Original Project: Biometric Data Collection for Performance Optimization in a Digital Game Scenario

Continuation: Chart API – Display of test results

Otito Mbelu

Rodrigo Almeida

BSc. Computing in Software Development

Year 3 Project

https://github.com/rodAlm08/chart_mock.git

Supervisor: Damien Costello

Department of Computer Science and Applied Physics

Atlantic Technological University - (ATU)

Ollscoil Teicneolaíochta an Atlantaigh

Atlantic Technological University

Contents

1. Introduction

This project is based on the previous work done by final year students of Computing in Software Development. It is a PC based first-person shooter game, designed as a testing platform for measuring user performance with respect to their physical condition.

The system involves gathering information about the user's physical activities through a smartwatch and trying to correlate it with their performances during gameplay.

The end goal of this research is to be able to reliably predict a user's performance given the most recent physical activities of the user and offer reliable suggestions for in-game configuration using previous test results as a benchmark.

1.1 Overview

The system consists of three major components that interact together to produce, process, and analyse data. The most significant component of the system is the User Application (the game itself). The user can play the game in various configurations and scenario to measure select relevant skills for PUIG. To better simulate real life conditions viz a viz PUIG gameplay, users can import their PUIG configuration into the test platform and can also make relevant adjustments when required. Other components of the system include a Backend web application and Smart Watch. The Backend application was designed specifically for Auth2.0 authentication call-back server required to successfully connect the smart watches to the system and pull information about the user's physical activities.

1.2 Background

At the start of this project, the application has already been designed and built and the various components of the system were supposedly running nominally. However, it turned out that the server-side part of the system was already out of commission before the commencement of the project. This was due to the service provider Hereoku, making changes to their terms of service which adversely affected the web application, and was subsequently taken offline by Hereoku.

As a result, the web application was reconfigured and redeployed to a different service provider (Amazon AWS EC2 Server). Other relevant changes were also made to the main application to integrate the whole system.

Our task was to improve the user interface and make it more readable and user-friendly. Specifically, we were responsible for designing a set of visualizations that would clearly and effectively display the test results. To achieve this goal, we decided to create a new API that would integrate with the existing project at a later time.

To develop the API, we began by reading an existing JSON file from previous tests. We were able to generate the graphs dynamically based on the data contained in the JSON file. In order to display the data in the chart, we created a bar chart controller script that ordered the data by date and determined the minimum and maximum values that would be displayed in the graph. Finally, we utilized a cube object to draw the bars and LineRenderer to draw the lines that make up the chart.

Our group was tasked with enhancing the user interface by creating a new API that would integrate with the existing project and generate a dynamic chart based on the test data. The work should improve the overall usability of the system, providing a more intuitive and informative user interface.

1.3 Scope

At the inception of the project, a meeting was held with the product owner where the brief was extensively discussed, and requirements clarified. All relevant documentation was handed over and authorizations were granted to the git repository and Firebase Database for seamless access. From our interactions with the product owner, he identified some components of the application which were to be redesigned to enhance the overall outlook of the system. The various components to be redesigned were the charts displaying the user's test results for the various categories of tests namely, Fine Motor test, Audio test and Visual test. With the current configuration of the application, at the end of each test play the user gets to view a chart showing history for a default period of time up to the last test. These charts were, however, displayed separately on different scenes/windows and required switching scenes to view other test categories.

By the end of our discussions, the scope of the project was limited to merging all the charts for the various test categories into a unified single graph. This is intended to form the basis of further research by gracefully displaying data from other test categories side by side in a single chart where digressions and trends will be laid bare for intuitive data analysis.

Down the line, the scope of the project was further expanded to accommodate the reconfiguration and redeployment of the backend application after the initial service providers withdrew their services.

1.4 Team Structure and Roles

Our project was carried out by a team of two members, each with distinct roles and responsibilities.

Rodrigo took the role of the lead developer and project manager, overseeing the technical aspects of the project and ensuring that the project was completed within the timeline. He also collaborated with the product owner and project supervisor to monitor the project's progress and assisted in the development and testing of the API.

On the other hand, Otito was responsible for the design and development of the user interface, creating wireframes and mock-ups, and implementing the user interface using Unity's UI system. Additionally, he contributed to API testing and documentation.

Overall, our team structure facilitated seamless collaboration between members and ensured that both technical and design aspects of the project were given appropriate attention.

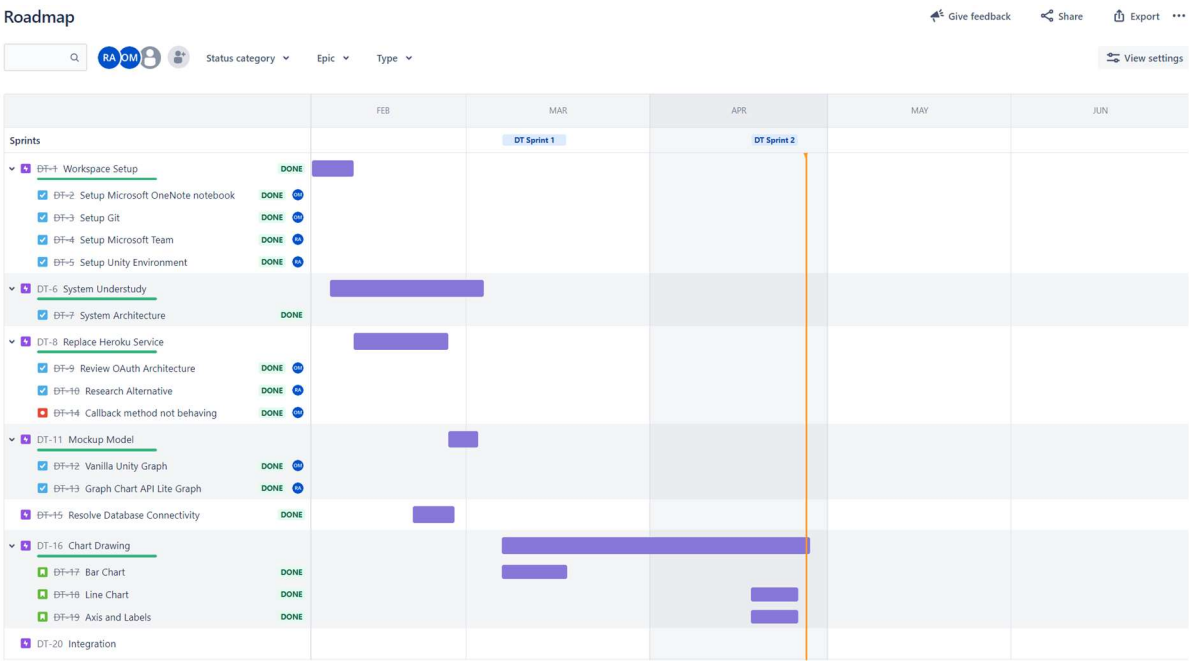
2. Project Management

2.1 Development Methodology

At the start of the project, we developed a comprehensive plan to ensure timely completion of all project tasks. The plan detailed the key milestones and activities necessary to meet project objectives. We monitored and updated the plan regularly to incorporate any changes in scope or schedule.

2.2 Planning and Strategy

To effectively manage the project, we utilized Jira software to plan and track our sprints. This enabled us to create and assign tasks, track progress, and maintain effective communication. The project was broken down into multiple sprints. At the end of each sprint, we conducted a retrospective meeting to evaluate progress and identify areas for improvement. Our use of agile methodology allowed us to quickly adapt to changes and ensure that the project remained on course.



2.1 Communication Strategy

Communication is a critical component of project success, and our team understood this from the start. We agreed on a communication plan that included scheduled team meetings online and in person.

We utilized various communication tools such as Microsoft Teams, email and One Note to stay connected with each other. We followed an Agile development approach that involved frequent iterations and reviews of our progress, and Jira Software helped us manage our tasks, sprints, and deadlines.

To maintain effective communication with our project supervisor and product owner, we scheduled regular meetings with them to discuss progress, receive feedback, and address any issues or concerns. We provided them with regular updates on our progress and informed them of any changes to the project plan.

Overall, our communication plan facilitated teamwork, kept us on schedule, and ensured that all parts shared the same vision and goals for the project.

2.3 Testing Strategy

3. System Design

3.1 System Configuration

3.2 Backend Adaptation

Website??

3.3 User Interface Design

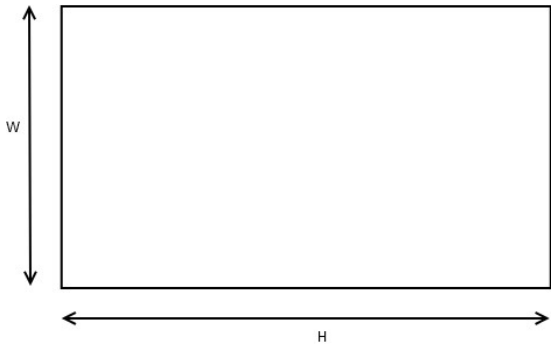
Chart Design

With eight different vales to plot in the chart, a design was proposed that will incorporate bar and line charts. The bar chart is effectively a histogram which comprises of three different variables to be plotted on the same timeline.

The decision to plot the three primary test values with a histogram was a compromise on space and a reaffirmation of our principle of grouping similar values together. This was justified by the fact that three values to be plotted as a single histogram are time-based values namely, Fine Motor – Average Tracking Time, Audio – Average Response Time, Visual – Average Tracking Time.

All other values were plotted as a line chart.

The drawing canvas, which is the area of the screen where the graph will be plotted has dimensions $width = W, height = H$



The y-axis of the graph represents the value to be plotted while the x-axis axis represents time.

1. Bar Charts:

These will represent the following values:

i. Average Tracking Time – Fine Motor.
ii. Average Response Time – Audio.
iii. Average Tracking Time – Video.

2. Line Charts:

List of Displayable:

- Fine Motor
i. Accuracy
ii. Average Distance
iii. Average Tracking Time

- Audio
i. Average Response Time
ii. Minimum Sound Threshold

- Visual
i. Shot Accuracy
ii. Target Accuracy
iii. Average Tracking Time

Chart requirements and constraints.

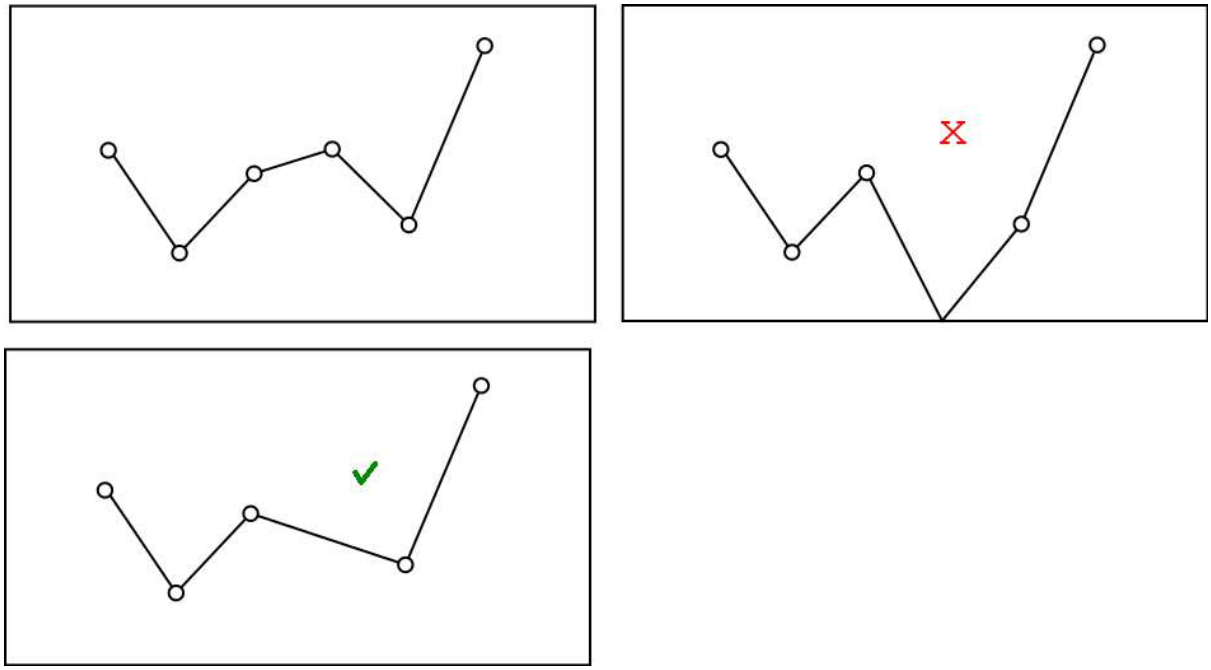
i. The Bar charts will be plotted on the same scale.
ii. The maximum value plottable on the Bar chart will be the maximum value obtainable from the various variables that will be plotted as Bar chart.
iii. The maximum value plottable by any of the Chart types (Bar & Line) will be at 95% of the height of the drawing canvas.
iv. The minimum variable plottable by any of the chart variables may anchor at the beginning (bottom) of the drawing canvas if appropriate otherwise will default to a reasonably lower value. For example, plotting values ranging from 2.1 to 55.8 should result in the API choosing a minimum value of 0 and a maximum value of 60. Also considering another scenario where a dataset with 100 values has a minimum value of 2,500 and a maximum value of 2,510. A naive plotting algorithm taking a minimum value of 0 and a maximum value of 2,510 will produce a very undesirable chart where all the data are plotted at the very top of the chart leaving most of the drawing canvas barren.
This implies that to maximize the canvas usage and thus improve clarity in the chart the minimum value must be chosen in such a manner that it is a function of the maximum value. Also considering the clarity that a datum value of 0(zero) brings, a better algorithm should whenever possible, default to a minimum value of 0 (zero) when within a specified threshold. This threshold could be identified by placing constraints that forbid the plotting of a set of values in such a way that more than 70% of the drawing canvas along the vertical is never visited.
v. By default, the API is designed to plot values over a 52-week period and can be adjusted to plot values over a user defined range of time. The number of values

to be plotted on the canvas is not necessarily a function of the time frame to be plotted because a user can play an unlimited number of tests per day and thus have hundreds of data within a span of a single day. Instead, the number of values to be plotted over a period directly relates to the number of tests the user had within the said timeframe.

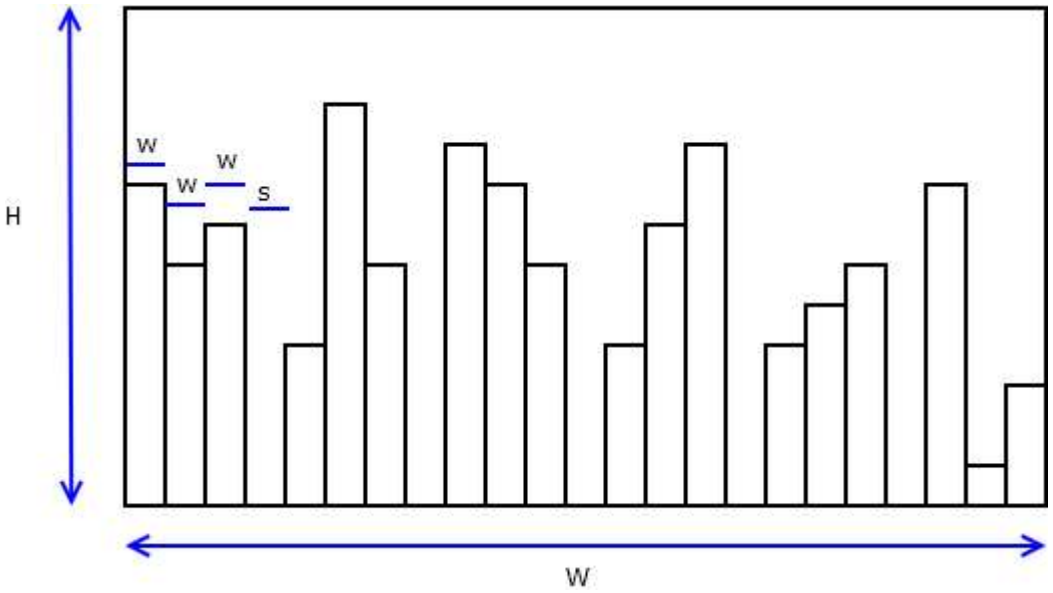
vi. The maximum number of values plottable on the x-axis, will be such that each bar chart will occupy a minimum of 1 pixel of display space, and the spacing between each batch of bar charts guaranteed a minimum of 1 pixel of display space.

Let W = width of the canvas, w = width of each bar chart, and s = space between adjacent batches of bar charts. Let n_{max}= maximum number of entries plottable on the chart.

vii. For the line graph which is basically a line connecting consecutive values of a scatter plot, there is a possibility that some entries in the dataset to be plotted(5 different values) are non-existent for a subset of the dataset. When this is the case such non-existent data should not default to a datum value instead, should be skipped and line plotting suspended till the next available value is fetched to connect the dot. This might seem trivial but is a valid constraint considering the way Unity LineRenderer which is the unity native API for rendering lines, works.



The diagrams depicted above explains the constraint discussed prior. The first chart shows a line graph with 6 points, the second chart shows an undesirable behavior where the 4th point is missing and defaults to the datum point, while the 3rd graph exhibits the required behavior of ignoring the non-existent point and proceeding to connect the next available point.



This implies that for the above-mentioned condition to be true then,

$$W = n_{max}(3 \times w) + (n_{max} - 1)s$$

Resolving for n_{max}, we say

$$W = 3wn_{max} + sn_{max} - s$$

$$W = n_{max}(3w + s) - s$$

$$W + s = n_{max}(3w + s)$$

$$n_{max} = \frac{W + s}{3w + s}$$

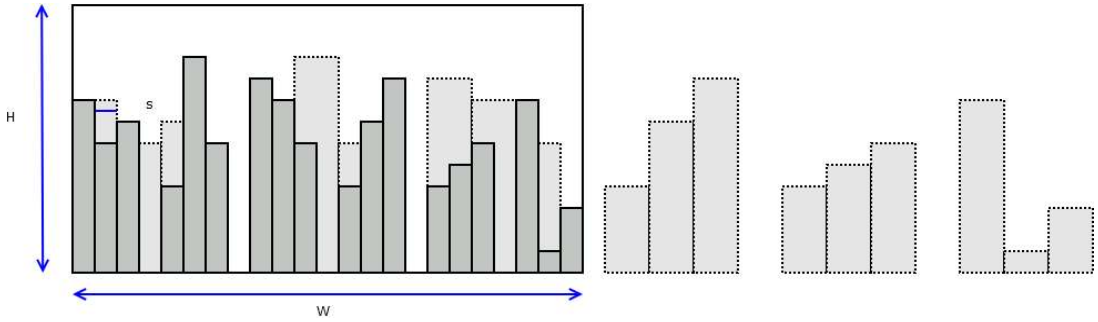
Then the maximum number of bar charts plottable $n_{max} = \frac{W+s}{(3\times w+s)}$

viii. When n number of values is to be plotted and $n > n_{max}$, then the values will be grouped in such a way that the time of the first entry t_0 appears at the very start of the x-axis and the time of the very last entry t_n appears at the very end of the x-axis. Then each plottable entry for the chart will be the aggregated average for the timespan represented by $t_{j,0} = \frac{t_n - t_0}{n_{max}}, t_{j,1} = t_{j,0} + \frac{t_n - t_0}{n_{max}}, t_{j,2} = t_{j,1} + \frac{t_n - t_0}{n_{max}} + \cdots,$

$$t_{j,i} = t_{j,i-1} + \frac{t_n - t_0}{n_{max}}$$

ix. When certain number of values n, is plotted on the chart, the user can choose to order a subset of the data. This will be implemented in such a way that the user will get the impression of a zooming operation, making the bar charts bigger along the x-axis. This is because of the ordered subset occupying the total width of the drawing canvas.

For Example, if n = 100 and the user orders a subset of n with 50 values, then given the initial width of bar chart w_{100}, the final width of the bar chart after ordering the subset will be $w_{50} = \frac{100}{50} \times w_{100}$. However, the width of the bar chart will be restricted in such a way that the bar charts are not unreasonably large when there are too little values to plot. A typical scenario would be attempting to plot a chart with say three values to occupy the whole canvas. In such scenarios the width of the bar chart will be limited to a certain value deemed appropriate. This value was evaluated and tested using heuristic methods to determine a suitable value.



From the diagram above the foreground group of charts represents the whole dataset plotted in such a way that it occupies the width of the canvas and the lighter background chart shows a subset of the original dataset (ordered from the very start to a point at the very center of the dataset) plotted in the same manner to occupy the whole width of the canvas. The width of the lighter chart is expected to be roughly twice that of the other one.

4. Conclusion & Final Thoughts

Otito

This piece of work shows the capability of the Unity API to perform such tasks that are not ideally meant for it. And can potentially developed to be a reusable Charting API capable of plotting various kinds of charts. While working on the project, I learned a lot more about the Unity API especially working with various User Interface (UI) Components.

Rodrigo

At the start of the project was a bit worried as I had another project in mind to work on but after our first meeting with the product owner and our supervisor the brief became clear and then I felt more comfortable with it. As time went by, we encountered a few problems that I found challenging but it gave me experience while troubleshooting and I also got

exposed to technologies like Firebase that I didn’t work with before and even more with Unity and C#. Meeting weekly with our supervisor and the project owner gave us a better insight into the way to go and what to avoid on the way. A high point of this project was working in a group, even with only two members, I found it a pleasant experience as Otito was always very supportive, knowledgeable and very committed to the project.

5. Future Development

In future work, there are several aspects that could be improved upon to elevate the system's overall functionality and performance. These areas include:

Integration: Due to time constraints, our team was unable to complete the integration phase of the project. In subsequent efforts, our team could prioritize integrating the API with the existing system to deliver a seamless user experience.

User Interface: Although the team made substantial progress in refining the user interface, there is still room for improvement. In future projects, the team could focus on enhancing the user interface to make it more user-friendly and intuitive.

Data Analysis: The team could explore advanced statistical methods to improve the system's data analysis capabilities. By providing users with more comprehensive insights into their performance, the system's value would be enhanced.

Overall, there is immense potential for future projects to boost the system's performance and functionality. By addressing these areas, the team could deliver a more effective and powerful tool for assessing user performance on a variety of tests.