الجمهوريـة الجزائـريـة الديمقراطيـة الشعبيـة
**République Algérienne Démocratique et Populaire**
وزارة التعليـم العالـي والبحـث العلمـي
**Ministère de l'Enseignement Supérieur et de la Recherche Scientifique**


**Nº Réf :……**

**Centre Universitaire**
**Abd Elhafid Boussouf Mila**

**Institut des Sciences et Technologie** **Département de Mathématiques et Informatique**

**Mémoire préparé en vue de l'obtention du diplôme de Master**

**En : Informatique**

**Spécialité: Sciences et Technologies de l'Information et de la Communication (STIC)**

# Di Statistics : Data Statistics and Anatytics system to DiData (Laboratory Managment System)

**Préparé par** :
**Aissam YEKHLEF**

**Soutenue devant le jury**

| | | | |
|---|---|---|---|
| **Mr. Adil MERABET** | **MAA** | **C.U.Abd Elhafid Boussouf** | **Président** |
| **Dr. Nardjes BOUCHEMAL.** | **MCA** | **C.U.Abd Elhafid Boussouf** | **Examinateur** |
| **Mr. Samir SELMANE.** | **MAA** | **C.U.Abd Elhafid Boussouf** | **Rapporteur** |

**Année Universitaire : 2020/2021**

# Acknowledgments

First we want to thank Almighty God, For giving us the strength and patience to accomplish this humble work.

Thanks to my supervisor Mr. Selmane Samir,

We would like to extend our sincere thanks to the committee members,Mme. Bouchemal Nardjes and Mr. merabet Adil for accepting to judge this work. Thanks to my family for the support and for offering a suitable atmosphere to complete my dissertation

# Dedication

I dedicate this modest work to the most cherished beings in my life, my family to whom I must thank them very much for their moral and material support, understanding, tenderness, love and sacrifice.

To the **DiData team** for helping me during the internship.

**Aissam**

# Abstract

DiData is a web-based platform that integrates scientific data such as clinical projects, laboratories, biobanks, and more. DiData supports data privacy regulations, compliance.

DiData has a lot of scientific data and researchers of DiData need to show all the Statistics about their data and insights. These data not readable and complex to manage from an non IT manager, So they need a third-party solution to Extract and simplify this data.

The goal of this project is the Analyze of scientific data from the system to know the insights, and have a good vision of how the data develop by criteria, and compare results of selected periods of time insights.



**Keywords**: DiData, Di-statistics, Web Application, Statistics, Analytics.



# Résumé

DiData est une plate-forme Web qui intègre des données scientifiques telles que des projets cliniques, des laboratoires, des biobanques, etc. DiData prend en charge les réglementations en matière de confidentialité des données, la conformité, ...etc.

DiData a beaucoup de données scientifiques et les chercheurs de DiData doivent montrer toutes les statistiques sur leurs données et leurs idées. ces données non lisibles et complexes à gérer depuis un non responsable informatique, Ils ont donc besoin d'une solution tierce pour extraire et simplifier ces données.

L'objectif de ce projet est l'analyse des données scientifiques du système pour connaître les idées et avoir une bonne vision de la façon dont les données se développent par critères et comparer les résultats de périodes sélectionnées



**Mots clés**: DiData, Di-statistics, Application Web, Statistiques, Analytics,.

# الملخص

ديداتا هي عبارة عن منصة قائمة على الويب تدمج البيانات العلمية مثل المشاريع السريرية والمختبرات والبنوك الحيوية والمزيد. تدعم ديداتا لوائح خصوصية البيانات والامتثال.

تحتوي ديداتا على الكثير من البيانات العلمية ويحتاج باحثو ديداتا إلى إظهار جميع الإحصائيات حول بياناتهم ورؤاهم. هذه البيانات غير قابلة للقراءة ومعقدة لإدارتها من غير مدير تكنولوجيا المعلومات ،لذا فهم بحاجة إلى حل من جهة خارجية لاستخراج هذه البيانات وتبسيطها.

الهدف من هذا المشروع هو تحليل البيانات العلمية من النظام لمعرفة الرؤى ، والحصول على رؤية جيدة لكيفية تطور البيانات حسب المعايير ، ومقارنة نتائج فترات محددة من الرؤى الزمنية

# Contents

# List of Figures

# List of Tables

# List of abbreviations

LIMS: laboratory information management system
API : Application programming interface
AJAX: Asynchronous JavaScript And XML SPA : single page application
MPA : multi page application
MVC : Model, View, Controller
MVVM: Mode-View-Viewmodel
DBMS : database management system
RDBMS : Relational database management system
PHP : preprosessor hypertext
SQL : structural query language
WAMP : Windows, Apache, MySQL, PHP
NPM : node package manager
WWW : world wide web
IoT : internet of things
JS : JavaScript
DB : database
MySQL : My Stractural query language
JSON : JavaScript Object Notation
JWT : Json Web Token
REST : REpresentation State Transfer
SOAP : Single Object Access Protocol
PWA : Progressive Web Application
VSCode : visual studio code
CLI : Command Line Interface
SaaS : Software as a service

# General Introduction

DiData is a ready-to-use and flexible web-based platform to integrate Patient registries and Biological samples data. DiData takes care of data privacy regulations, compliance requirements, and removes all headaches allowing you to entirely focus on your work. [1]

DiData provide solutions regardless of the user's IT background, that means the Data is:

- huge data ( millions of entities ).
- complex to read.
- sensitive data (clinic laboratories data).
- has a custom data structure .

## DiData Vision

DiData's objective is to provide high-tech healthcare data management solutions regardless of the user's IT background. We believe that a tailored tool in the right hands is key for research breakthroughs and would open the door to new opportunities. From the early stages of DiData, we collaborated with medical doctors and scientists to ensure that our environment provides both a great user experience and cutting-edge IT solutions to tackle their challenges.

## DiData needs

DiData need a third party as a service (SaaS) to help it with its data Statistics and analytics,

There are many of them like excel, tableau, or python.

The problems of these third parties are :

- problem of security and can't share the database credentials with others to make DiData a secure software.
- complexity of data, data is just rows and columns on tables which have relationship between them in a specific structure.
- the need for knowing how this data is structured.

# Our Solution

We decide to build a web application to DiData called **Di-Statistics**, it complete the didata system, it help DiData better understand their insights.

Our thesis consists of three chapters organized as follows:

- **Data Statistics and analytics**: we start this chapter a global view on Data, data Science, Data Analytics, and the relation between them.
- **Conception and Modelisation** is an analysis phase of our system that breaks it down into categories until it reaches the dynamic model.
- **Implementation**
  - Part 1 : Tools and Technologies this is the first part, we talk about the tools that we used in the implementation
  - Part 2 : Work Steps this is the second part, we talk about the steps we are taking to implement the Di-Statistics project from the installation to the test.

# Chapter 1

# Data Statistics and analytics.

## 1.1 Introduction

In this chapter, we will give a brief overview of Data and Data Science in general and especially Data Analytics by choosing the right tool as a case study. After that, we will introduce the key basic concepts.

## 1.2 Maching Learning

**What is maching learning ?** Machine learning is a method of data analysis that automates analytical model building. It is a branch of artificial intelligence based on the idea that systems can learn from data, identify patterns and make decisions with minimal human intervention. [7]

## 1.3 Data Science

Data science combines multiple fields, including statistics, scientific methods, artificial intelligence (AI), and data analysis, to extract value from data. Those who practice data science are called data scientists, and they combine a range of skills to analyze data collected from the web, smartphones, customers, sensors, and other sources to derive actionable insights.

Data science encompasses preparing data for analysis, including cleansing, aggregating, and manipulating the data to perform advanced data analysis. Analytic applications and data scientists can then review the results to uncover patterns and enable business leaders to draw informed insights. [8]

## 1.4 Data Anatytics

**What Is Data Analytics?**

### 1.4.1 Definition

Data analytics is the science of analyzing raw data to make conclusions about that information. Many of the techniques and processes of data analytics have been automated into mechanical processes and algorithms that work over raw data for human consumption.

Data analytics help a business optimize its performance. [**?**]

### 1.4.2 Understanding Data Analytics

**Data analytics** is a broad term that encompasses many diverse types of data analysis. Any type of information can be subjected to data analytics techniques to get insight that can be used to improve things.

**Data analytics** techniques can reveal trends and metrics that would otherwise be lost in the mass of information. This information can then be used to optimize processes to increase the overall efficiency of a business or system.
For example, manufacturing companies often record the runtime, downtime, and work queue for various machines and then analyze the data to better plan the workloads so the machines operate closer to peak capacity.
**Data analytics** can do much more than point out bottlenecks in production. Gaming companies use data analytics to set reward schedules for players that keep the majority of players active in the game. Content companies use many of the same data analytics to keep you clicking, watching, or re-organizing content to get another view or another click.
**Data analytics** is important because it helps businesses optimize their performances. Implementing it into the business model means companies can help reduce costs by identifying more efficient ways of doing business and by storing large amounts of data. A company can also use data analytics to make better business decisions and help analyze customer trends and satisfaction, which can lead to new—and better—products and services. [**?**]

### 1.4.3 Data Analysis Steps

The process involved in data analysis involves several different steps:
**The first step** is to determine the data requirements or how the data is grouped. Data may be separated by age, demographic, income, or gender. Data values may be numerical or be divided by category.
**The second step** in data analytics is the process of collecting it. This can be done through a variety of sources such as computers, online sources, cameras, environmental sources, or through personnel.
Once the data is collected, it must be organized so it can be analyzed. This may take place on a spreadsheet or other form of software that can take statistical data.
The data is then cleaned up before analysis. This means it is scrubbed and checked to ensure there is no duplication or error, and that it is not incomplete. This step helps correct any errors before it goes on to a data analyst to be analyzed. [**?**]

### 1.4.4 Data Analytics Types

**Data analytics** is a broad field. There are four primary types of data analytics: descriptive, diagnostic, predictive, and prescriptive analytics. Each type has a different goal and a different place in the data analysis process. These are also the primary data analytics applications in business.

- **Descriptive analytics** helps answer questions about what happened. These techniques summarize large datasets to describe outcomes to stakeholders. By developing key performance indicators (KPIs,) these strategies can help track successes or failures. Metrics such as return on investment (ROI) are used in many industries. Specialized metrics are developed to track performance in specific industries. This process requires the collection of relevant data,

processing of the data, data analysis, and data visualization. This process provides essential insight into past performance.

- **Diagnostic analytics** helps answer questions about why things happened. These techniques supplement more basic descriptive analytics. They take the findings from descriptive analytics and dig deeper to find the cause. The performance indicators are further investigated to discover why they got better or worse. This generally occurs in three steps: Identify anomalies in the data. These may be unexpected changes in a metric or a particular market. Data that is related to these anomalies is collected. Statistical techniques are used to find relationships and trends that explain these anomalies.

- **Predictive analytics** helps answer questions about what will happen in the future. These techniques use historical data to identify trends and determine if they are likely to recur. Predictive analytical tools provide valuable insight into what may happen in the future and its techniques include a variety of statistical and machine learning techniques, such as neural networks, decision trees, and regression.

- Prescriptive analytics helps answer questions about what should be done. By using insights from.

- **predictive analytics**, data-driven decisions can be made. This allows businesses to make informed decisions in the face of uncertainty. Prescriptive analytics techniques rely on machine learning strategies that can find patterns in large datasets. By analyzing past decisions and events, the likelihood of different outcomes can be estimated. These types of data analytics provide the insight that businesses need to make effective and efficient decisions. Used in combination they provide a well-rounded understanding of a company's needs and opportunities. [3]

### 1.4.5 Why Is Data Analytics Important?

The applications of data analytics are broad. Analyzing big data can optimize efficiency in many different industries. Improving performance enables businesses to succeed in an increasingly competitive world. One of the earliest adopters in the financial sector. Data analytics has an important role in the banking and finance industries, used to predict market trends and assess risk. Credit scores are an example of data analytics that affects everyone. These scores use many data points to determine lending risk. Data analytics is also used to detect and prevent fraud to improve efficiency and reduce risk for financial institutions. The use of data analytics goes beyond maximizing profits and ROI, however. Data analytics can provide critical information for healthcare (health informatics), crime prevention, and environmental protection. These applications of data analytics use these techniques to improve our world. Though statistics and data analysis have always been used in scientific research, advanced analytic techniques and big data allow for many new insights. These techniques can find trends in complex systems. Researchers are currently using machine learning to protect wildlife.

### 1.4.6 Who Is Using Data Analytics?

Data analytics has been adopted by several sectors, such as the travel and hospitality industry, where turnarounds can be quick. This industry can collect customer data and figure out where the problems, if any, lie and how to fix them. Healthcare is another sector that combines the use of high volumes of structured and unstructured data and data analytics can help in making quick decisions. Similarly, the retail industry uses copious amounts of data to meet the ever-changing demands of shoppers.

### 1.4.7 Data Analytics in Laboratory Management Systems

The use of data analytics in healthcare is already widespread. Predicting patient outcomes, efficiently allocating funding, and improving diagnostic techniques are just a few examples of how data analytics is revolutionizing healthcare. The pharmaceutical industry is also being revolutionized by machine learning. Drug discovery is a complex task with many variables. Machine learning can greatly improve drug discovery. Pharmaceutical companies also use data analytics to understand the market for drugs and predict their sales.
Example: COVID-19 Analytics helping to know more information about the virus.

## 1.5 Data Science Vs Data Analytics:

**Data Scientists** and Data Analysts utilize data in different ways. Data Scientists use a combination of Mathematical, Statistical, and Machine Learning techniques to clean, process, and interpret data to extract insights from it. They design advanced data modeling processes using prototypes, ML algorithms, predictive models, and custom analysis.
While **data analysts** examine data sets to identify trends and draw conclusions, Data Analysts collect large volumes of data, organize it, and analyze it to identify relevant patterns. After the analysis part is done, they strive to present their findings through data visualization methods like charts, graphs, etc.
Thus, Data Analysts transform complex insights into business-savvy language that both technical and non-technical members of an organization can understand.
Let's take a look at the Data Analytics and Data Science Responsibilities :

### 1.5.1 Responsibilities of Data Scientists

To process, clean, and validate the integrity of data.
To perform Exploratory Data Analysis on large datasets.
To perform data mining by creating ETL pipelines.
To perform statistical analysis using ML algorithms like logistic regression, KNN, Random Forest, Decision Trees, etc.
To write code for automation and build resourceful ML libraries.
To glean business insights using ML tools and algorithms.

To identify new trends in data for making business predictions.

### 1.5.2 Responsibilities of Data Analysts

To collect and interpret data.
To identify relevant patterns in a dataset.
To perform data querying using SQL.
To experiment with different analytical tools like predictive analytics, prescriptive analytics, descriptive analytics, and diagnostic analytics.
To use data visualization tools like Tableau, IBM Cognos Analytics, etc., for presenting the extracted information.

### 1.5.3 Data Statistics Types:

Levels of measurement

**Nominal**

Is called also as Categorical or Qualitative, examples: sex, preferred type of chocolate, number of your willya, color nominal values can be stored as word or text, give a number to present data value Like this: male => 1, female=>0 . The summary measures: frequencies, proportions Can be displayed as Pie Charts, Column bar chart -most used-, state column bar chart.

**Ordinal**

Examples of ordinal variables are Rank, Satisfaction, Fanciness Like norminal data, ordinal data can be We can't create a mean or average for ordinal data The summary measures: frequencies, proportions, and sometimes means Can be displayed as a Column bar chart, but not as Pie Chart.

**Interval/Ratio**

Is also known as scale, quantitative, parametric
Interval/ration data can be: discrete with whole numbers, for example: 5 customers, 15 points, 12 entities
Interval/ration data can be : continuous 4.2 miles, 25.6 kg, 6.7 minutes
Interval/ration is very mathematically, so the summary measures Mean, Median, Standard Deviation
Can be displayed as a Column bar chart, hexagram – where data is grouped-,Boxplot and Line Chart.

## 1.6 Data Analytics Tools:

- R and Python
- Microsoft Power BI
- Microsoft Excel
- Tableau
- Apache Spark
- QlikView
- Splunk

Let's dive into some tools

### 1.6.1 R and Python

- Python is a general-purpose, object-oriented programming language that emphasizes code readability through its generous use of white space, Several Python libraries support data science tasks, including the following:
  - Numpy for handling large dimensional arrays
  - Pandas for data manipulation and analysis.
  - Matplotlib for building data visualizations
- R is a free software environment for statistical computing and graphics. It compiles and runs on a wide variety of UNIX platforms, Windows and MacOS.

### 1.6.2 Microsoft Excel

Microsoft Excel is a helpful and powerful program for data analysis and documentation. It is a spreadsheet program, which contains some columns and rows, where each intersection of a column and a row is a "cell." Each cell contains one point of data or one piece of information. By organizing the information in this way, you can make information easier to find, and automatically draw information from changing data.

### 1.6.3 Microsoft Power BI

Connect to and visualize any data using the unified, scalable platform for self-service and enterprise business intelligence (BI) that's easy to use and helps you gain deeper data insight. [9]
with Power BI we can :

- Create amazing data experiences
- Gain insight from your largest BI deployments
- Make decisions with confidence

### 1.6.4 Tableau

Tableau is the world's leading analytics platform, Tableau helps people see and understand data. Our visual analytics platform is transforming the way people use data to solve problems. See why organizations of all sizes trust Tableau to help them be more data-driven.

### 1.6.5 SQL

(Structured Query Language) is a standardized programming language that's used to manage relational databases and perform various operations on the data in them.

What do we need Exactly? We need a tool that :

- can manage the database and make it securely.
- has the same technologies already used by DiData system.
- can use SQL to integrate with Database.

## 1.7 Our Soloution

Create a 3rd-party Di-Statistics (SAAS) that can be integrated with DiData and helping it to record Data Statistics and Analytics.
we can use SQL tool to get the data from DB, and do the treatment.

## 1.8 Conclusion

In this chapter, we presented the definitions and the tools of Data Analytics that we can use to Analyze data and their characteristics and the principles advantages, and disadvantages of each one. These Tools are available for free like R, Python, SQL, and others premium like Power BI, Tableau
we have chosen the SQL tool for our study because it can access the database directly and manage the data as we need.

# Chapter 2

# Conception and modelisation

## 2.1 Introduction

In this chapter, we are going to describe how the di-statistics (our solution) will look likes. First, we need to take a look at one of the most important tools in the conception.

## 2.2 UML

short for Unified Modeling Language, is a standardized modeling language consisting of an integrated set of diagrams, developed to help system and software developers for specifying, The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems, The UML is a very important part of developing object-oriented software and the software development process. [47]

### 2.2.1 UML Diagram Types

UML uses elements and associates them in different ways to form diagrams that represent static, or structural aspects of a system, and behavioral diagrams, which capture the dynamic aspects of a system. There are 13 Diagrams in UML: Class Diagram, Component Diagram, Composite Structure Diagram, Object Diagram, Package Diagram, Activity Diagrams, Communication Diagram, Interaction Overview Diagram, Sequence Diagram, State Machine Diagram, Timing Diagram, Use Case Diagram This figure shows how types of UML structured :

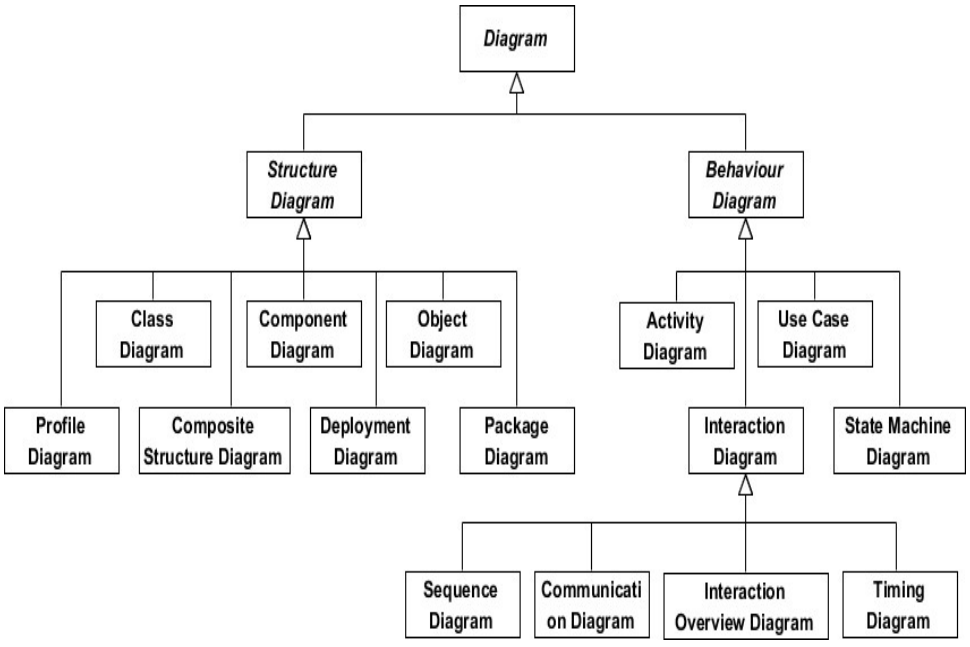


Figure 2.1: types of UML Diagrams

In our study, we will use the Use case, Sequence, Class, and Object diagrams

**Use case Diagram :**

Represents a particular functionality of a system, created to illustrate how functionalities relate and their internal/external controllers (actors).

**Sequence Diagram :**

Sequence Diagram Shows how objects interact with each other and the order of occurrence. They represent interactions for a particular scenario.

**Class diagram:**

Class Diagram The most commonly used UML diagram, and the principal foundation of any object-oriented solution. Classes within a system, attributes and operations, and the relationship between each class. Classes are grouped together to create class diagrams when diagramming large systems.

**Object diagram**

An object is an instance of a class in a particular moment in runtime that can have its own state and data values. Likewise, a static UML object diagram is an instance of a class diagram; it shows a snapshot of the detailed state of a system at a point in time, thus an object diagram encompasses objects and their relationships which may be considered a special case of a class diagram or a communication diagram.

### 2.2.2 Drafting of specification

**A - Project Display**

Di-Statistics system is a backend-based application as ( RESTful API ) with a client we called Di-Statistics-vuejs, that means this client built with vue.js framework.

**B - Actors and functional requirements**

Di-Statistics system has two users, they are Researcher and Administrator.

**Researcher :**is the person who use the system and he can:

- Sign in to Di-Statistics.
- Select Entitytype statistics.
- Select Field statistics
- Compare two statistics based on criteria ( filters ).

- Export data (statistics)

**Administrator :** is the manager of the system, he can:

- Select Sensitive data.
- Manage the users (add, update, delete user).

**C– Use case diagram:**

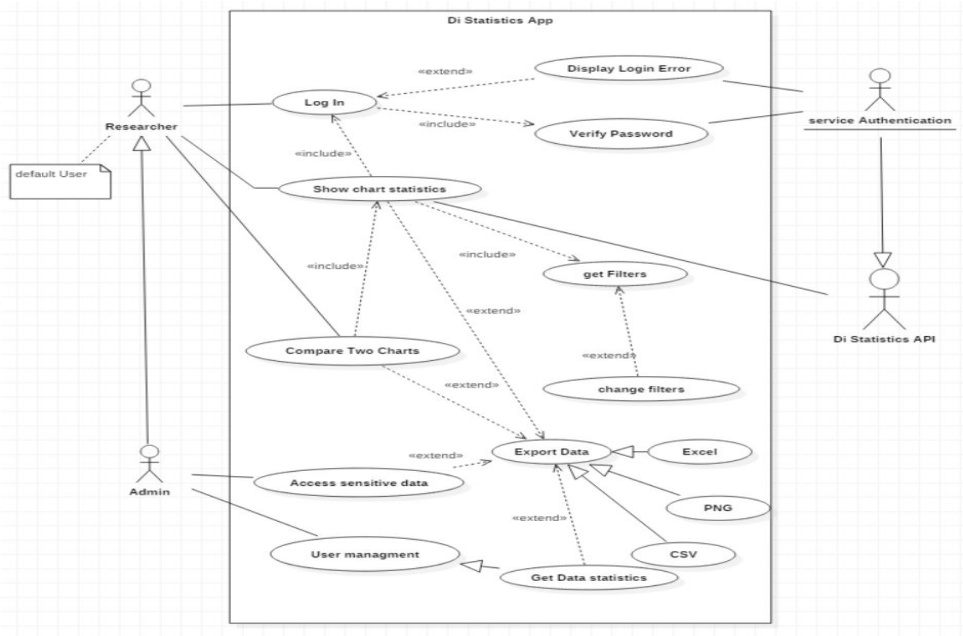


Figure 2.2: Use case Diagram

**D – Sign in (Researcher)**

Textual description of sign-in.

| | |
|---|---|
| **Actor** | Researcher |
| **Precondition** | The researcher is not logged in, and already has an account. |
| **Post-condition** | The system displays the researcher's home page (Dashboard). |
| **Nominal Scenario** | 1-The researcher asks to sign in using email and password.<br>2- The System checks the existence of the account.<br>3- The system display the researcher's home page. |
| **Optional Scenario** | 1- The researcher uses username instead of email. |
| **Alternative scenario** | 3- The system asks the researcher to retry. |

Table 2.1: Textual description of sign in.

Sequence diagram of sign in.

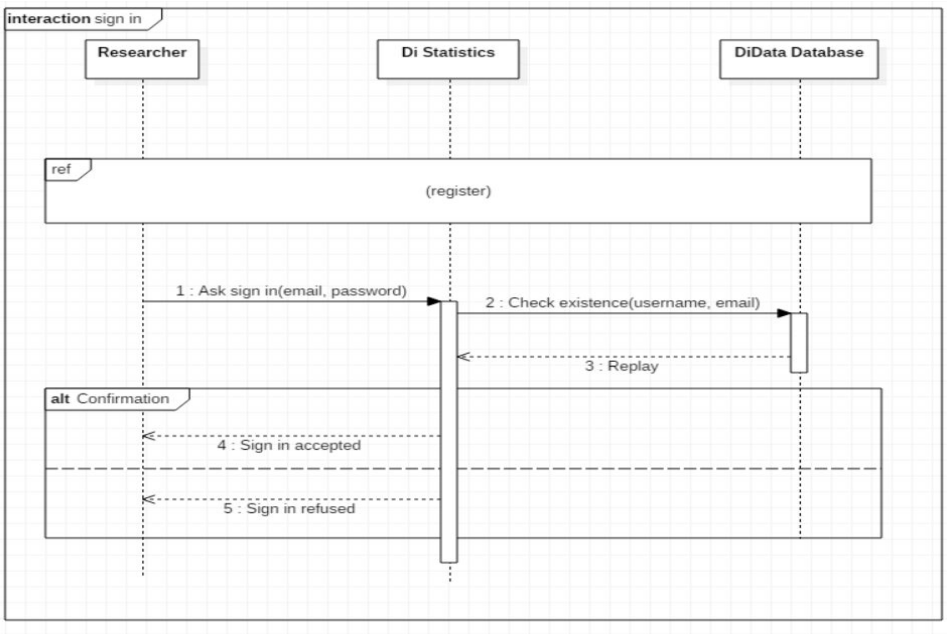


Figure 2.3: Sequence diagram of sign in.

**E – Show statistics chart**

| | |
|---|---|
| **Actor** | Researcher |
| **Precondition** | The researcher must be logged in. |
| **Post-condition** | The system displays the chart of statistics. |
| **Nominal Scenario** | 1- The researcher selects an Entitytype.<br>2- The researcher select criteria and filters.<br>3- The System shows chart of statistics.<br>4- Export the data statistics |
| **Alternative scenario** | 4- The researcher exit before exporting data statistics. |

Table 2.2: Textual description of Show statistics chart ( Researcher ).

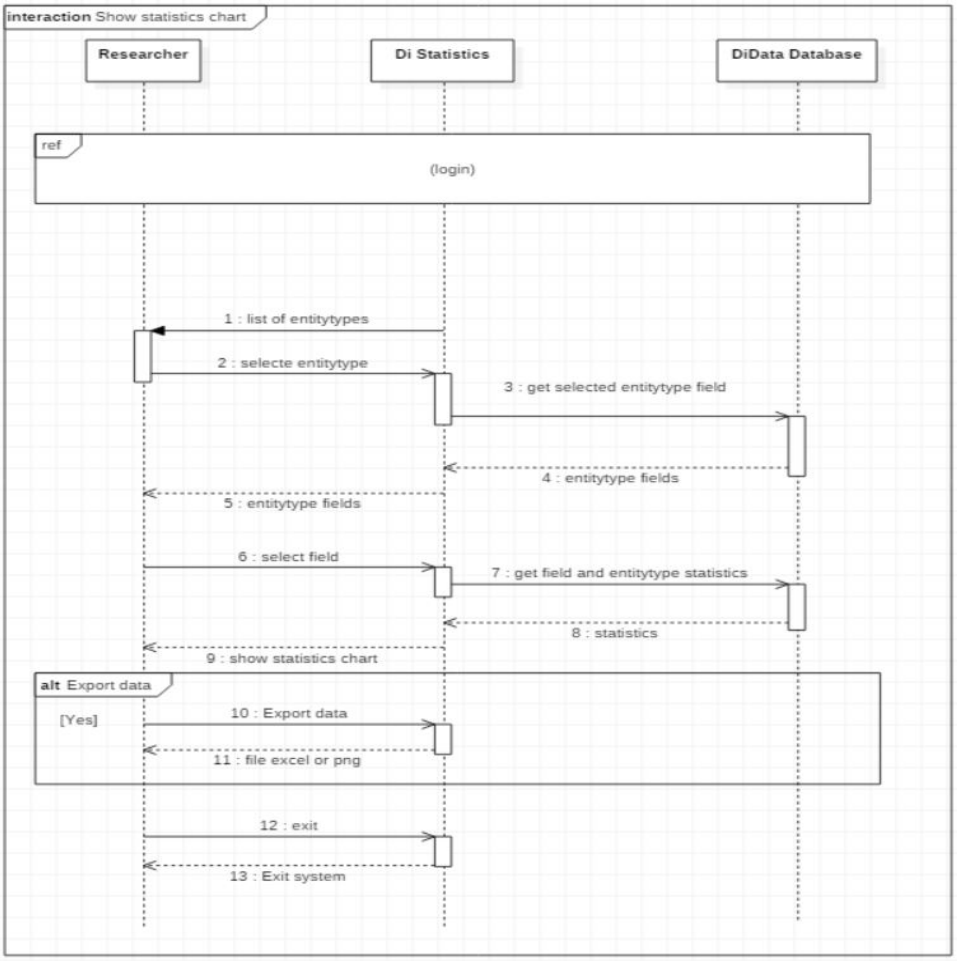


Figure 2.4: Sequence diagram of Show statistics chart.

**F – Show Users Statistics:**

| | |
|---|---|
| **Actor** | Admin |
| **Precondition** | Login as an Admin. |
| **Post-condition** | The system displays the users statistics. |
| **Nominal Scenario** | 1- The admin asks for users list.<br>2- The system shows users list.<br>3- Select one or more users for showing statistics.<br>4- System responses with chart of users data statistics.<br>5- Export users data statistics. |
| **Alternative scenario** | 5- The Admin only browse users statistics and doesn't need export this data. |

Table 2.3: Textual description of Show Users Statistics ( Admin ).

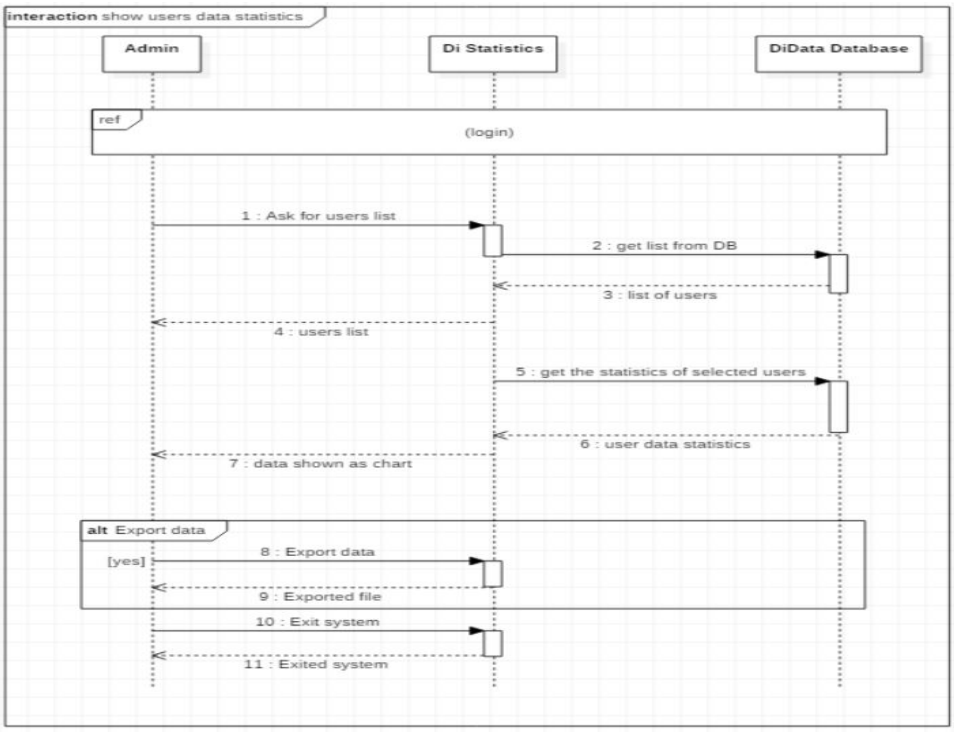


Figure 2.5: Sequence diagram of Show statistics chart.

### 2.2.3 Di-Statistics core-classes :

These are the most important classes used in our study :

- Entitytype.
- Entity.
- Field.

| | **defnition** | **example** |
|---|---|---|
| **Entity** | A thing in the real world with independent existence | Any particular row (a record) in a relation(table) is known as an entity. |
| **Entity Type** | A category of a particular entity. | The name of a relation (table) in RDBMS is an entity type |
| **Field** | A column or an attribute of a an Entityttpe. | Ex : id, name, … |

Table 2.4: Di-Statistics core-classes.

**Relation between Entity and Entitytype:**

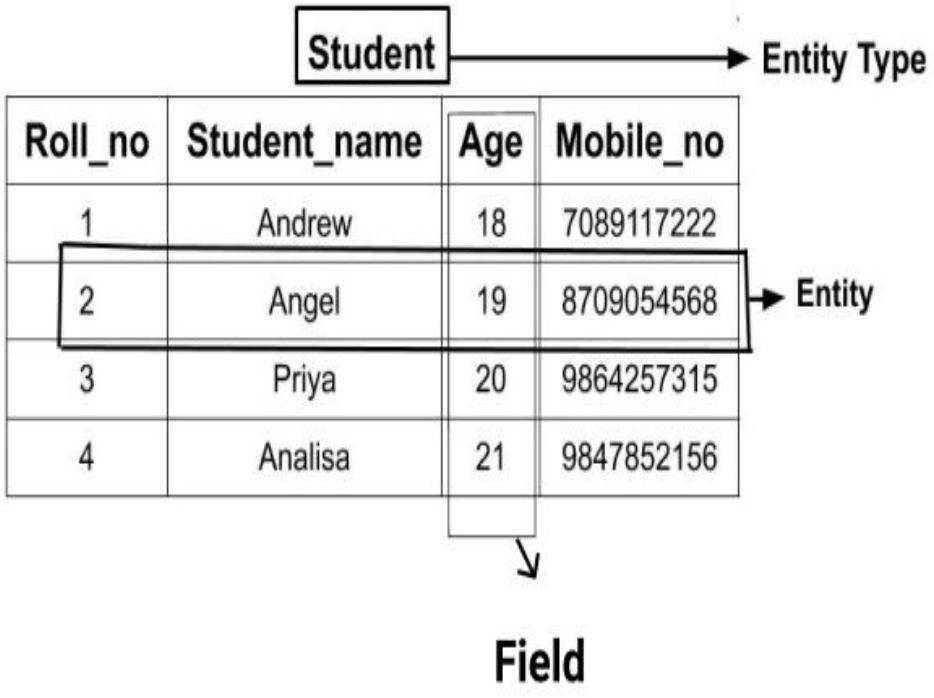


Figure 2.6: Relation between Entity and Entitytype.

In the previous figure, the entitytype is Student class, and every row of the table Is a student object, So the relation between them:

- Entitytype has many Entities
- Entitytype has many fields
- Entity has many fieldValue (value of each field)

**Note:** We decide the type of the statistics charts Based on the Fieldtype of a field (column chart, pie chart, ...)

## 2.3 Class Diagram

### 2.3.1 Class diagram conception

| Class | Codification | Description | Type |
|---|---|---|---|
| **User** | id | The id of the user. | int. |
| | username | The username of the user | String |
| | email | The email of the user | String |
| | password | The password of the user | String |
| | type | The type of the user | Enumuration |
| | first-name | The first name of the user | String |
| | last-name | The last-name of the user | String |
| | project-id | The id of the project of user | int |
| **Project** | id | The id of the | int |
| | name | project of the user. | String. |
| | | The name of the project | |
| **Entitytype** | id | The id of the entitytype. | int |
| | name | The name of the entitytype | String. |
| | label | The label of the entitytype | String. |
| | project-id | The id project of entitytype | int |
| **Entity** | id | The id of the entity. | int |
| | entitytype-id | The id entitytype of the | int. |
| | | entity | |
| **Field** | id | The id of the field. | int |
| | name | The name of the field | String. |
| | lable | The lable of the field | String. |
| | fieldtype | The id of a fieldtype | String |
| | is-sensitive | The id of the field | boolean |
| **FieldEntitytype** | field-id | The id of the field. | int |
| | entitytype-id | The id of the entitytype | int. |
| **FieldValue** | entity-id | The id of the entity. | int |
| | field-id | The id of the field | int. |
| | value | The value of the field by | String |
| | | the entity | |

Table 2.5: Class diagram conception

### 2.3.2 Class Diagram

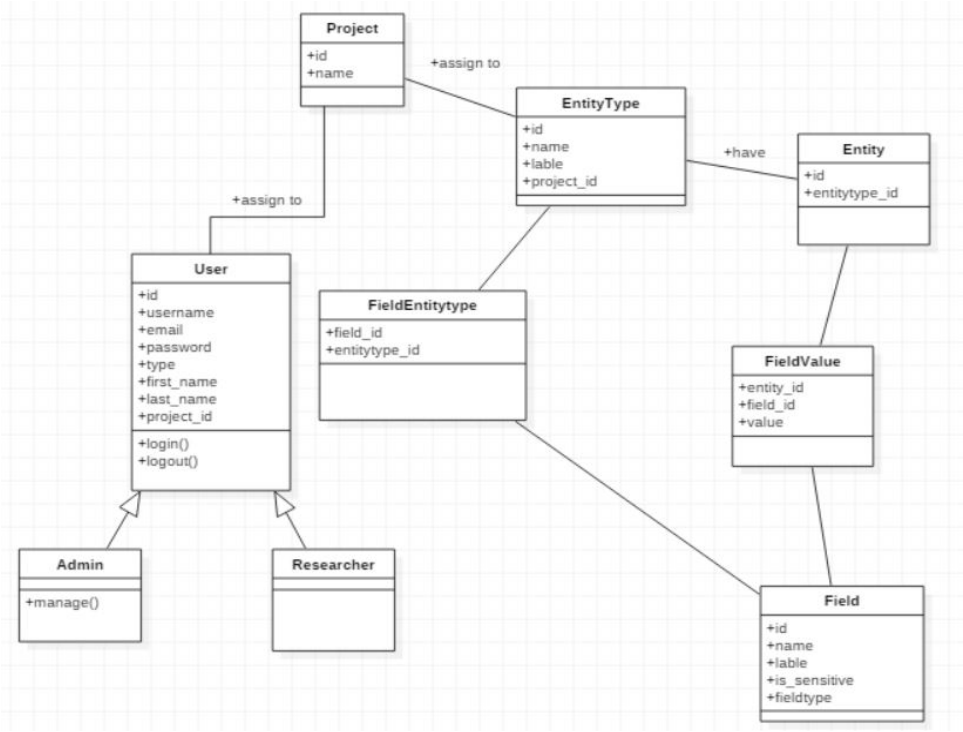


Figure 2.7: Class diagram.

### 2.3.3 Object Diagram

In this diagram, we are going to simplify the class diagram with implement some objects on it.

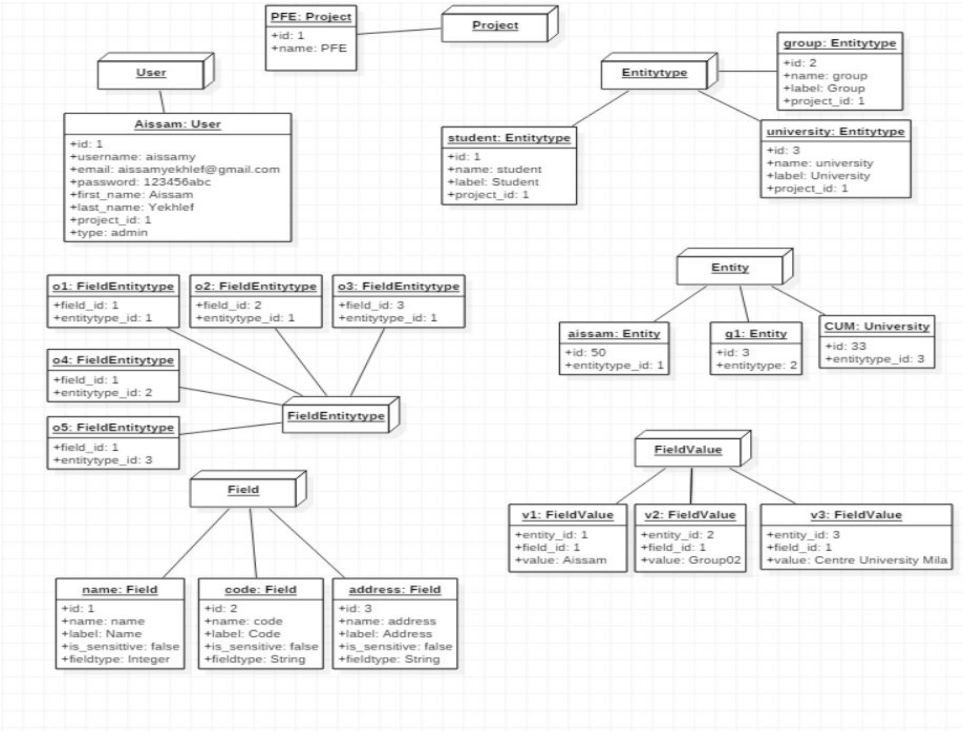


Figure 2.8: Object Diagram.

**Object Diagram of Di-Statistics system**

**Why this Diagram is good for our system ?** In this diagram we showed the power of our system, if we increase the number of entitytypes (usually systems tables), we still have the same tables in our database (just the same 7 tables) And the same field, no need to create a field already have created, unless we need a new Field, we create it and assign it to an Entitytype,

This design helps us to reuse the fields, And we can change the data type of a field just one time, not on all the tables and columns It's a reusable design for all the Entities we need to create.

## 2.4 Conclusion

In this chapter, we have presented the textual description of use case, sequence diagrams, class diagram, and object diagram of our application. The next chapter will be about the tools and programming languages that are we used in the implementation.

# Chapter 3

# Implementation

# Part I

# Tools and Technologies

## 3.1 Introduction

In this part, we will discuss about all the technologies and tools we have used to implement our application.

## 3.2 The Terms and Tools used

### 3.2.1 HTML

HyperText Markup Language (HTML) is the most basic building block of the Web. It defines the meaning and structure of web content. Other technologies besides HTML are generally used to describe a web page's appearance/presentation (CSS) or functionality/behavior (JavaScript). [10]

### 3.2.2 CSS

Cascading Style Sheets (CSS) is a stylesheet language used to describe the presentation of a document written in HTML or XML (including XML dialects such as SVG, MathML or XHTML). CSS describes how elements should be rendered on screen, on paper, in speech, or on other media. [11]

### 3.2.3 JavaScript

JavaScript (JS) is a lightweight, interpreted, or just-in-time compiled programming language with first-class functions. While it is most well-known as the scripting language for Web pages, many non-browser environments also use it, such as Node.js, Apache CouchDB, and Adobe Acrobat. JavaScript is a prototype-based, multi-paradigm, single-threaded, dynamic language, supporting object-oriented, imperative, and declarative (e.g. functional programming) styles. [12]

### 3.2.4 HTTP

Hypertext Transfer Protocol (HTTP) is an application-layer protocol for transmitting hypermedia documents, such as HTML. It was designed for communication between web browsers and web servers, but it can also be used for other purposes. HTTP follows a classical client-server model, with a client opening a connection to make a request, then waiting until it receives a response. HTTP is a stateless protocol, meaning that the server does not keep any data (state) between two requests. [13]

### 3.2.5 API

API stands for application programming interface, which is a set of definitions and protocols for building and integrating application software. [16]

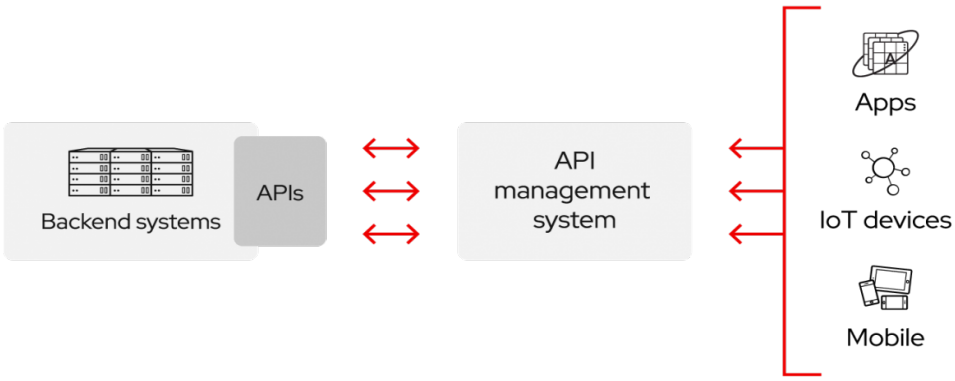


Figure 3.1: API concept.

### 3.2.6 REST

REST is an acronym for REpresentational State Transfer. It is an architectural style for distributed hypermedia systems and was first presented by Roy Fielding in 2000 in his famous dissertation. [14]

### 3.2.7 SOAP

SOAP is a standard protocol that was first designed so that applications built with different languages and on different platforms could communicate. Because it is a protocol, it imposes built-in rules that increase its complexity and overhead, which can lead to longer page load times. However, these standards also offer built-in compliances that can make it preferable for enterprise scenarios. The built-in compliance standards include security, atomicity, consistency, isolation, and durability (ACID), which is a set of properties for ensuring reliable database transactions. [15]

### 3.2.8 JSON

JSON (JavaScript Object Notation) is a lightweight data-interchange format. It is easy for humans to read and write. It is easy for machines to parse and generate. It is based on a subset of the JavaScript Programming Language Standard ECMA-262 3rd Edition - December 1999. JSON is a text format that is completely language-independent but uses conventions that are familiar to programmers of the C-family of languages, including C, C++, Java, JavaScript, Perl, Python, and many others. These properties make JSON an ideal data-interchange language. [17]

### 3.2.9 JWT

JSON Web Token (JWT) is an open standard (RFC 7519) that defines a compact and self-contained way for securely transmitting information between parties as a JSON object. This information can be verified and trusted because it is digitally signed. JWTs can be signed using a secret (with the HMAC algorithm) or a public/private key pair using RSA or ECDSA. [18]

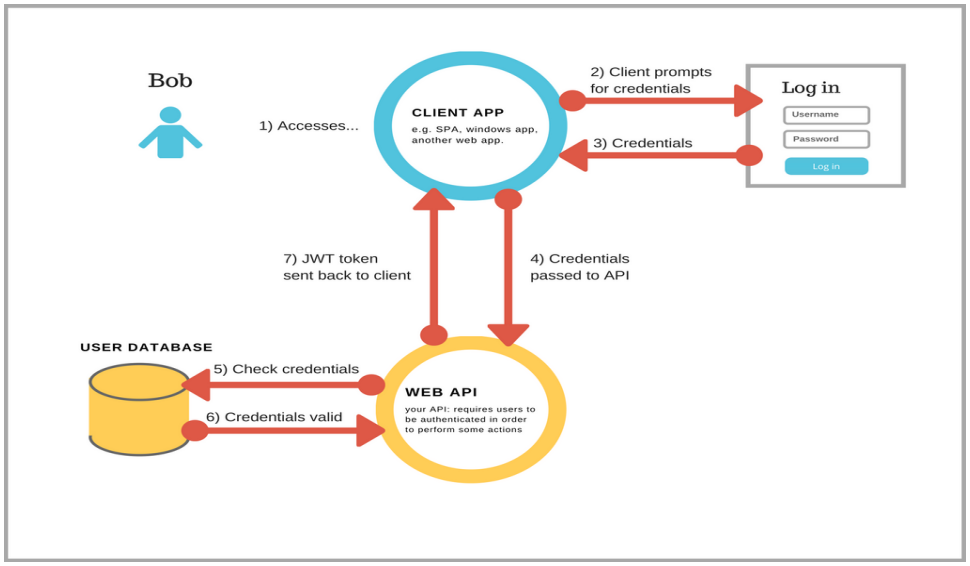


Figure 3.2: JWT concept .

### 3.2.10 Postman

Postman is an API platform for building and using APIs. Postman simplifies each step of the API lifecycle and streamlines collaboration so you can create better APIs—faster. [51]

Figure 3.3: Postman logo.

### 3.2.11 Framework

a framework is a real or conceptual structure intended to serve as a support or guide for the building of something that expands the structure into something useful. [19]

### 3.2.12 JS-Framwork

JavaScript frameworks are an essential part of modern front-end web development, providing developers with tried and tested tools for building scalable, interactive web applications. Many modern companies use frameworks as a standard part of their tooling, so many front-end development jobs now require framework experience. In this set of articles, we are aiming to give you a comfortable starting point to help you begin learning frameworks. [20]

### 3.2.13 Vue.js

Vue (pronounced /vju□/, like view) is a progressive framework for building user interfaces. Unlike other monolithic frameworks, Vue is designed from the ground up to be incrementally adoptable. The core library is focused on the view layer only and is easy to pick up and integrate with other libraries or existing projects. On the other hand, Vue is also perfectly capable of powering sophisticated Single-Page Applications (SPA) when used in combination with modern tooling and supporting libraries. [21]

### 3.2.14 Node.js

Node.js is a JavaScript runtime built on Chrome's V8 JavaScript engine, can Node.js uses JavaScript on the server, it's free and open-source server environment. [23]

### 3.2.15 npm

npm - npm is the world's largest software registry. Open-source developers from every continent use npm to share and borrow packages, and many organizations use npm to manage private development as well. [24]

### 3.2.16 Wampserver

Wampserver is a Windows web development environment. It allows you to create web applications with Apache2, PHP, and a MySQL database. Alongside, PhpMyAdmin allows you to manage easily your databases. [25]


Figure 3.4: wampserver Logo.

### 3.2.17 Apache

The Apache HTTP Server Project is an effort to develop and maintain an open-source HTTP server for modern operating systems including UNIX and Windows. The goal of this project is to provide a secure, efficient, and extensible server that provides HTTP services in sync with the current HTTP standards. [26]

### 3.2.18 PHP

PHP is a popular general-purpose scripting language that is especially suited to web development.it is fast, flexible, and pragmatic, PHP powers everything from blogs to the most popular websites in the world. [27]

### 3.2.19 Composer

Composer is a tool for dependency management in PHP. It allows you to declare the libraries your project depends on and it will manage (install/update) them for you. [28]

### 3.2.20 Laravel

Laravel is a web application framework with expressive, elegant syntax. We've already laid the foundation — freeing you to create without sweating the small things. [29]

### 3.2.21 DBMS

DBMS software primarily functions as an interface between the end-user and the database, simultaneously managing the data, the database engine, and the database schema to facilitate the organization and manipulation of data. [30]

### 3.2.22 MySQL

MySQL, the most popular Open Source SQL database management system, is developed, distributed, and supported by Oracle Corporation. [31]

### 3.2.23 Visual Studio Code

Visual Studio Code is a lightweight but powerful source code editor which runs on your desktop and is available for Windows, macOS and Linux.. [32]

### 3.2.24 Google chrome

The browser built by Google. [33]

### 3.2.25 git

Git is a free and open-source distributed version control system designed to handle everything from small to very large projects with speed and efficiency. [34]

### 3.2.26 github

GitHub is a web-based interface that uses Git, the open-source version control software that lets multiple people make separate changes to web pages at the same time. [35]

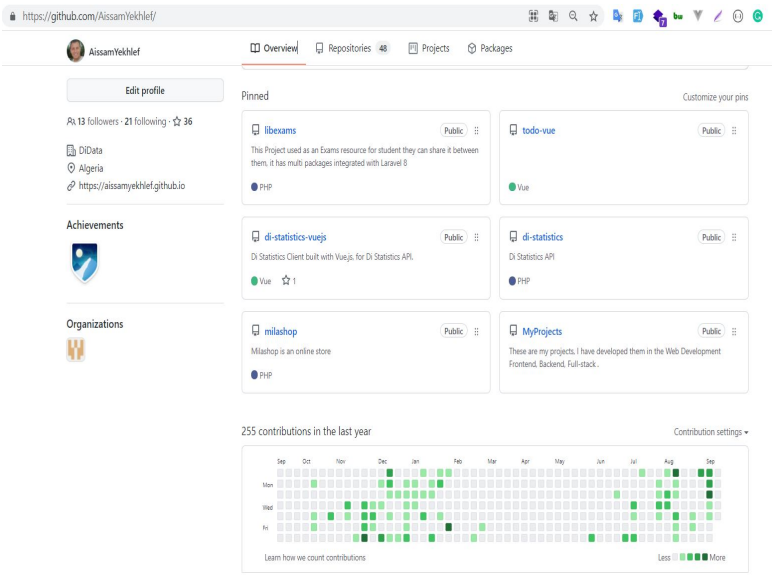


Figure 3.5: github personal account.

### 3.2.27 LaTex

LaTeX is a high-quality typesetting system; it includes features designed for the production of technical and scientific documentation. LaTeX is the de facto standard for the communication and publication of scientific documents. [36]

### 3.2.28 Overleaf

a web-based, easy to use, online, collaborative LaTeX editor. [37]

### 3.2.29 Figma

Figma is a vector graphics editor and prototyping tool. [38]

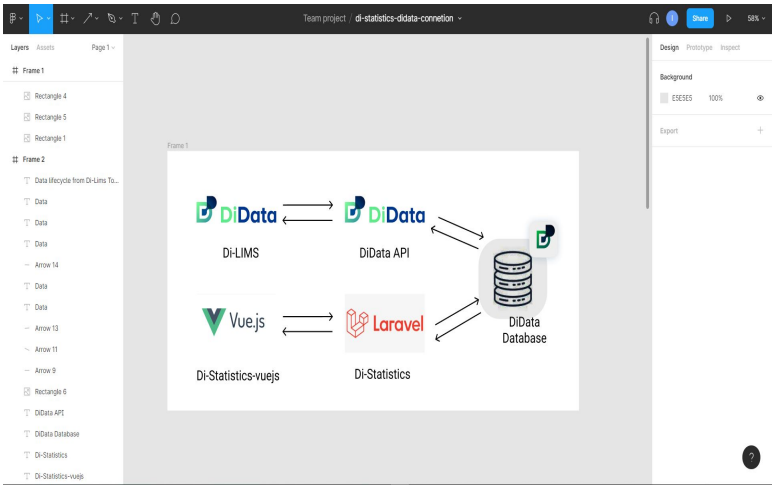


Figure 3.6: figma workspace.

### 3.2.30 StartUML

A sophisticated software modeler for agile and concise modeling. [39]

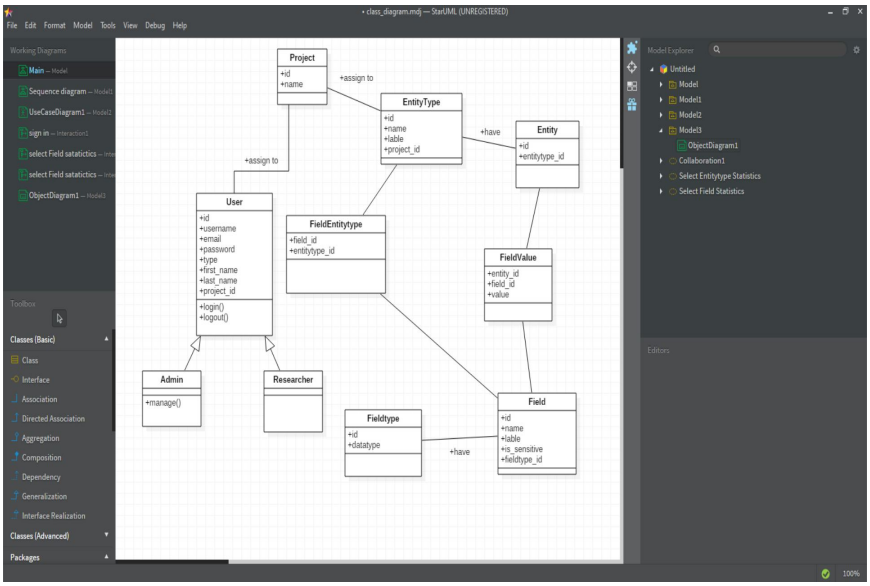


Figure 3.7: staruml window.

### 3.2.31 Axios

Axios is a simple promise-based HTTP client for the browser and node.js. Axios provides a simple to use library in a small package with a very extensible interface. [40]

### 3.2.32 Vuex

Vuex is a state management pattern + library for Vue.js applications. It serves as a centralized store for all the components in an application. [22]

### 3.2.33 Vuetify

Vuetify is a complete UI framework built on top of Vue.js. The goal of the project is to provide developers with the tools they need to build rich and engaging user experiences. [41]

### 3.2.34 Apexchartjs

ApexCharts is a modern charting library that helps developers to create beautiful and interactive visualizations for web pages. [42]

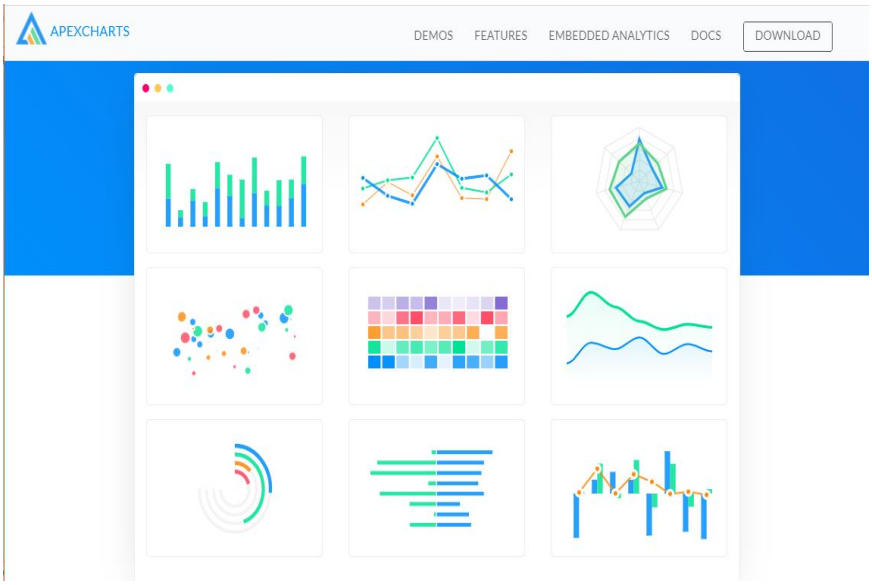


Figure 3.8: apexchart.

### 3.2.35 MVC

MVC (Model-View-Controller) is a pattern in software design commonly used to implement user interfaces, data, and controlling logic. [44]

### 3.2.36 MVVM

MVVM (Model-View-Viewmodel) is a pattern in software design commonly used to implement user interfaces and started from the view and returned to it. [45]

### 3.2.37 SPA

Single Page Application is a term used for the application that has only one page, and this page just change the components inside it, use AJAX [52] to get the resource from the API, so no need to reload all the whole page, used by Twitter, Facebook, ...

# Part II

# Work Steps

In this part, we will talk about the process of our work From the installation to the test part First of all, we need to install these tools :

## 3.3 Backend project Di-Statistics:

We follow these steps :

- Install wampserver (PHP, Apache, MySQL).
- install composer
- install Laravel installer by running this command :

```
composer global require laravel/installer
```

- create new Laravel project by run these commands :

```
laravel new di-statistics
cd di-statistics
```

- adding the Authentication by installing Jwt (JSON Web Token) by run this command :

```
composer required tymon/jwt-auth
```

- install VScode (Visual studio code)
- install git

## 3.4 Frontend project Di-Statistics-vuejs:

- install Node.js
- install vue.js CLI globally by npm.

```
npm install -g @vue/cli
```

- create new vue project.

```
vue create di-statistics-vuejs
cd di-statistics-vuejs
```

- install vuex.

```
npm install vuex
```

- install vuetify

```
npm install vuetify
```

- install axios

```
npm install axios
init the frontend project git folder
git init
```

- install Google Chrome
- add vue devtools extension to chrome [**?**](https://devtools.vuejs.org/)

## 3.5 Start the coding part:

Now we finished the installation part and we are going to add some coding
To build first the backend app (Di-Statistics) as a RESTful API service How Di-statistics and DiData connects:

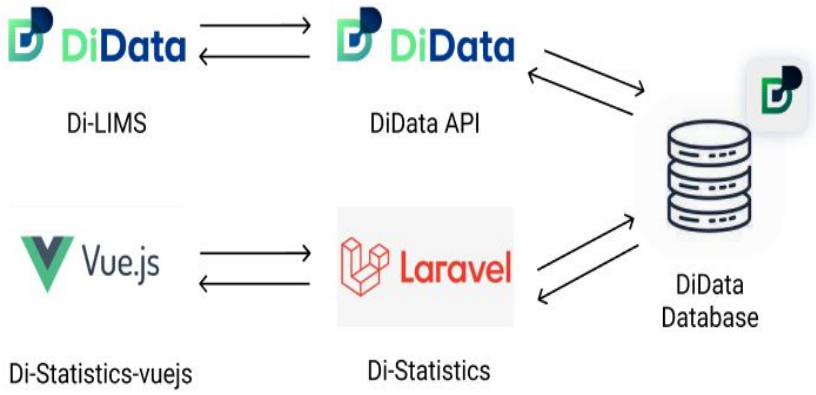


Figure 3.9: didata and di-statistics connection.

So DiData and Di-Statistics share the same database, If some data inserted into DiData DB our system can read this data and use it realtime,

First of all, we need to create some classes like models, controller Laravel by default has a CLI called artisan to simplify the creation of these classes The make:model command for creating a new model and make:controller for creating a new controller class So, we need to create 3 models and 3 controllers:

```
php artisan make:model EntityType
php artisan make:controller EntityType
```

or we can just run one commad for two commads with pass arguments like this:

```
php artisan make:model Entity --controller --api
php artisan make:model Field --controller --api
```

–controller: arg stands for controller
–api: arg stands for resource for generating the REST methods
After that we need to add the relation between the models like this : One-to-many(Entitytype has many entities )

```php
public function _entities()
{
   return $this->hasMany(Entity::class, 'entitytype_id');
}
```

Many-to-many:

```php
public function _fields()
{
      return $this->belongsToMany(Field::class,
      'fieldentitytype', 'entitytype_id', 'field_id')
      ->withTimestamps();
}
```

App/Models/Entitytype.php

entitytype-id: primary key for Entitytype model
fieldentitytype: the association table between Entitytype and Field models we used underscore '-' before the method name that means this method is a relationship, the same thing in the other side Field model: Field has many Entitytypes

```php
public function _entitytypes()
{
       return $this->belongsToMany(
     EntityType::class, 'fieldentitytype', 'field_id', 'entitytype_id'
     )->withPivot('position')->withTimestamps();
}
```

App/Models/Field.php
How the model class looks like after run these commands :

```php
class EntityType extends Model
{   use HasFactory;
    public $table ="EntityType";
    public function _fields()
    {
        return $this
->belongsToMany(Field::class, 'fieldentitytype', 'entitytype_id', 'field_id')
->withTimestamps();
    }
    public function _entities()
    {
        return $this->hasMany(Entity::class, 'entitytype_id');
    } }
class Field extends Model
{
    use HasFactory;

    public $table = 'Field';


    public $fillable = [
        'name',
        'label',
        'fieldtype_id',
        'is_sensitive' ];
    public function _entitytypes()
    {
        return $this
->belongsToMany(EntityType::class,'fieldentitytype','field_id','entitytype_id')
->withTimestamps();
    }
}
```

## 3.6 Build the REST API

This api has all the Endpoints that are we access to them by the prefix api/endpoint.

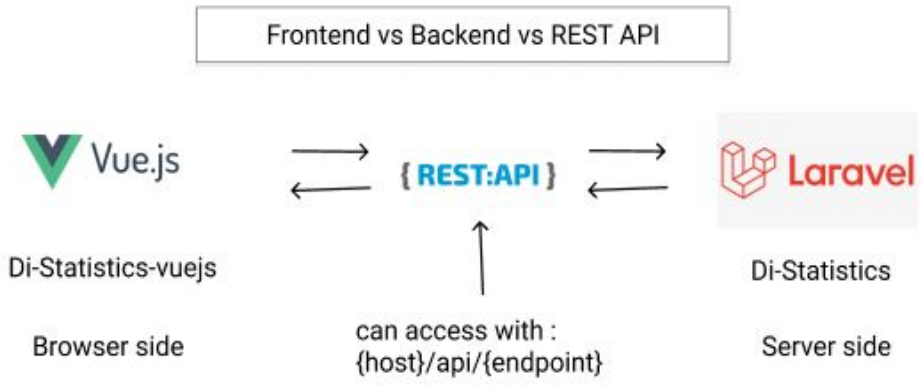


Figure 3.10: create-you-a-rest-api.

The REST API is an endpoints can access to them with authentication -JWT- (except login and register), these routes looks like in the figure below:

```php
Route::group([
    'prefix' => 'auth'

], function () {

    Route::post('login', [AuthController::class, 'login']);
    Route::post('register', [AuthController::class, 'register']);
    Route::post('logout', [AuthController::class, 'logout']);
    Route::post('refresh', [AuthController::class, 'refresh']);
    Route::post('me', [AuthController::class, 'me']);

});
Route::group([
    'middleware' => 'auth.jwt',

], function () {

    Route::get('/fields', [FieldController::class, 'index']);
    Route::get('/entitytypes', [EntityTypeController::class, 'index']);
    Route::get('/entitytypes/{entityType}', [EntityTypeController::class, 'show']
;
    Route::get('/entitytypes/{entityType}/fields', [EntityTypeController::class,
'fields']);
    Route::group([
        'prefix' => 'statistics',
    ],
    function(){
        Route::get('fields/{field}',
                    StatisticsController::class, 'field_statistics']
        );
      }); });
```

routes/api.php

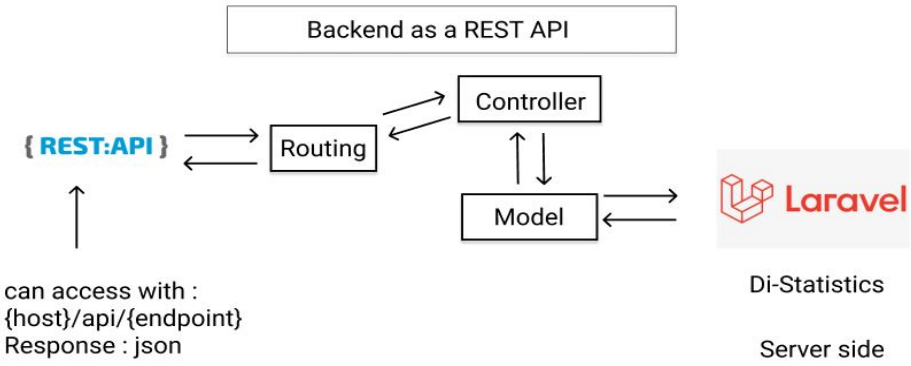


Figure 3.11: backend-as-resr-api.

## 3.7 Coding the Frontend Side

In this section, we will be able to use the REST API, that we have created in the previous section. So, first of all we need to take a look at how di-statistics-vuejs connect to the REST API how it works:

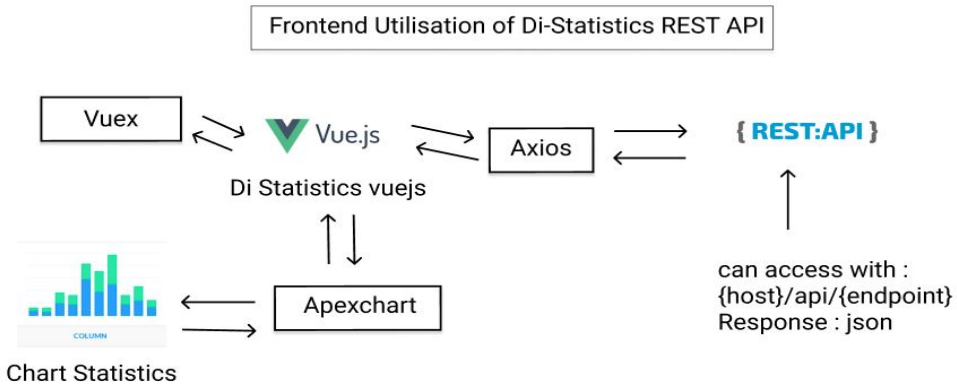


Figure 3.12: frontend-rest-api.

Now, we will take a look at the code itself:

```
const actions = {
   async load_entitytypes({commit}) {
     return await  axios
      .get("/api/entitytypes",
        { params: {with_fields: true}  }
      ).then( ({ data }) => {
          commit('setEntitytypes', data);
        return data;
      });
  },

}
```

This is an example of how get data (entitytypes) from the rest api using axios : The response is looks like this :


```
[
   {
       "id": 1,
       "context_id": 1,
       "name": "Subject",
       "label": "Subject",
       "created_at": "2021-08-27T23:53:07.000000Z",
       "updated_at": "2021-08-27T23:53:07.000000Z",
       "icon": null
   },
   {
       "id": 2,
       "context_id": 1,
       "name": "Visit",
       "label": "Visit",
       "created_at": "2021-08-27T23:53:07.000000Z",
       "updated_at": "2021-08-27T23:53:07.000000Z",
       "icon": null
   },
]
```


Data response by filters – criteria-


Series is the list of our statistics
Count => is the number of created entities
year => the year of creation
mounth => the mounth of creation
day => the day of creation
there are also :
weekly => the week number of creation
hourly => the exact hour of creation

## 3.8 Pages Screenshots:

Login page, Dashboard page, Statistics page.

### 3.8.1 Login page

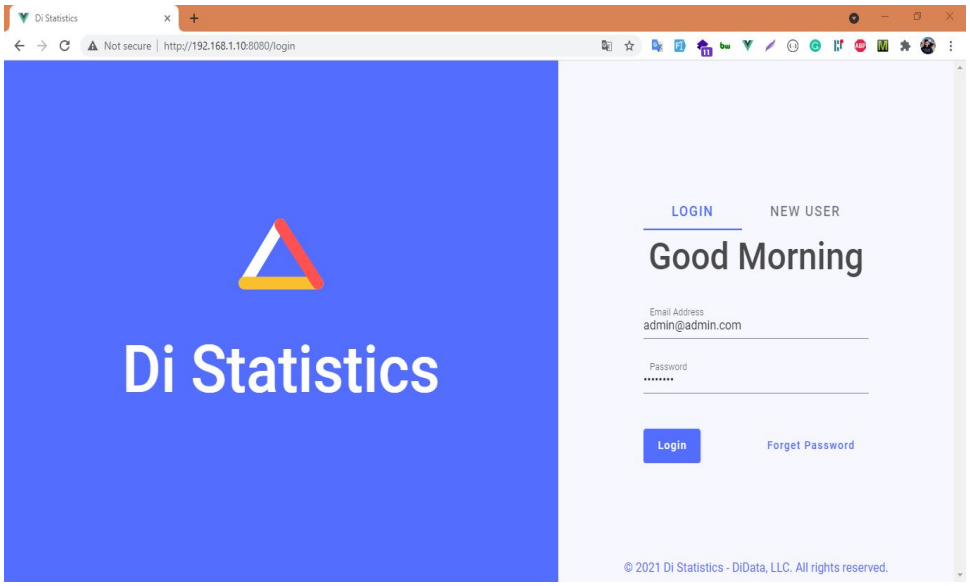


Figure 3.13: Login di-statistics page.

### 3.8.2 Dashboard page

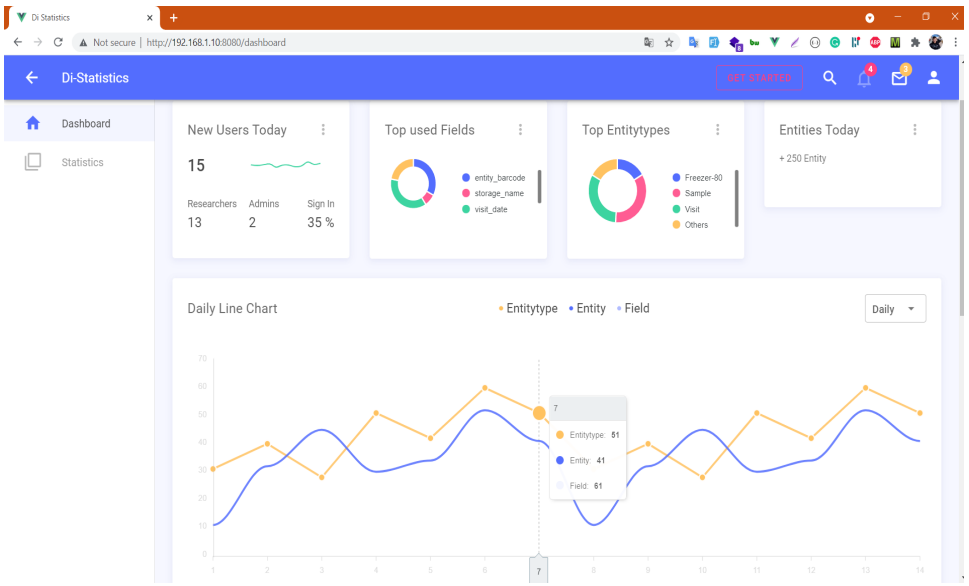


Figure 3.14: Dashboard di-statistics.

### 3.8.3 Profile Section

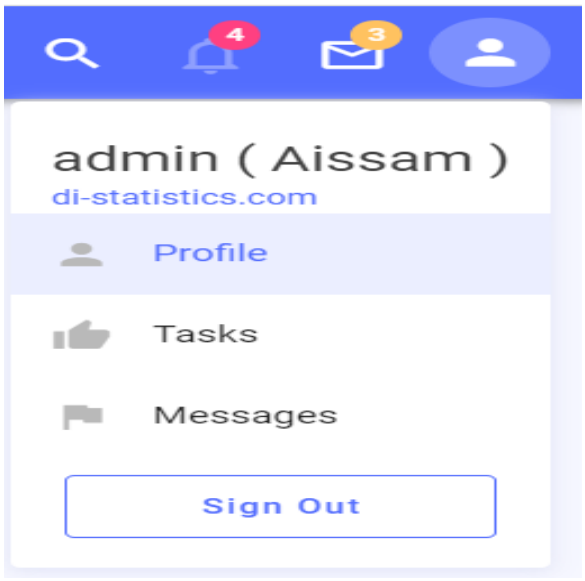


Figure 3.15: Profile section.

### 3.8.4 Statistics page

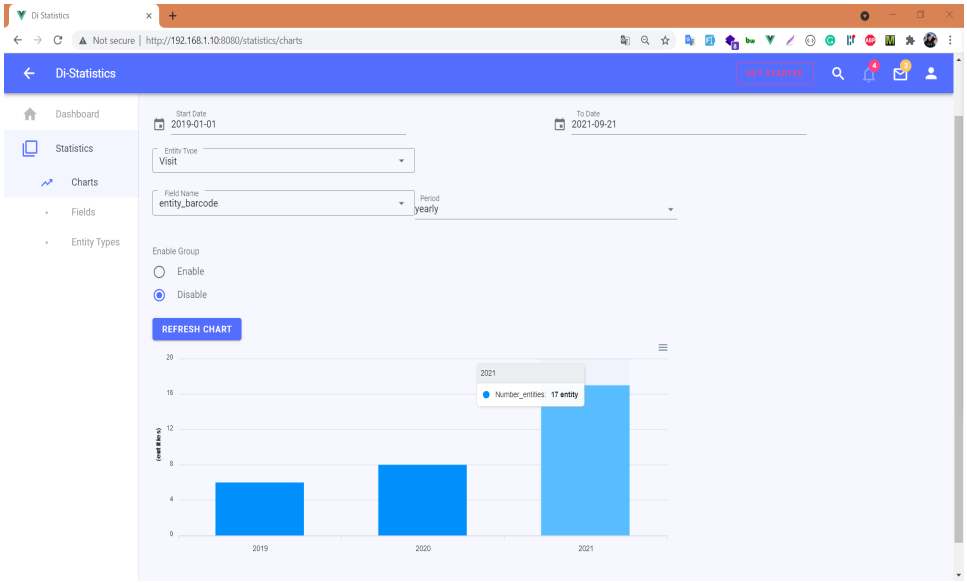


Figure 3.16: Statistics page.

### 3.8.5 Statistics page with filters

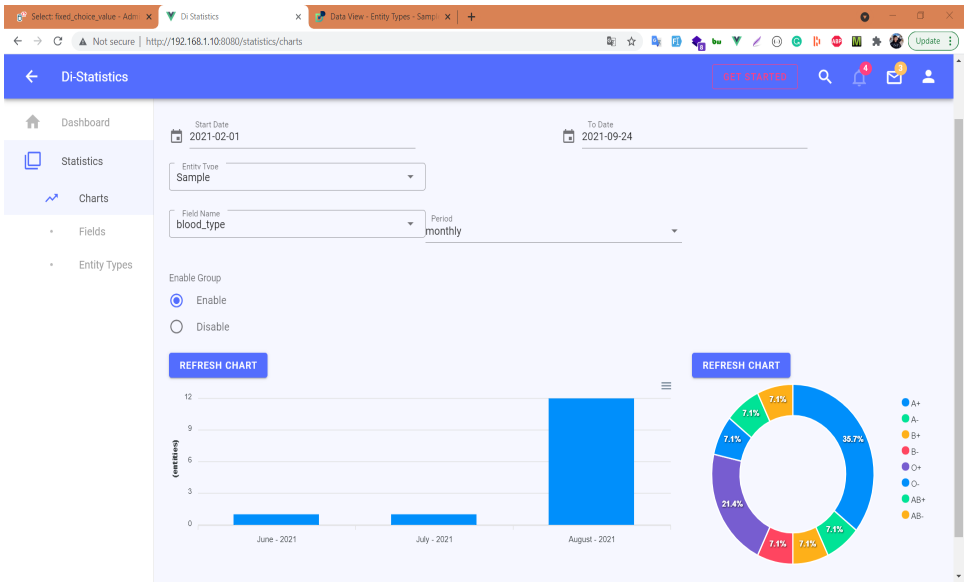


Figure 3.17: Statistics Page with filters

## 3.9 Conclusion

In this chapter, we have presented some aspects of the implementation of the Di-Statistics web app, We enriched it with some screenshots of our application.

# Gerneral Conclusion

The objective of this thesis is to develop a Software as a Service (SaaS) system for DiData system to help in Statistics operations. To realize this project, we studied the need for DiData and Analyze. We proposed in this thesis a Data Analytic system called **Di-Statistics**, composed of two web applications one for Di-Statistics as a **REST API** that can read the Data from the DiData Database and Manage the need of Researcher to Analyze between two periods of time, the second web application is a **client** for our REST API service, which can request statistics with special criteria.

The **main feature** of this system is the use of web technologies to analyze data securely, and create a layer of Data Analitics for DiData to share the Analitics and Statictics of Data to know the insights.

**Finally**, we like to mention that there are a lot of ideas we want to realize in the future.

For example, we can use other data source not only for DiData system to build a Data Analitycs system has a lot of features that are compatible with other companies systems.

# Bibliography


[1] DiData website : https://swissdidata.com

[2] Data Analytics [Online]. Available: : https://www.investopedia.com/terms/d/data-analytics.asp

[3] what is data analytics? [Online]. Available: : https://www.mastersindatascience.org/learning/what-is-data-analytics

[4] data science vs data analytics [Online]. Available:: https://www.upgrad.com/blog/data-science-vs-data-analytics

[5] R project [Online]. Available: : https://www.r-project.org

[6] Tableau the analytics tool [Online]. Available: : https://www.tableau.com

[7] Machine Learning [Online]. Available: : https://www.sas.com/en/insights/analytics/machine-learning.html

[8] What is Data Science? [Online]. Available: : https://www.oracle.com/be/data-science/what-is-data-science/

[9] Microsoft Power BI website [Online]. Available: : https://powerbi.microsoft.com/en-us/what-is-power-bi

[10] HTML [Online]. Available: : https://developer.mozilla.org/en-US/docs/Web/HTML

[11] CSS [Online]. Available: : https://developer.mozilla.org/en-US/docs/Web/CSS

[12] JavaScript [Online]. Available: : https://developer.mozilla.org/en-US/docs/Web/JavaScript

[13] HTTP [Online]. Available: : https://developer.mozilla.org/en-US/docs/Web/HTTP

[14] REST [Online]. Available: : https://restfulapi.net

[15] SOAP [Online]. Available: : https://www.redhat.com/en/topics/integration/whats-the-difference-between-soap-rest

[16] Application Programming Interface [Online]. Available: : https://www.redhat.com/en/topics/api/what-are-application-programming-interfaces

[17] JSON [Online]. Available: : https://www.json.org/json-en.html

[18] JWT [Online]. Available: : https://jwt.io/introduction

[19] Framework [Online]. Available: : https://whatis.techtarget.com/definition/framework

[20] JavaScript Framworks [Online]. Available: : https://developer.mozilla.org/en-US/docs/Learn/Tools_and_testing/Client-side_JavaScript_frameworks

[21] Vue.js [Online]. Available: : https://vuejs.org

[22] Vuex [Online]. Available: : https://vuex.vuejs.org

[23] nodejs [Online]. Available: : https://nodejs.org/en

[24] npm [Online]. Available: : https://docs.npmjs.com/about-npm

[25] Wampserver [Online]. Available: : https://www.wampserver.com/en

[26] Apache [Online]. Available: : http://httpd.apache.org

[27] PHP [Online]. Available: : https://www.php.net

[28] composer [Online]. Available: : https://getcomposer.org/

[29] Laravel [Online]. Available: : https://laravel.com

[30] Database managment system [Online]. Available: : https://www.omnisci.com/technical-glossary/dbms

[31] MySQL [Online]. Available: : https://dev.mysql.com/doc/refman/8.0/en/what-is-mysql.html

[32] Visual Studio Code [Online]. Available: : https://code.visualstudio.com/docs

[33] Google chrome [Online]. Available: : https://www.google.com/chrome

[34] git [Online]. Available: : https://git-scm.com

[35] GitHub [Online]. Available: : https://github.com

[36] LaTex [Online]. Available: : https://www.latex-project.org

[37] Overleaf [Online]. Available: : https://www.overleaf.com

[38] Figma [Online]. Available: : https://www.figma.com

[39] StartUML [Online]. Available: : https://staruml.io

[40] Axios [Online]. Available: : https://axios-http.com

[41] Vuetify [Online]. Available: : https://vuetifyjs.com/en/introduction/why-vuetify

[42] Apexchartjs [Online]. Available: : https://apexcharts.com

[43] client-server architecture [Online]. Available: :

[44] MVC [Online]. Available: : https://developer.mozilla.org/en-US/docs/Glossary/MVC

[45] MVVM [Online]. Available: : https://en.wikipedia.org/wiki/Model-view-viewmodel

[46] SPA [Online]. Available: : https://developer.mozilla.org/en-US/docs/Glossary/SPA

[47] What is UML? [Online]. Available: : https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-uml

[48] Types of UML? [Online]. Available: :https://www.lucidchart.com/pages/what-is-UML-unified-modeling-language

[49] Object Diagram [Online]. Available: : https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-object-diagra

[50] Web Service [Online]. Available: https://en.wikipedia.org/wiki/Web_service

[51] Postman [Online]. Available: : https://www.postman.com/product/what-is-postman

[52] AJAX [Online]. Available: : https://developer.mozilla.org/en-US/docs/Glossary/AJAX

[53] What is Open Source? [Online]. Available: : https://opensource.com/resources/what-open-source figuers

[54] REST API Design [Online]. Available: : https://www.altexsoft.com/blog/rest-api-design

[55] JSON Web Token [Online]. Available: : https://jwt.io