Project

A fitness monitoring IoT application

Artur Costa
Informatic Department
Faculty of Sciences of the University of Lisbon
Lisbon, Portugal
fc62740@alunos.fc.ul.pt

Enzo Sousa
Informatic Department
Faculty of Sciences of the University of Lisbon
Lisbon, Portugal
fc62744@alunos.fc.ul.pt

***Abstract* — This project focuses on creating an IoT application for monitoring and analysing personal performance in outdoor activities. Using an IoT device, an iPhone on the wrist, to collect accelerometer and gyroscope data, we recorded measurements on all three axes, complemented with detailed temporal information. To facilitate interpretation of the data, we developed an intuitive dashboard that allows users to visualize their performance and information related to physical activities in a clear and easy way.**

Keywords* — *IoT, application, monitoring, activities, accelerometer, gyroscope, data, dashboard, devices.

I. INTRODUCTION

As we know, the Internet of Things (IoT) refers to the interconnected network of devices that communicate with each other and exchange data via the Internet. Currently, the IoT has been widely applied in various sectors. Specifically, and in relation to our project, it has had a great application in personal monitoring, including fitness and consequently in healthcare, as it plays a crucial role in improving the overall fitness experience, thus promoting a healthier lifestyle.

Portable devices, including fitness trackers, smartwatches and other devices, have become indispensable on citizens' fitness journeys. These devices are equipped with a sophisticated set of sensors, actuators and connectivity features that collectively form a comprehensive ecosystem designed to monitor and improve various aspects of a citizen's health and physical activity.

Portable devices: Portable devices are equipped with sensors to monitor various health parameters, such as heart rate, steps taken, calories burned and much more. These devices act as the main data collection points for a citizen's physical activity.

Sensors and actuators: IoT fitness devices are equipped with a variety of sensors, including accelerometers, gyroscopes, heart rate monitors and GPS modules. These sensors collect data in real time, which is then processed to obtain meaningful information about a citizen's physical activity and health.

Data processing and analysis: The collected data is sent to a centralized system or a cloud infrastructure for processing and analysis. Advanced analysis algorithms can provide personalized recommendations, information on training performance and trends over time. This data-driven approach helps users make informed decisions about their fitness routines.

Connectivity protocols: IoT devices use various communication protocols, such as Bluetooth, Wi-Fi or cellular networks, to connect and transmit data. This allows for seamless communication between devices.

Relating what has been said about the integration of IoT and fitness to our project, we didn't use portable devices, sensors and actuators and connectivity protocols (in other words, we didn't use these devices directly to collect the data), but we did use portable devices, sensors and actuators and connectivity protocols to collect the data and then analyse and process it (in order to obtain the values that we can see in the training). With regard to data processing and analysis, this process is carried out in our project, and is developed and detailed later in this report.

II. PROJECT OVERVIEW

In this project, an IoT application dedicated to monitoring and analysing personal performance in leisure activities is developed. As I said earlier in the introduction, an IoT device (an iPhone) was worn on the wrist (but not used directly by us) to collect accelerometer and gyroscope data (measurements from all three axes are collected). In addition, the data set is supplemented with temporal information, detailing the exact date and time associated with each measurement.

The main objective of this project is to understand the daily variations in user performance. We will explore how much the user runs and walks daily, analysing distances covered, the duration of activities and the number of calories consumed during walks and runs. In addition, we determine statistical information by week and month.

To make it easier for users to interpret and interact with the data collected, an intuitive dashboard was created. This dashboard allows users to easily and clearly visualize their performance and physical activity data.

III. PROJECT DEVELOPMENT

In this section, we'll mention the tools used during the development of our project.

A. *Docker*

Docker is a versatile containerization tool that empowers developers by providing a comprehensive environment for creating, distributing and running containerized applications, aligning perfectly with the microservices architecture paradigm. This approach encourages the decomposition of complex applications into smaller, independently deployable components, promoting modularity, scalability and maintainability. Docker containers offer a lightweight and consistent runtime environment, encapsulating the application code and all the necessary dependencies, libraries and configurations. This isolation guarantees consistent behaviour in different environments, from development to production, simplifying the development lifecycle and facilitating collaborative development efforts.

B. *MQTT Broker*

The MQTT (Message Queuing Telemetry Transport) Broker is a lightweight and efficient messaging protocol designed for communication between devices and is a fundamental component in communication architectures. The MQTT Broker acts as an intermediary between the devices that publish (MQTT publisher) and the devices that subscribe to messages (MQTT subscriber), facilitating the exchange of information.

C. *MySQL*

MySQL is an open-source relational database management system (RDBMS) that uses a Structured Query Language (SQL). This language is used to define, manipulate and query data.

D. *Flutter*

Flutter is an open-source application development framework. It is used to build user interfaces for mobile devices, web and desktop from a single source code. Flutter uses the Dart programming language and is designed to offer an efficient and consistent development experience on various platforms.

E. *Python*

To carry out this project, we used the Python programming language. We will now present the libraries used throughout the development of this project:

- The "requests" library is used to make HTTP requests. It simplifies the process of sending HTTP requests to web servers and makes it easier to handle responses.
- The "Flask" library is a web framework used to develop web applications. It makes it easy to create web applications quickly, flexibly and with a simple syntax. Flask is used to build everything from small APIs to more complex web applications.
- The "mysql-connector-python" library is a connector used to interact with MySQL databases using the Python language. It simplifies communication between Python applications and MySQL servers, allowing SQL queries to be executed.
- The "numpy" library is used for numerical and scientific computing. It provides support for multidimensional arrays and other advanced mathematical operations.
- The "pandas" library is a powerful tool for data manipulation and analysis. It provides flexible and efficient data structures designed to make it easier to clean, transform and analyse data sets.

IV. PROJECT ARCHITECTURE

We were told that, as we developed our project, we would have to implement 5 services. We will now show and explain what these 5 services consist of.

A. *First Service*

The first service involved implementing an "MQTT Broker", a key component for enabling communication using the MQTT protocol. For this first service, we followed the instructions provided by the teachers (install, configure and run the Mosquitto Broker). The "MQTT Broker" was then configured to listen on port 1883 (the standard MQTT port) and to be accessed remotely (from the main machine).

B. *Second Service*

The second service included an "MQTT subscriber" responsible for receiving data from the MQTT Broker, and the implementation of a "Processor" service, in charge of handling and processing the data received. This "Processor" plays an essential role in the system by performing various functions:

- Data loading: The "Processor" loads the training data set, ensuring that the system can access the data required for the machine learning processes.
- Interaction with the database: The "Processor" interacts with the database service, transmitting the input data for storage.
- Machine Learning Processing: The "Processor" will call the "ProcessorML" service to start training machine learning models. This step is essential for improving forecasting capabilities.
- Control panel updates: "Processor" updates the information in the Dashboard service, ensuring that the latest data and information is shown to users in an interface.

C. *Third Service*

The third service had to do with the Database (DB) service, a crucial component responsible for storing and retrieving the data processed by the "Processor". In this service, we had some responsibilities to consider.

- Database engine selection: We had to carefully select a suitable database engine based on the project's requirements (in this case we selected MySQL).
- Installation: Once selected, we had to proceed with the installation in the docker environment. This involved deploying the chosen database software and ensuring that it was correctly configured and accessible.
- Storage and retrieval configuration: Designing and implementing the necessary data tables, defining data structures and establishing efficient data access mechanisms.
- Integration with the Processor service: Integration between the database and the Processor services, ensuring that data can be transmitted from the Processor for storage and retrieval.

D. Fourth Service

The fourth service was the "ML Processor", a service dedicated to training, testing and validating machine learning (ML) models. This service enables the development of predictive models based on online input data to understand whether a user is walking or running.

E. Fifth Service

The fifth and final service (Dashboard service) represented the user interface component of our project. The technology suggested for implementing the dashboard functionality had been the Node-RED server. However, we ended up using another tool, Flutter, which will explain why we opted for this tool later.

In addition to the 5 services, an architecture was proposed to us, which we then analysed and came to the conclusion that it was the architecture we would use to implement our project.

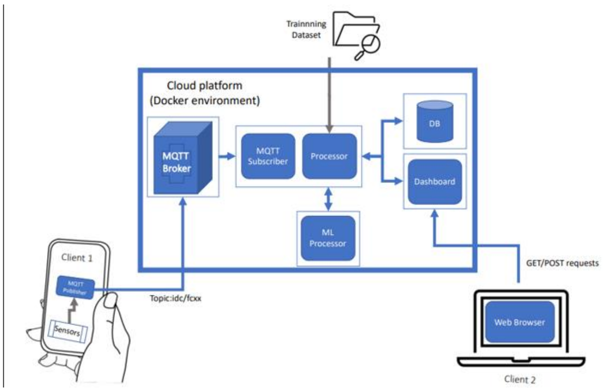


Figure 1- Project Architecture

At the start of the flow, the data is processed. The "Processor" component uses the training data set to generate models, specifically opting for the training-DT algorithm due to the promising results it has shown. The generated models are then forwarded to the "ML Processor" service with the relevant data.

Simultaneously, an "MQTT publisher" is responsible for sending topics to the "MQTT Broker". On the other side, an MQTT subscriber reads the data from the MQTT publisher and sends the data to the ML Processor. This MQTT subscriber plays a crucial role in the efficient exchange of information between the system components.

The Processor receives the data from the MQTT subscriber, carries out the necessary processing and sends the data to the ML Processor service. In addition, the "Processor" takes on the additional responsibility of sending the already processed data to a database, guaranteeing storage.

A distinctive feature of your system is that the "Processor" is not only an internal task executor, but also serves as an API. This function eliminates the need for an intermediary dashboard, establishing direct communication between the Processor and the Web Browser.

V. Data analysis, processing and storage

A web application was developed using Flask. Initially, we used a file called "training-DT" to access the training data. After properly processing this data, we sent it to a database.

The application has an endpoint called "/predict", configured to accept data via the POST method. When it receives data for analysis at this endpoint, it loads a Decision Tree model (training-DT) previously trained using the “joblib” library. It then makes an activity prediction based on this data and the model, and all the resulting information is sent to the database.

In addition, the Flask application is connected to a database, making it possible to store both the training data and the forecast results. This approach creates a continuous flow, where the application not only trains the initial model, but also provides predictions in real time, maintaining an organisation over the information presented in the database.

```
@app.route('/predict', methods=['POST'])
def predict():
    feature_dict = request.get_json()
    if not feature_dict:
        return {
            'error': 'Body is empty.'
        }, 500

    try:
        data = []
        model_name = feature_dict[0]['model']
        model = joblib.load('model/' + model_name + '.dat.gz')
        data.append(feature_dict[1])
        response = get_model_response(data, model)
    except ValueError as e:
        return {'error': str(e).split('\n')[-1].strip(),
                'message': str(e)}, 500

    return response, 200
```

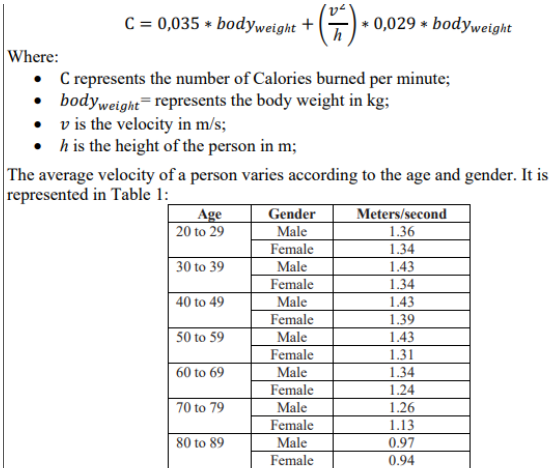
Figure 2- Predict POST Request

VI. Analysing the Web Browser

Here we'll show you how our interface is composed, what results we can see in it and the possible interactions.

Figure 3 - User Information

To start using the application, we need to enter the user's data so that we can relate it to the table and the formula given by the teachers to calculate the calories burned.



$$C = 0{,}035 * body_{weight} + \left(\frac{v^2}{h}\right) * 0{,}029 * body_{weight}$$

Where:

- C represents the number of Calories burned per minute;
- $body_{weight}$= represents the body weight in kg;
- v is the velocity in m/s;
- h is the height of the person in m;

The average velocity of a person varies according to the age and gender. It is represented in Table 1:

Age	Gender	Meters/second
20 to 29	Male	1.36
	Female	1.34
30 to 39	Male	1.43
	Female	1.34
40 to 49	Male	1.43
	Female	1.39
50 to 59	Male	1.43
	Female	1.31
60 to 69	Male	1.34
	Female	1.24
70 to 79	Male	1.26
	Female	1.13
80 to 89	Male	0.97
	Female	0.94

Figure 4 - Values to calculate calories

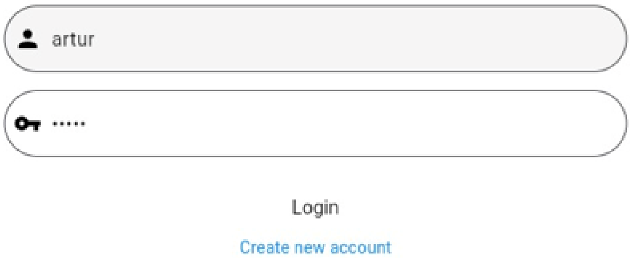


Figure 5 - Log in

After entering the user's information, the user logs in to view the data. If the user doesn't already have an account, they first create one and then log in.

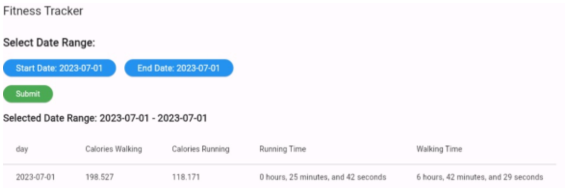


Figure 6 - Data Visualisation (1)

Once logged in, this is what the user can see. As you can see, right from the start you can select a start date and an end date to visualise your activity over a given time interval. When you select the desired time interval, what you see is a table showing the day you selected, the calories you burned walking and running and the time you spent running and walking during that time interval.

Figure 7 - Data Visualisation (2)

In this figure, we see the data in real time. As you can see, we show the date we started training, we show the total training time, we show the total distance travelled in that training and we show how fast the user is travelling.

VII. Challenges Faced

Throughout this project, we've come across some difficulties. We'll show you what they were and in the next topic we'll show you the solutions we found to the difficulties we encountered.

The only difficulty we encountered was in using the suggested technology, Node-Red. At first, we managed to implement the basics that were requested (what the Dashboard was supposed to present to the user), but we couldn’t fetch the data in real time to make it available. Since we were spending a lot of time trying to solve this problem, we decided not to use Node-Red and to use another tool.

VIII. Solutions Found

We'll now show you the solutions we found to the difficulties mentioned above.

To get round the difficulty of using Node-Red, we used the Flutter framework. The advantage of having used Flutter is that we were already familiar with this framework, while Node-Red was new to us, making it much easier to develop the application. In this project (and everywhere else Flutter is used), it has the role of helping to develop the front-end. By assisting in the development of the front-end, Flutter's intuitive and more accessible syntax allowed us to create a control panel more easily.

IX. Future Works

If we had more time to implement and develop this project, we believe we could realise an even better project (more detailed and more complex) than the one we did. By this I mean mainly the front-end and back-end.

X. What Wasn't Implemented

We were unable to correctly time the time we ran and the time we walked.

XI. Conclusion

The development of this project has given us an overview of the potential and challenges that this currently evolving field presents.

The implementation of the IoT architecture, highlighting all the components used for the correct operation of this project, provided correct management of data flows. The application provides information on the physical activity carried out by the user.

The "training-DT" algorithm used to predict activity (whether the user was running or walking) showed positive results. Despite the challenges encountered, we managed to overcome them in a positive way, fulfilling everything that was asked of us (apart from the part we noted in the report).