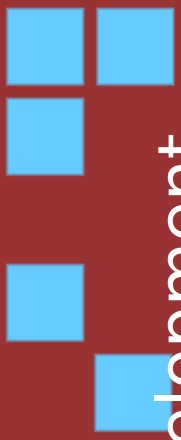

B.Sc. (Hons) in Software Development

Ollscoil Teicneolaíochta an Atlantaigh

Atlantic Technological University

Biometric Data Analysis in Digital Game Scenario

By
Rodrigo Almeida

November 21, 2023

Minor Dissertation

Department of Computer Science & Applied Physics,
School of Science & Computing,
Atlantic Technological University (ATU), Galway.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

This chapter should provide a clear context for your project and set out its objectives. You can cite references from the bibliography using IEEE format, such as Claude Shannon [**?**] and John Von Neumann *et al* [**?**]. Use Google Scholar's BibTex export function to get a LaTeXformatted citation to copy and paste into the *references.bib* document.

Chapter 2

Methodology

Describe the way you went about your project. Was your approach to the problem valid? You need to discuss both your software development methodology and your research methodology.

Chapter 3

Technology Review

3.1 Executive Dashboard

When it came to developing the Executive Dashboard for data analysis it was crucial to select the tools that could transform data into easily comprehensible information. After conducting research and exploring alternatives Angular and Chart.js were chosen for specific reasons.

3.1.1 Angular

Angular is based on TypeScript and is an open-source framework, and it's a great tool for building client-side web applications. It can be used for single-page applications (SPAs) or for enterprise-level solutions.

Main Concepts: Components, Modules and Services

Angular is composed of three foundational blocks: Components, Modules and services.

Components

Components are like Lego blocks of the application. Each component holds a portion of the user interface and its behaviour. Components serve as the bridge between the application data and what the user experiences on the screen.[1]

Modules

In Angular modules serve as containers that group related components, directives, and services together that can be combined with other modules. It plays an im-

portant role in improving maintainability and re-usability, key concepts of Angular development.[2]

Services

Angular services use typescript classes with injectable decorators. The decorator tells angular that the class is a service and can be injected into other components that need that service.[3]

3.1.2 Angular architecture

Angular uses the Model-View-Controller (MVC) pattern, with a variation known as Model-View-ViewModel (MVVM). The controller is responsible for the interaction between the model and the view.

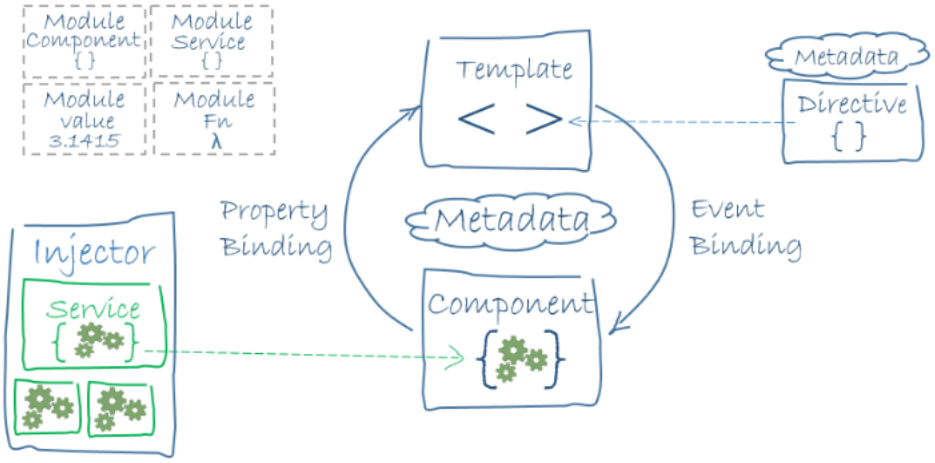

Figure 3.1: Angular application architecture

3.1.3 Advantages of using Angular

Component-Based Architecture

As mentioned earlier in discussions Angular organisess its functionalities into components. These components have the ability to communicate with each other enabling updates to sections without affecting the rest of the application.

Mobile-Friendly Approach

Angular incorporates techniques such, as lazy-loading, which means loading parts of the application (like images) only when they are needed. This ensures that users do not experience long waiting times.

Two-Way Data Binding

With Angular two way data binding data can seamlessly flow between the component and the view allowing for synchronization.

Asynchronous Programming

By utilizing programming executes code in a non-sequential manner and employs multi-threading to enhance performance. This speeds up operations and prevents system freezes, providing users with a seamless experience.

Single-Page Applications

Angular creates a dynamic single-page application which can be navigated without page reloads, improving the user experience with better user interaction and engagement.

Code Re-usability

The component-based architecture of Angular promotes the re-usability of UI components saving development time.

Dependency Injection

With dependency injection in place, Angular allows for the creation of objects that rely on other objects. This improves modularity and efficiency, within the app.

Angular Material

Angular's documentation offers a range of built user interface components and modules that adhere to Google's Material Design principles. This greatly facilitates the developer's work, simplifying the design process and enabling application development.

Angular CLI

Angular command line interface gives the developer the ability to generate Angular projects, modules, services, and components with a single command, this not only saves time but also reduces configuration errors, it gives the developer the freedom to dive into creative aspects of the project, focusing on innovation and functionality rather than getting bogged down by initial setup complexities.

Angular command line interface gives the developer the ability to generate Angular projects, modules, services, and components with a single command, this not only saves time but also reduces configuration errors, it gives the developer the freedom to dive into creative aspects of the project, focusing on innovation and functionality rather than getting bogged down by initial setup complexities.

3.2 Data Visualization

Data visualization is the process of translating large volumes of data into visual representations. It is a powerful tool that allows for the identification of patterns and trends that would otherwise be difficult to identify. It is a key component of data analysis and is used to communicate information clearly and efficiently.

3.2.1 Chart.js

Chart.js is robust, lightweight and open source javascript library that provides features for creating visually appealing charts and graphs that can be embedded into a web page. It is a great tool for data visualization and is easy to use. It is based on HTML5 canvas and is responsive, meaning that the charts will adapt to the size of the container element. It is also compatible with all modern browsers and is supported by a large community of developers. [4] It can be compared like a the artist's toolkit for developers, providing a flexible API to create a variety of chart types, such as line charts, radar charts, pie charts and many more. It also provides a range of customization options, allowing developers to create unique and visually appealing charts. [4]

Table 3.2.1 is showing some of the features of Chart.js[4]:

Feature	Description
Easy to use	Aesthetically charts can be created without the need of extensive configuration. The library follows a declarative approach allowing the developer to define the data and settings of the chart in a single object.
Responsive	Charts generated by Chart.js are responsive, adapting to different screen sizes and devices, ensuring that the visualization remains accessible and readable across various platforms.
Customization	Chart.js provides a high degree of customization. Colors, fonts, and other visual elements can be customized to create unique and visually appealing charts.
Interactivity	Chart.js provides built-in support for tooltips and animation, allowing the user to explore data points, adding a layer of engagement to the project.
Cross-browser compatibility	Chart.js is compatible with all modern browsers, including Chrome, Firefox, Safari, Edge, and Internet Explorer 11.

Table 3.1: Chart.js Features

Chart.js is a reliable and effective tool for data visualization, it is also easy to use and provides a range of customization options. It is also supported by a large community of developers, which is a great advantage.

3.2.2 D3.js

D3.js or Data-Driven Documents is a JavaScript library for manipulating documents based on data. It has the ability ti bind data to the DOM (Document Obect Model) and creates interactive and dynamic visualization. It provides a lower level and more granular approach to data visualization, giving the developer more control over the visualization process.[5]

Table 3.2.2 is showing some of the features of D3.js[5]:

Feature	Description
Data-Driven	D3.js is data-driven, meaning that it can bind data directly to HTML or SVG elements. This type of control is advantageous for creating highly customized visualizations.
Modular	D3.js is modular, composed of many small modules that can be used independently or combined together to create a custom visualization.
Extensible	D3.js is extensible, meaning that it can be extended to support any type of visualization.
Community	D3.js is supported by a large community of developers, which is a great advantage.

Table 3.2: D3.js Features

D3.js is a powerful tool for data visualization, it provides a high degree of customization and control over the visualization process. However, it is more complex and requires more time to learn and implement. It is also not supported by all browsers, which is a disadvantage.

In conclusion, while D3.js is a powerful tool for data visualization, it is more complex and requires more time to learn and implement. It is also not supported by all browsers, which is a disadvantage. Considering the scope of the project and the time constraints, Chart.js was chosen as the tool for data visualization. It is easy to use and provides a high degree of customization, allowing for the creation of unique and visually appealing charts. It is also supported by a large community of developers, which is a great advantage.

3.3 AI module - Data analysis

Ninety per cent of the world's data was generated in just the last two years. In the early 2000s, the amount of data being generated exploded exponentially with the use of the internet, social media, and various other technologies. Organizations found themselves facing a massive volume of data that was very hard to process. To address the challenge, the concept of Big data emerged. Big data refers to extremely large and complex data sets that are difficult to process using traditional methods. [6]

3.3.1 Hadoop

As the volume of data grew rapidly across the globe, organizations needed a way to manage it to gain valuable insights. That's where Hadoop came in. In 2006,

a team of engineers from Yahoo developed Hadoop inspired by Google's MapReduce. Hadoop has introduced a new way of handling data called distributed processing. Now, to process all data it wouldn't use single machines anymore, instead, Hadoop uses multiple computers, allowing large amounts of data to be processed way faster than before. Hadoop has mainly two main components: HDFS (Hadoop Distributed File System) and MapReduce. HDFS stores the into multiple computers, and MapReduce is responsible for processing the data in parallel allowing the organizations to store and process large amounts of data. Hadoop was a great advance in big data processing, however, it had a few limitations. One of the biggest problems was that it relied on storing data on disk. This slowed down data processing because every time a job ran it had to save its data to disk, read it back, process it, and save it back to disk Another problem is that Hadoop processes data only in batches. This means a new job couldn't be submitted before the other job ended. With Hadoop, storing and processing large amounts of data became possible, even with some limitations it was an important step in big data processing.[7]

3.3.2 Apache Spark

As described Hadoop has a few limitations, there was a need to process all this data faster and in real-time. And here is where Apache Spark comes into action. In 2009, researchers at the University of California developed Apache Spark as a research project. At this point, RDD (Resilient Distributed Dataset) was introduced and deemed to be a very powerful concept.[8] RDD is the backbone of Apache Spark, storing the data in memory for faster access and processing. It will not read and write the data from the disk, instead, Spark processes the entire data just in memory. The meaning of memory here is the RAM (Random Access Memory) stored inside the computer. This in-memory data processing of data makes Spark much faster than Hadoop. Spark also gives the ability to write code in various programming languages such as JAVA, Scala, Python and R, along with an optimized engine that supports general execution graphs.

Main components and Features

The Apache Spark ecosystem consists of the following main components[9]:

- Spark SQL: Formerly known as Shark. Spark SQL is a distributed framework that works with structured and semi-structured data. It facilitates analytical and interactive applications for both streaming and historical data which can be accessed from various sources such as JSON, Parquet and Hive table.[9]

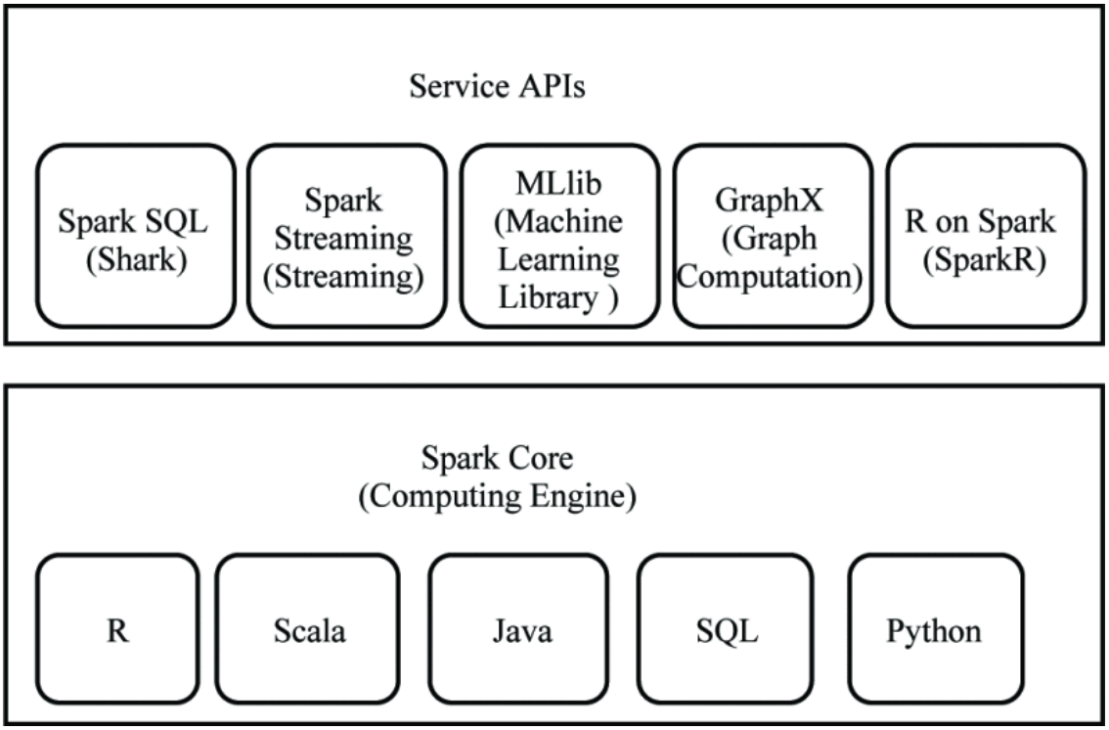

Figure 3.2: Apache Spark Ecosystem

- Spark Streaming: Allows to process real-time data. Is a scalable fault-tolerant streaming processing system that supports both batch and streaming workloads. Spark streaming enhances the fast scheduling capability of Apache Spark by inserting data into mini-batches. An operation known as transformation is then applied to those mini-batches that can be easily obtained from live streams and data sources such as Twitter, Apache Kafka, IoT sensors, and Amazon Kinesis.[9]

- Spark Core: One of the most important parts of the Spark ecosystem is called Spark Core. It helps process data across multiple computers and ensures everything works efficiently and smoothly. Various functionalities of Apache Spark are built on top of the Spark core. It provides a vast range of APIs as well as applications for programming languages such as Scala, Java, and Python APIs to facilitate the ease of development. In-memory computation is implemented in Spark core in order to deliver speed and solve the issue of MapReduce.

- MLlib: It delivers high-quality algorithms with high speed and makes ma-

chine learning easy to use and scale. Several machine learning algorithms such as regression, classification, clustering, and linear algebra are present. It also provides a library for lower-level machine learning primitives like the generic gradient descent optimization algorithm. It also provides other functions such as model evaluation and data import. It can be used in Java, Scala, and Python. [9]

With all these components working together, Apache Spark became a powerful tool for processing and analysing Big Data. Spark manages and coordinates the execution of tasks on data across a cluster of computers.

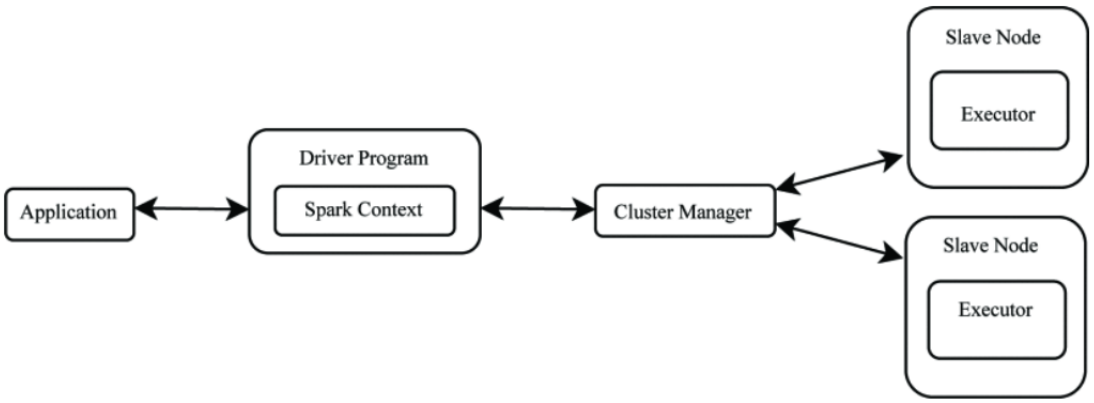

Figure 3.3: Apache Spark Architecture

Figure 3.3 shows the Spark Architecture. It consists of a master node which has a driver program that is responsible for calling the main program of an application, this driver could be the Spark shell or the application, written in Scala or Java, for example. It is basically the entry point and is responsible for creating Spark context, which behaves like a gateway to all of the functionalities of Apache Spark. It allows for communication and coordination with other nodes within the cluster, known as Slave nodes. Within the slave node, there are many numbers of executors which act as workers. Whenever a job is initiated, the driver process will make sure it goes through the Apache application properly. It analyzes the work that needs to be done, divides it into smaller tasks and assigns it to the executors. The driver process is the heart of the Apache Spark application and executors are the real workers, which execute the work assigned by the driver and report back the progress and results of the computation.[9]

Due to the powerful capabilities of distributed computing, large volumes of data can be processed quickly and efficiently. For all these reasons, it was deemed

to be the perfect technology to be used in this project. As the project evolves and more data is gathered from our research volunteers, Spark's performance scales with the size of the data, processing with high speed.

Chapter 4

System Design

Provide a detailed explanation of the overall system architecture [**?**], i.e. the HOW of the project. Use UML, system architecture diagrams, screenshots, code snippets and algorithms to illustrate your design.

4.1 Working with Images

You can embed an image in a LaTeXdocument using the technique shown below. System diagrams and images with a small numbers of colours (100s, not 1000s) should be stored in PNG format. Although LaTeXdoesn't care where you place your images, it is good practice to place them in a single sensible directory and apply some sort of hierarchy to them, e.g. the path images/chapter1 might contain all of the images for Chapter 1 of your dissertation.

Chapter 5

System Evaluation

Evaluate your project against the objectives set out in the introduction. This chapter should present results if applicable and discuss the strengths and weaknesses of your system. This is a clear opportunity for you to demonstrate your critical thinking in relation to the project.

5.1 Working with Tables

Table 5.1 can be referenced with the label given to the table, i.e. **ref{table:HexToBin}**. Note that LaTeXwill place the table wherever it deems fit. Don't bother trying to change where a table or figure is placed until your document is ready for final layout.

Hexadecimal to Binary					
Hex	Binary 2	Hex	Binary	Hex	Binary
1	00000001	B	00001011	15	00010101
2	00000010	C	00001100	16	00010110
3	00000011	D	00001101	17	00010111
4	00000100	E	00001110	18	00011000
5	00000101	F	00001111	19	00011001
6	00000110	10	00010000	1A	00011010
7	00000111	11	00010001	1B	00011011
8	00001000	12	00010010	1C	00011100
9	00001001	13	00010011	1D	00011101
A	00001010	14	00010100	1E	00011110

Table 5.1: Conversion from Hexadecimal to Binary

Chapter 6

Conclusion

Briefly summarise your context and objectives. Remind the reader about the overall rationale and goals of the project. Highlight your findings from the System Evaluation chapter.

Bibliography

[1] Angular Components.

[2] Angular Modules.

[3] Angular Services.

[4] Helder Da Rocha. *Learn Chart. js: Create interactive visualizations for the web with chart. js 2*. Packt Publishing Ltd, 2019.

[5] Nick Qi Zhu. *Data visualization with D3. js cookbook*. Packt Publishing Ltd, 2013.

[6] News. *Significance*, 9(4):2–3, 08 2012.

[7] Ivanilton Polato, Reginaldo Ré, Alfredo Goldman, and Fabio Kon. A comprehensive view of hadoop research—a systematic literature review. *Journal of Network and Computer Applications*, 46:1–25, 2014.

[8] Salman Salloum, Ruslan Dautov, Xiaojun Chen, Patrick Xiaogang Peng, and Joshua Zhexue Huang. Big data analytics on apache spark. *International Journal of Data Science and Analytics*, 1:145–164, 2016.

[9] Eman Shaikh, Iman Mohiuddin, Yasmeen Alufaisan, and Irum Nahvi. Apache spark: A big data processing engine. In *2019 2nd IEEE Middle East and North Africa COMMunications Conference (MENACOMM)*, pages 1–6, 2019.