"
Public метод класса. Запускает передачу данных по шине I2C для акселерометра и гироскопа GY-521. Адрес указывается в качестве входящего параметра. Если пин AD0 на GND, адрес 0x68, иначе 0x69
Входные данные : функция принимает параметр типа int.
Выходные данные : функция не возвращает никаких параметров.
- Функция "Getwx"
Public метод класса. Возвращает значение акселерометра по оси X.
Входные данные : функция не принимает никаких параметров.
Выходные данные : функция возвращает переменную типа float.
- Функция "Getwy"
Public метод класса. Возвращает значение акселерометра по оси Y.
Входные данные : функция не принимает никаких параметров.
Выходные данные : функция возвращает переменную типа float.

- Функция "Getwz"
Public метод класса. Возвращает значение акселерометра по оси Z.
Входные данные : функция не принимает никаких параметров.
Выходные данные : функция возвращает переменную типа float.
- Функция "Getx"
Public метод класса. Возвращает значение гироскопа по оси X.
Входные данные : функция не принимает никаких параметров.
Выходные данные : функция возвращает переменную типа float.
- Функция "Gety"
Public метод класса. Возвращает значение гироскопа по оси Y.
Входные данные : функция не принимает никаких параметров.
Выходные данные : функция возвращает переменную типа float.
- Функция "Getz"
Public метод класса. Возвращает значение гироскопа по оси Z.
Входные данные : функция не принимает никаких параметров.
Выходные данные : функция возвращает переменную типа float.
- Функция "SerialPrint"
Public метод класса. Выводит значения датчика в SerialMonitor. Используется для отладки.
Входные данные : функция не принимает никаких параметров.
Выходные данные : функция не возвращает никаких параметров.
- Файл "ESP32_PackageController.h"
Файл является заголовочным, в нем находится объявление и реализация класса TPackage и PackageController.

Класс TPackage.

Используется для хранения данных пакета, их получения и записи. Представлены следующие методы класса :

- Функция "GetTime"
Public метод класса. Возвращает значение по индексу в массиве, который хранит дату начала записи пакета.
Входные данные : функция принимает параметр типа int.
Выходные данные : функция возвращает переменную типа byte.

- Функция "GetSpO2"
Public метод класса. Возвращает значение по индексу в массиве, который хранит значения оксигенации.
Входные данные : функция принимает параметр типа int.
Выходные данные : функция возвращает переменную типа указатель на float.
- Функция "GetWx"
Public метод класса. Возвращает значение по индексу в массиве, который хранит значения акселерометра по оси X.
Входные данные : функция принимает параметр типа int.
Выходные данные : функция возвращает переменную типа указатель на float.
- Функция "GetWy"
Public метод класса. Возвращает значение по индексу в массиве, который хранит значения акселерометра по оси Y.
Входные данные : функция принимает параметр типа int.
Выходные данные : функция возвращает переменную типа указатель на float.
- Функция "GetWz"
Public метод класса. Возвращает значение по индексу в массиве, который хранит значения акселерометра по оси Z.
Входные данные : функция принимает параметр типа int.
Выходные данные : функция возвращает переменную типа указатель на float.
- Функция "GetX"
Public метод класса. Возвращает значение по индексу в массиве, который хранит значения гироскопа по оси X.
Входные данные : функция принимает параметр типа int.
Выходные данные : функция возвращает переменную типа указатель на float.
- Функция "GetY"
Public метод класса. Возвращает значение по индексу в массиве, который хранит значения гироскопа по оси Y.
Входные данные : функция принимает параметр типа int.
Выходные данные : функция возвращает переменную типа указатель на float.

- Функция "GetZ"
Public метод класса. Возвращает значение по индексу в массиве, который хранит значения гироскопа по оси Z.
Входные данные : функция принимает параметр типа int.
Выходные данные : функция возвращает переменную типа указатель на float.
- Функция "GetPulse"
Public метод класса. Возвращает значение по индексу в массиве, который хранит значения пульса.
Входные данные : функция принимает параметр типа int.
Выходные данные : функция возвращает переменную типа указатель на byte.
- Функция "isFull"
Public метод класса. Возвращает булево значение. True, если пакет полностью заполнен, иначе False
Входные данные : функция не принимает никаких параметров.
Выходные данные : функция возвращает переменную типа bool.
- Функция "Reset"
Public метод класса. Обнуляет итераторы внутри класса для подготовки к записи следующего пакета.
Входные данные : функция не принимает никаких параметров.
Выходные данные : функция не возвращает никаких параметров.
- Функция "AddGyro"
Public метод класса. Добавляет в пакет одну запись с гироскопа и акселерометра. Возвращает булево значение успешности выполнения метода.
Входные данные : функция принимает 6 параметров типа float.
Выходные данные : функция возвращает переменную типа bool.
- Функция "AddPulse"
Public метод класса. Добавляет в пакет одну запись пульса с пульсоксиметра. Возвращает булево значение успешности выполнения метода.
Входные данные : функция принимает параметр типа byte.
Выходные данные : функция возвращает переменную типа bool.
- Функция "AddSpO2"
Public метод класса. Добавляет в пакет одну запись оксигенации с пульсоксиметра. Возвращает булево значение успешности выполнения метода.
Входные данные : функция принимает параметр типа byte.
Выходные данные : функция возвращает переменную типа bool.

– Функция "AddTime"

Public метод класса. Добавляет в пакет одну запись даты и времени начала записи пакета. Возвращает булево значение успешности выполнения метода.

Входные данные : функция принимает 6 параметров типа int.

Выходные данные : функция возвращает переменную типа bool.

Класс PackageController.

Используется для создания пакета в виде массива byte, который будет отправлен по Bluetooth. Представлены следующие методы класса :

– Функция "GetPack"

Public метод класса. Возвращает пакет byte.

Входные данные : функция не принимает никаких параметров.

Выходные данные : функция возвращает переменную типа указатель на byte.

– Функция "GetSize"

Public метод класса. Возвращает размер пакета.

Входные данные : функция не принимает никаких параметров.

Выходные данные : функция возвращает переменную типа int.

– Функция "CreatePack"

Public метод класса. Создает пакет по данным полученным входящим параметром в виде ссылки на TPackage.

Входные данные : функция принимает ссылку на объект класса TPackage.

Выходные данные : функция не возвращает никаких параметров.

Структура программы для графического пользовательского интерфейса на Python.

Данные приложения разделены на два модуля : графический интерфейс, обработка и считывание данных.

Графический интерфейс обеспечивает различное представление данных, полученных из приложения.

Модуль обработки и считывания данных обеспечивает параллельное считывание и обновления данных с устройства на ПК по технологии Bluetooth.

Программа реализована на языке программирования Python и состоит из 10 файлов, 4 из которых являются файлами, в которых реализован интерфейс пользователя. 1 файл для анимации файлов с расширением .gif, 4 файла для подключения к контроллеру, считывания и хранения данных и 1 файл для анализа данных с помощью машинного обучения.

- Файл "PagesClass.py"

В файле представлены объявление и реализация класса MainWindow, который содержит в себе контейнер страниц графического приложения и отвечает за их отрисовку. Представлены следующие методы класса :

- Функция "__init__"

Конструктор класса MainWindows. Инициализируются параметры главного экрана, а также создается контейнер, в котором будут храниться страницы нашего интерфейса.

Входные данные : функция не принимает никаких параметров.

Выходные данные : функция не возвращает никаких параметров.

- Функция "run"

Метод запускает цикл для отслеживания событий приложения.

Входные данные : функция не принимает никаких параметров.

Выходные данные : функция не возвращает никаких параметров.

- Функция "show_frame"

Метод отображает страницу из контейнера, определенную во входном параметре.

Входные данные : объект класса из контейнера страниц.

Выходные данные : функция не возвращает никаких параметров.

- Файл "MenuPages.py"

В файле представлены объявление и реализация классов, отвечающих за меню приложения. К ним относятся классы : WelcomePage, StartPage, EntryPage, RegisterPage.

Класс WelcomePage.

Отображает начальную страницу с приветствием. Представлены следующие методы класса :

- Функция "__init__"

Конструктор класса WelcomePage. Отвечает за создание и инициализацию объектов и виджетов страницы

Входные данные : функция принимает контейнер страниц класса MainWindow и объект класса MainWindow.

Выходные данные : функция не возвращает никаких параметров.

Класс StartPage.

Отображает страницу, в которой необходимо выбрать способ авторизации в приложении. Присутствуют два способа авторизации : Войти или зарегистрироваться в приложении Представлены следующие методы класса :

- Функция "__init__"

Конструктор класса StartPage. Отвечает за создание и инициализацию объектов и виджетов страницы

Входные данные : функция принимает контейнер страниц класса MainWindow и объект класса MainWindow.

Выходные данные : функция не возвращает никаких параметров.

Класс EntryPage.

Отображает страницу, в которой необходимо войти в приложение по существующим логину и паролю с целью авторизации. Представлены следующие методы класса :

- Функция "__init__"

Конструктор класса EntryPage. Отвечает за создание и инициализацию объектов и виджетов страницы

Входные данные : функция принимает контейнер страниц класса MainWindow и объект класса MainWindow.

Выходные данные : функция не возвращает никаких параметров.

- Функция "check_register_data"
 Отвечает за проверку корректности ввода данных авторизации.
 Входные данные : функция не принимает никаких параметров.
 Выходные данные : функция не возвращает никаких параметров.
- Функция "find_register_data"
 Отвечает за поиск данных авторизации в файле с данными авторизации пользователей. При нахождении возвращает булево значение True, иначе False.
 Входные данные : функция не принимает никаких параметров.
 Выходные данные : функция возвращает переменную типа bool.

Класс RegisterPage.

Отображает страницу, в которой необходимо создать аккаунт для авторизации. Представлены следующие методы класса :

- Функция "__init__"
 Конструктор класса StartPage. Отвечает за создание и инициализацию объектов и виджетов страницы
 Входные данные : функция принимает контейнер страниц класса MainWindow и объект класса MainWindow.
 Выходные данные : функция не возвращает никаких параметров.
- Функция "check_register_data"
 Отвечает за проверку корректности ввода данных авторизации.
 Входные данные : функция не принимает никаких параметров.
 Выходные данные : функция не возвращает никаких параметров.
- Функция "add_register_data"
 Добавляет данные зарегистрировавшегося пользователя в файл с данными авторизации.
 Входные данные : функция не принимает никаких параметров.
 Выходные данные : функция не возвращает никаких параметров.
- Функция "is_overwrite_data"
 Проверяет входящие данные регистрации на перекрытие с уже существующими данными в файле авторизации. Возвращает True если записи уже существует в файле авторизации.
 Входные данные : функция принимает объект класса str.
 Выходные данные : функция возвращает переменную типа bool.

- Файл "ProfilePages.py"

В файле представлены объявление и реализация классов, отвечающих за главную страницу профиля пользователя и подключение к устройству. К ним относятся классы : MainPagePage, SettingPage, ProfileDataPage.

Класс MainPagePage.

Отвечает за главную страницу профиля, на которой располагаются кнопки для навигации по приложению.

Кнопки:

1. "Настройки" открывает страницу с подключением к устройству.
2. "Здоровье" переводит на страницу с выбором отображений графиков о текущем состоянии.
3. "Мой профиль" позволяет посмотреть или добавить более подробную информацию о текущем пользователе.

Представлены следующие методы класса :

– Функция "__init__"

Конструктор класса MainPagePage. Отвечает за создание и инициализацию объектов и виджетов страницы

Входные данные : функция принимает контейнер страниц класса MainWindow и объект класса MainWindow.

Выходные данные : функция не возвращает никаких параметров.

Класс SettingPage.

Отвечает за подключение к устройству по выбранному порту и baudрейту.

Представлены следующие методы класса :

– Функция "__init__"

Конструктор класса SettingPage. Отвечает за создание и инициализацию объектов и виджетов страницы

Входные данные : функция принимает контейнер страниц класса MainWindow и объект класса MainWindow.

Выходные данные : функция не возвращает никаких параметров.

- Функция "connect_check"
 Проверяет поля с параметрами подключения на корректность. Включает или отключает доступ к кнопке подключения
 Входные данные : функция не принимает никаких параметров.
 Выходные данные : функция не возвращает никаких параметров.
- Функция "baud_select"
 Метод обслуживает виджет выбора baudрейта.
 Входные данные : функция не принимает никаких параметров.
 Выходные данные : функция не возвращает никаких параметров.
- Функция "update_coms"
 Метод обслуживает виджет выбора порта подключения. Доступные порты находятся автоматически.
 Входные данные : функция не принимает никаких параметров.
 Выходные данные : функция не возвращает никаких параметров.
- Функция "connection"
 Проверяет верность выбранных данных подключения и вызывает методы класса ConnectHandler, отвечающего за параллельное считывание данных с устройства, для установления соединения по Bluetooth.
 Входные данные : функция не принимает никаких параметров.
 Выходные данные : функция не возвращает никаких параметров.

Класс ProfileDataPage.

Отображает страницу с дополнительной информацией о пользователе.

Представлены следующие методы класса :

- Функция "__init__"
 Конструктор класса ProfileDataPage. Отвечает за создание и инициализацию объектов и виджетов страницы
 Входные данные : функция принимает контейнер страниц класса MainWindow и объект класса MainWindow.
 Выходные данные : функция не возвращает никаких параметров.
- Функция "delete_data"
 Очищает данные из полей с дополнительной информацией.
 Входные данные : функция не принимает никаких параметров.
 Выходные данные : функция не возвращает никаких параметров.

– Функция "fill_start_data"

Загружает в поля с дополнительной информацией данные из файла с записями.

Входные данные : функция не принимает никаких параметров.

Выходные данные : функция не возвращает никаких параметров.

– Функция "changedataprofile"

Делает поля с дополнительной информацией доступными для изменения пользователю.

Входные данные : функция не принимает никаких параметров.

Выходные данные : функция не возвращает никаких параметров.

– Функция "savedataprofile"

Сохраняет поля с дополнительной информацией в файл с записями.

Входные данные : функция не принимает никаких параметров.

Выходные данные : функция не возвращает никаких параметров.

- Файл "RTInfoPages.py"

В файле представлены реализация и объявление классов, отвечающих за страницы с состоянием пользователя. К ним относятся графики сердцебиения, оксигенации и анализ активности. Классы реализованные в файле : HealthPage, HeartBeatPage, SPO2BeatPage.

Класс HealthPage.

Отвечает за отрисовку страницы с меню для выбора отображения показаний. Меню содержит кнопки:

1. "Домой" перейти на предыдущую страницу.
2. "BPM" переводит на страницу с графиком о текущем сердцебиении.
3. "SPO2" переводит на страницу с графиком о текущей оксигенации.

Представлены следующие методы класса :

– Функция "__init__"

Конструктор класса HealthPage. Отвечает за создание и инициализацию объектов и виджетов страницы

Входные данные : функция принимает контейнер страниц класса MainWindow и объект класса MainWindow.

Выходные данные : функция не возвращает никаких параметров.

Класс HeartBeatPage.

Отвечает за отрисовку графиков функций с сердцебиением и вывода среднего сердцебиения. Представлены следующие методы класса :

- Функция "`__init__`"

Конструктор класса HeartBeatPage. Отвечает за создание и инициализацию объектов и виджетов страницы

Входные данные : функция принимает контейнер страниц класса MainWindow и объект класса MainWindow.

Выходные данные : функция не возвращает никаких параметров.

- Функция "`filter`"

Используется для фильтрации показаний сердцебиения. Входные данные : функция принимает массив данных сердцебиения и параметр типа `int`.

Выходные данные : функция не возвращает никаких параметров.

- Функция "`getdata`"

Осуществляет запрос данных о сердцебиении через определенный промежуток времени. Входные данные : функция принимает параметр типа `int`.

Выходные данные : функция не возвращает никаких параметров.

- Функция "`plot`"

Отрисовывает график сердцебиения. Принимает массив значений сердцебиения. Входные данные : функция принимает массив данных сердцебиения.

Выходные данные : функция не возвращает никаких параметров.

Класс SPO2BeatPage.

Отвечает за отрисовку графиков функций с оксигенацией и вывода средней оксигенации. Представлены следующие методы класса :

- Функция "`__init__`"

Конструктор класса SPO2BeatPage. Отвечает за создание и инициализацию объектов и виджетов страницы

Входные данные : функция принимает контейнер страниц класса MainWindow и объект класса MainWindow.

Выходные данные : функция не возвращает никаких параметров.

– Функция "filter"

Используется для фильтрации показаний оксигенации.

Входные данные : функция принимает массив данных оксигенации и параметр типа int.

Выходные данные : функция не возвращает никаких параметров.

– Функция "getdata"

Осуществляет запрос данных об оксигенации через определенный промежуток времени.

Входные данные : функция принимает параметр типа int.

Выходные данные : функция не возвращает никаких параметров.

– Функция "plot"

Отрисовывает график оксигенации. Принимает массив значений оксигенации.

Входные данные : функция принимает массив данных оксигенации.

Выходные данные : функция не возвращает никаких параметров.

• Файл "UsersData.py"

В файле представлены реализация и объявление класса WorkerUserData, занимающегося обслуживанием класса ProfileDataPage, а именно работа с файлами, хранящими дополнительную информацию о пользователе. Представлены следующие методы класса :

– Функция "__init__"

Конструктор класса WorkerUserData.

Входные данные : функция не принимает никаких параметров.

Выходные данные : функция не возвращает никаких параметров.

– Функция "find_data"

Находит позицию записи о пользователе в файле.

Входные данные : функция не принимает никаких параметров.

Выходные данные : функция возвращает переменную типа int.

– Функция "set_user_login"

Устанавливает логин, введенный пользователем.

Входные данные : функция принимает параметр объект класса str.

Выходные данные : функция не возвращает никаких параметров.

– Функция "write_data_in_file"

Записывает дополнительные данные о пользователе (Имя, фамилия, номер телефона, электронная почта, вес, рост, возраст) в файл.

Входные данные : функция принимает 6 параметров объект класса str.

Выходные данные : функция не возвращает никаких параметров.

– Функция "rewrite_data_in_file"

Перезаписывает дополнительные данные о пользователе (Имя, фамилия, номер телефона, электронная почта, вес, рост, возраст) в файл.

Входные данные : функция принимает 6 параметров объект класса str.

Выходные данные : функция не возвращает никаких параметров.

– Функция "get_download_data"

Возвращает дополнительные данные из файла.

Входные данные : функция не принимает никаких параметров.

Выходные данные : функция возвращает объект класса list.

- Файл "SensorsData.py"

В файле представлены реализация и объявление классов SensorDataWorker, ArraySensorDataWorker, отвечающие за создание пакетов данных и их сохранение в файлах с расширением .пру, которые будут использованы для обучения модели.

Класс SensorDataWorker. Занимается созданием и обработкой двумерных numpy массивов. Представлены следующие методы класса :

– Функция "__init__"

Конструктор класса SensorDataWorker.

Входные данные : функция не принимает никаких параметров.

Выходные данные : функция не возвращает никаких параметров.

– Функция "get_array"

Возвращает переменную с двумерным массивом, которая хранится как параметр класса.

Входные данные : функция не принимает никаких параметров.

Выходные данные : функция возвращает объект класса ndarray.

– Функция "pack_is_ready"

Возвращает булево значение, показывающее является пакет полным или нет.

Входные данные : функция не принимает никаких параметров.

Выходные данные : функция возвращает переменную типа bool.

– Функция "pack_is_ready"

Добавляет запись в массив.

Входные данные : функция принимает 6 параметров типа float.

Выходные данные : функция не возвращает никаких параметров.

- Функция "clear_np_arr"
Очищает массив.
Входные данные : функция не принимает никаких параметров.
Выходные данные : функция не возвращает никаких параметров.
- Функция "save_to_file"
Сохраняет ndarray в файл с расширением .пру
Входные данные : функция не принимает никаких параметров.
Выходные данные : функция не возвращает никаких параметров.
- Функция "load_sensor_data"
Загружает данные из файла в переменную типа ndarray.
Входные данные : функция не принимает никаких параметров.
Выходные данные : функция не возвращает никаких параметров.

Класс SensorDataWorker. Занимается созданием и обработкой трехмерных numpy массивов из двумерных numpy массивов. Представлены следующие методы класса :

- Функция "__init__"
Конструктор класса ArraySensorDataWorker.
Входные данные : функция не принимает никаких параметров.
Выходные данные : функция не возвращает никаких параметров.
- Функция "add_pack"
Добавляет в массив один двумерный массива типа ndarray.
Входные данные : функция принимает двумерный массив типа ndarray.
Выходные данные : функция не возвращает никаких параметров.
- Функция "get_array"
Возвращает переменную с трехмерным массивом, которая хранится как параметр класса.
Входные данные : функция не принимает никаких параметров.
Выходные данные : функция возвращает объект класса ndarray.
- Функция "save_file"
Сохраняет ndarray в файл с расширением .пру
Входные данные : функция не принимает никаких параметров.
Выходные данные : функция не возвращает никаких параметров.
- Функция "load_sensor_data"
Загружает данные из файла в переменную типа ndarray.
Входные данные : функция не принимает никаких параметров.
Выходные данные : функция не возвращает никаких параметров.

- Файл "Heart.py"

В файле представлены реализация и объявление класса HeartGif, который занимается созданием и отрисовкой файла с расширением .gif. Файлы типа .gif разделяются по картинкам и записываются в массив картинок. Анимация происходит в виде быстрой замены картинок из массива. Представлены следующие методы класса :

- Функция "__init__"

Конструктор класса HeartGif.

Входные данные : объект родительского класса, положение в разметке страницы (2 параметра типа int) и размеры (2 параметра типа int).

Выходные данные : функция не возвращает никаких параметров.

- Функция "update"

Изменяет текущую картинку на следующую через определенный момент времени, указанный входным параметром.

Входные данные : функция принимает параметр типа int.

Выходные данные : функция не возвращает никаких параметров.

- Файл "GlobalVariables.py"

В файле содержатся глобальные переменные, используемые программой.

- Файл "ConnectHandler.py"

В файле представлены реализация и объявление класса ConnectHandler, который занимается параллельным считыванием данных с устройства.

- Функция "__init__"

Конструктор класса ConnectHandler.

Входные данные : функция не принимает никаких параметров.

Выходные данные : функция не возвращает никаких параметров.

- Функция "connect"

Обеспечивает подключение по Bluetooth по входным параметрам.

Входные данные : функция принимает 2 объекта класса str.

Выходные данные : функция не возвращает никаких параметров.

- Функция "createthread"

Создает новый поток.

Входные данные : функция не принимает никаких параметров.

Выходные данные : функция не возвращает никаких параметров.

- Функция "readSerial"
Считывает данные из Serial Port и записывает в локальные переменные.
Входные данные : функция не принимает никаких параметров.
Выходные данные : функция не возвращает никаких параметров.
- Функция "GetSpO2"
Возвращает list с данными оксигенации. Входные данные : функция не принимает никаких параметров.
Выходные данные : функция возвращает объект класса list.
- Функция "GetPulse"
Возвращает list с данными пульса. Входные данные : функция не принимает никаких параметров.
Выходные данные : функция возвращает объект класса list.
- Функция "GetAx"
Возвращает list с данными акселерометра по оси X. Входные данные : функция не принимает никаких параметров.
Выходные данные : функция возвращает объект класса list.
- Функция "GetAy"
Возвращает list с данными акселерометра по оси Y. Входные данные : функция не принимает никаких параметров.
Выходные данные : функция возвращает объект класса list.
- Функция "GetAz"
Возвращает list с данными акселерометра по оси Z. Входные данные : функция не принимает никаких параметров.
Выходные данные : функция возвращает объект класса list.
- Функция "GetGx"
Возвращает list с данными гироскопа по оси X. Входные данные : функция не принимает никаких параметров.
Выходные данные : функция возвращает объект класса list.
- Функция "GetGy"
Возвращает list с данными гироскопа по оси Y. Входные данные : функция не принимает никаких параметров.
Выходные данные : функция возвращает объект класса list.
- Функция "GetGz"
Возвращает list с данными гироскопа по оси Z. Входные данные : функция не принимает никаких параметров.
Выходные данные : функция возвращает объект класса list.

- Функция "GetTime"
Возвращает list с данными даты и времени начала записи пакета. Входные данные : функция не принимает никаких параметров.
Выходные данные : функция возвращает объект класса list.
- Функция "data_destroy"
Очищает локальные переменные класса с данными.
Входные данные : функция не принимает никаких параметров.
Выходные данные : функция не возвращает никаких параметров.
- Функция "data_destroy"
Закрывает Serial Port и останавливает передачу данных.
Входные данные : функция принимает объект класса str.
Выходные данные : функция не возвращает никаких параметров.
- Функция "getconditionthread"
Возвращает состояние потока.
Входные данные : функция не принимает никаких параметров.
Выходные данные : функция переменную типа bool.
- Функция "setconditionthread"
Изменяет состояние потока.
Входные данные : функция принимает один параметр типа bool.
Выходные данные : функция не возвращает никаких параметров.

- Файл "ConnectHandler.py"

В файле представлены реализация и объявление классов ActivePrediction и PredictHandler, которые занимаются анализом активности человека на основе записей акселерометра и гироскопа с помощью модели машинного обучения.

Класс ActivePrediction содержит в себе методы для работы с моделью машинного обучения.

- Функция "__init__"
Конструктор класса ActivePrediction.
Входные данные : функция не принимает никаких параметров.
Выходные данные : функция не возвращает никаких параметров.
- Функция "create_model"
Создает модель обучения на основе 3 объектов массивов, отражающих разные состояние активности человека.
Входные данные : функция принимает 3 объекта класса str.
Выходные данные : функция не возвращает никаких параметров.

- Функция "features"
Подготавливает массив данных. Входные данные : функция принимает 1 объекта класса ndarray.
Выходные данные : функция возвращает 1 объект класса ndarray.
- Функция "load_model"
Загружает модель из файла в локальную переменную.
Входные данные : функция не принимает никаких параметров.
Выходные данные : функция не возвращает никаких параметров.
- Функция "predict"
Дает пример на вход модели и возвращает значение активности.
Входные данные : функция принимает 1 объект типа ndarray.
Выходные данные : функция возвращает 1 объект типа list.
- Функция "predict"
Преобразует объект класса ndarray в пригодный для передачи как параметр.
Входные данные : функция принимает 1 объекта класса ndarray.
Выходные данные : функция возвращает 1 объект класса ndarray.

Класс-обертка PredictHandler содержит в себе методы для работы с классом ActivePrediction.

- Функция "__init__"
Конструктор класса ActivePrediction.
Входные данные : функция не принимает никаких параметров.
Выходные данные : функция не возвращает никаких параметров.
- Функция "next_predict"
Дает модели следующий пакет данных.
Входные данные : функция не принимает никаких параметров.
Выходные данные : функция не возвращает никаких параметров.
- Функция "return_value"
Преобразует значение активности из значение в строку описывающее это значение. Входные данные : функция принимает 1 объект класса list.
Выходные данные : функция возвращает 1 объект класса str.

Результаты работы.

В ходе работы были проведены реальные эксперименты. На протяжении нескольких дней производилось тестирование устройства. Испытуемый симулировал разное состояние активности человека (активное, средняя активность, не активен), а так же занимался определенной физической нагрузкой. Для проверки точности измерений использовался пульсоксиметр.

В ходе тестирования было выявлено неточное вычисление измерений датчика оксигенации. Для получения показаний использовался прибор, в котором присутствовали только светодиоды с красным и инфракрасным излучением. Был сделан вывод, что для точного получения показаний необходимо использовать датчик насыщения крови кислородом со встроенным зеленым светодиодом. Поэтому следующей нашей целью станет замена датчика пульса и оксигенации МАХ30102 на МАХ30105 и последующее его интегрирование в основной модуль устройства, чтобы не использовать кольцо.

Кроме того, эксперименты показали, что анализ состояние активности человека на основе акселерометра и гироскопа с помощью машинного обучения дает точно результатов около 90%, поэтому следует модифицировать модель машинного обучения на анализ данных с других датчиков устройства.

Руководство пользователя.

При запуске программы появляется графический интерфейс с экраном приветствия(Рис. 1).

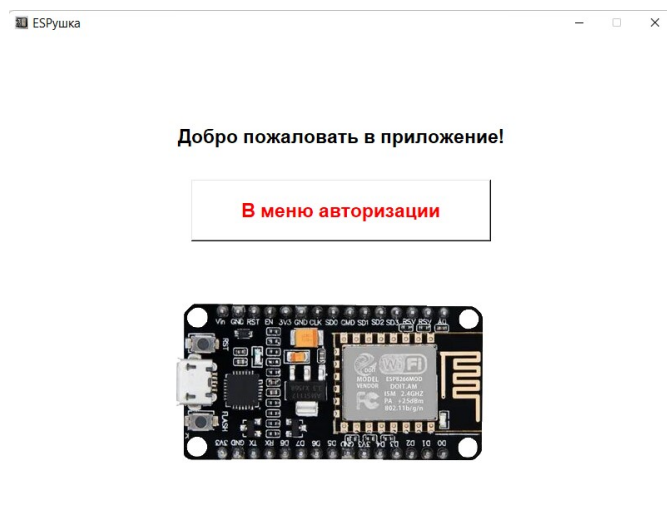


Рис. 1: Экран приветствия.

После нажатия на кнопку войти пользователь перейдет на экран авторизации(Рис. 2), где ему будет предложен выбор войти с учетом имеющегося аккаунта или зарегистрироваться(Рис. 3).

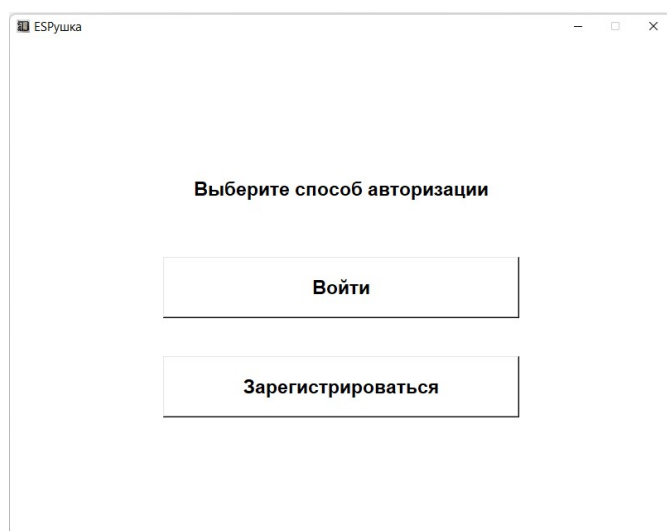




Рис. 2: Экран выбора авторизации.

ESПушка

Войдите в аккаунт





(а) Экран входа.

ESПушка

Создание аккаунта

Придумайте Логин *

Придумайте Пароль *

Повторите Пароль *

(b) Экран регистрации.

Рис. 3: Экран авторизации.

После авторизации пользователь попадает на главный экран своего профиля (Рис. 4), где можно перейти на окно просмотра дополнительной информации о своем аккаунте (Рис. 5) или перейти в настройки и начать подключение к устройству (Рис. 6).

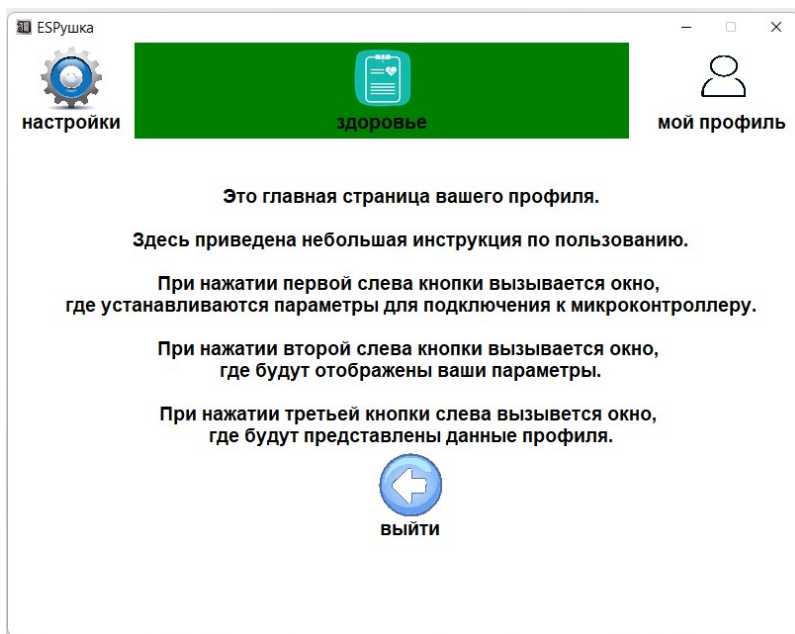


Рис. 4: Экран главного профиля.

The screenshot shows the additional information screen of the ESPushka application. At the top left, there is a house icon and the text 'home'. Below this, there is a list of labels for user information: 'Имя' (Name), 'Фамилия' (Surname), 'Телефон' (Phone), 'Email', 'Вес' (Weight), 'Рост' (Height), and 'Возраст' (Age). Each label is followed by a corresponding empty text input field. At the bottom of the screen, there are three buttons: 'Изменить данные' (Change data), 'Сохранить данные' (Save data), and 'Обновить' (Update).

Рис. 5: Экран доп. информации.

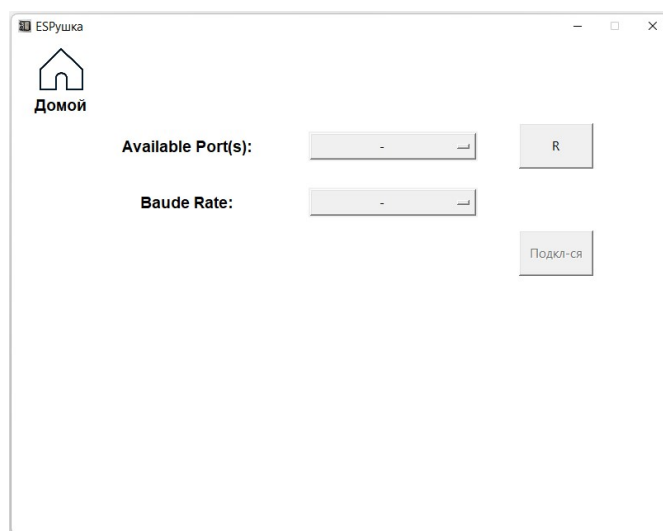


Рис. 6: Экран настроек.

После подключения к устройству, необходимо перейти в окно здоровье, где пользователь может посмотреть график своего пульса или оксигенации (Рис. 7 и 8).

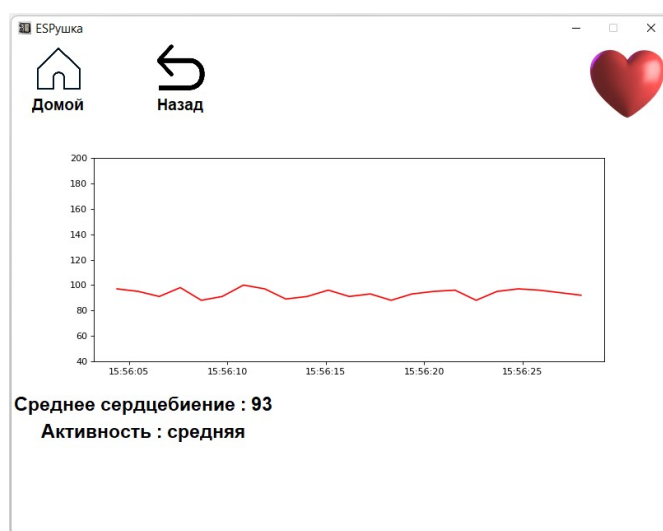


Рис. 7: Экран графика пульса.

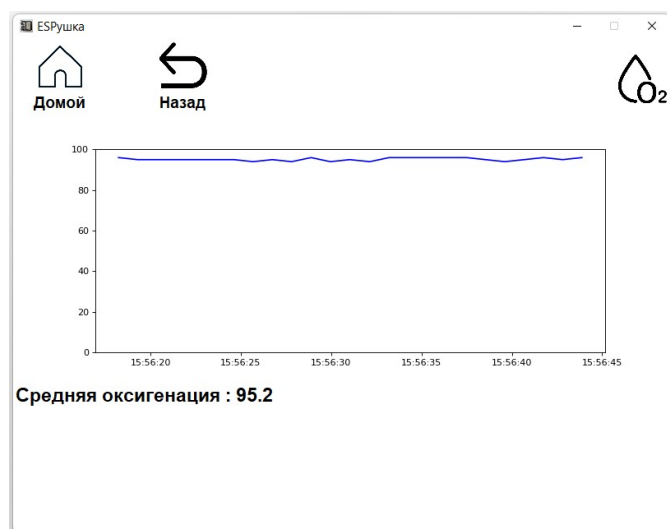


Рис. 8: Экран графика оксигенации.

Заключение.

В результате были разработано устройство для регулярного определения состояния здоровья человека. В ходе экспериментов были обнаружены недостатки связанные с неточностью датчиков для измерения, в связи с чем показания прибора могли быть неточными.

Также нами была написана программа, которая принимает полученные данные и анализирует их с помощью модели машинного обучения. Полученные показания визуализируются с помощью удобного для восприятия графического программного интерфейса.

Список литературы

- [1] Документация к фреймворку Tkinter <https://docs.python.org/3/library/tkinter.html>
- [2] "ESP32 Datasheet"(PDF). Espressif Systems. 2017-03-06. Retrieved 2017-03-14.
- [3] "Microcontroller Maniacs Rejoice: Arduino Finally Releases the 32-Bit Due". Wired. Retrieved 20 February 2018.
- [4] "Getting Started: FOUNDATION > Introduction". arduino.cc. Archived from the original on 2017-08-29. Retrieved 2017-05-23.
- [5] J. M. Hughes: Arduino: A Technical Reference. O'Reilly Media, 2016, ISBN 978-1-4919-3450-0, S. 122 ff.
- [6] McKinney, Wes (2017). Python for Data Analysis : Data Wrangling with Pandas, NumPy, and IPython (2nd ed.). Sebastopol: O'Reilly. ISBN 978-1-4919-5766-0.
- [7] VanderPlas, Jake (2016). "Introduction to NumPy". Python Data Science Handbook: Essential Tools for Working with Data. O'Reilly. pp. 33–96. ISBN 978-1-4919-1205-8.

Приложения.

Код приложения находится в репозитории :
https://github.com/danielbulgakov/ITlab_Project