

**Polytechnic University of Puerto Rico**

**Department of Computer Engineering and Computer Science**

**COE 4002-39: Capstone I / CS 4002-49: Computer Science Project I**

**WI-20 / Prof. Othoniel Rodriguez**

**Covid-19 Data Visualization & Analysis:**

Software Detailed Design Specifications (SDD)

Version 2.0

|  |  |  |
| --- | --- | --- |
| Name | ID | Program |
| Jose M. Lopez Torres | #72583 | CS |
| Elias J. Marrero Ruiz | #97850 | CS |
| Ernesto Feliciano Muniz | #96861 | COE |
| Diego Canals Reyes | #80877 | COE |

# Table of Contents

[Table of Contents 2](#_Toc73659570)

[Introduction 3](#_Toc73659571)

[Context & Goals 4](#_Toc73659572)

[Team’s Evaluation 4](#_Toc73659573)

[Proposed Solution 4](#_Toc73659574)

[Client Involvement 5](#_Toc73659575)

[Search Process 5](#_Toc73659576)

[Table of Requirements 5](#_Toc73659577)

[Table 1. Graphs 5](#_Toc73659578)

[Table 2. Graph Features 6](#_Toc73659579)

[Table 3. Webpage Features 6](#_Toc73659580)

[Ranking of Alternatives On deciding which framework will be used for the project. The team decided to compare each framework and rate them from one (1) to ten (10) on three (3) categories: 7](#_Toc73659581)

[Learning Curve: This is based on the understanding of each member of the team in relation to the framework that is being evaluated. 7](#_Toc73659582)

[Adaptability: This category was used to evaluate the ability to adjust each framework to our own needs and preferences. 7](#_Toc73659583)

[Resources: In this category the team did a research for each framework to determine which framework provides the best resourceful for our project. 7](#_Toc73659584)

[Table 4. Front-end Framework Learning Curve 7](#_Toc73659585)

[Table 5. Back-end Framework Learning Curve 7](#_Toc73659586)

[Table 6. Software Components 7](#_Toc73659587)

[Security 8](#_Toc73659588)

[Software Architecture 9](#_Toc73659589)

[Detailed Software Architecture 9](#_Toc73659590)

[Perspectives 9](#_Toc73659591)

[Software Components 9](#_Toc73659592)

[Software Connectors 10](#_Toc73659593)

# Introduction

In the following parts we will discuss the team’s evaluation on how we chose the reusable code for this project. Then, we will explain the client’s involvement in the decisions of the team’s evaluation with a brief explanation of the search process for every framework. After that, three (3) tables were made to specify the weight, complexity, and priority of every user story. Then, another three (3) tables to demonstrate how we ranked the frameworks by their learning curve, adaptability and included resources. After this, a diagram of the software’s architecture was made to simplify its complexity, and finally, a complete explanation in detail of every layer in the system.

# Context & Goals

* The outcome aimed for in this document, will be to detail the design and structure as simple and clear as possible. First, the ideal procedure would be to rate every user story with some specific characteristics, like weight and prioritization for example. After that, it would be explained the use of the chosen frameworks, why specific versions, and the applications for each one within the project. Then move on to the detail and specifications of the system’s architecture, this would be accomplished by understanding how the front-end will be working and addressing different tasks, as well as the back-end, and characterizing the database’s involvement with the back-end. Finally, an explanation of some additional connections to the project would be necessary for the desired outcome.

# Team’s Evaluation

* The selection of the reusable framework was mainly made by the learning curve of each software for each member. This evaluation consisted of selecting three different frameworks for each phase of the application (frontend, backend, and database) to be compared with and determine which framework was going to be implemented. A research was made in order to identify which framework would be better suited for our webpage.

# Proposed Solution

* The proposed systems of architecture in the meetings were reduced to the most efficient ones for this project, and these systems were the Layered Architecture and the Model-View-Controller. Basically, the layered architecture consists of a client-server connection in which the processing and data management are separated, this way changes can be made in one layer without affecting the others. On the other hand, the model-vie-controller pattern consists of dividing the program’s logic into those three logical components (model, view, controller), separating the internal representations of data, which also ends up being layer-like. But at the end, the team chose the layered architecture, as it applied and resembled more the idea that we had in mind.

# Client Involvement

* In this specific area, the client Danilo Perez has not been involved as much as the Professor Othoniel Rodriguez, which will be the acting client in this document. The client has suggested a couple of alternatives for re-usable software like libraries (Cucumber, jQuery, ReactJS) and frameworks (Bootstrap, AngularJS, WinJS), which have helped greatly when it came to deciding in what we were going to use to develop the website. Not only that but when we bring ideas, the client helps us by constructively criticizing and sometimes suggesting other ways of producing the same results while being more efficient. We also received ideas on how to structure the architecture of the system, like using a database for the data from the APIs.

# Search Process

* There were three frameworks chosen (one per phase).
  + The three frameworks presented for the front end were React JS, Django, and Angular JS. In this case the React JS was selected because it was the framework with less learning curve for all the members.
  + The three presented for the back end were Node JS, PHP, and Django. Node JS was the framework selected for the website since it presented the lesser learning curve for all members.

# Table of Requirements

In the following tables, we will describe the User Stories’ weight, complexity, and priority for each one of the User Stories.

**Weight:** Based on how much time it would take a team member to complete them.

**Complexity:** Based on the difficulty level to develop and implement them.

**Priority:** Based on which User Stories are the ones that need to be completed first.

## Table 1. Graphs

| Requirement | Weight | Complexity | Priority |
| --- | --- | --- | --- |
| As a user I want to select from a drop-down menu different countries so I can view the data of each country. | High | 6 | 95 |
| As a user I want to be able to see points in the xy scattered plots where the information is visible as numbers. | High | 4 | 85 |
| As a user I want click on the option of graphs in the navbar and go to the graphs section. | High | 4 | 75 |
| As a user I want to view the cases of recovered people, active cases, confirmed cases and number of deaths of a given country and see them as a bar plot. | High | 4 | 65 |
| As a user I want to have a view of 4 plots of each given country in the same page. | Medium | 2 | 55 |
| As a user I want to be able to sort the countries name in the table summary in descending and ascending order. | Medium | 4 | 45 |
| As a user I want to sort the table of Summary Totals by Cases Confirmed , Deaths and Recovered in ascending and descending order. | Medium | 4 | 35 |
| As a user I want to view different age ranges of the cases in a plot so that it can be visualized the data better | Medium | 2 | 25 |
| As a user I want to view total vaccinated people by a time period so that I can review the data | Low | 2 | 15 |

Note: Weight: Low-Medium-High; Complexity: 1-10; Priority: 1-100

## Table 2. Graph Features

|  |  |  |  |
| --- | --- | --- | --- |
| Requirement | Weight | Complexity | Priority |
| As a user I want to be able to download/embed a given graphic to another website so that I can share the data | High | 10 | 95 |
| As a user I want to change the selected graphic from bar plot to a line graph so that it can be visualized differently. | High | 8 | 80 |
| As a user I want to search total cases for a time period so that I can review the data. | Medium | 4 | 65 |
| As a user I want to search total deaths for a time period so that I can review the data. | Medium | 2 | 50 |
| As a user I want to change the current graphic through a button so that I can view a different graph | Medium | 6 | 35 |
| As a user I want to enable a colorblind mode through a button of a graphic so that I can see different colors on the graph | Low | 2 | 20 |
| As a user I want to hover over a graphic so that I can see a specific data point | Low | 2 | 5 |

Note: Weight: Low-Medium-High; Complexity: 1-10; Priority: 1-100

## Table 3. Webpage Features

| Requirement | Weight | Complexity | Priority |
| --- | --- | --- | --- |
| As a user I want to view a weekly summary of cases, deaths and recovered people by day so that I can review the data. | High | 6 | 100 |
| As a user I want to verify the references of the data showed so that I can also verify it. | Medium | 2 | 75 |
| As a user I want to view a summary of cases , deaths and recovered people by day so that I can review the data. | Low | 2 | 50 |
| As a user I want to report an error in the webpage so that it can be fixed | Low | 2 | 25 |

Note: Weight: Low-Medium-High; Complexity: 1-10; Priority: 1-100

# Ranking of Alternatives On deciding which framework will be used for the project. The team decided to compare each framework and rate them from one (1) to ten (10) on three (3) categories:

# Learning Curve: This is based on the understanding of each member of the team in relation to the framework that is being evaluated.

# Adaptability: This category was used to evaluate the ability to adjust each framework to our own needs and preferences.

# Resources: In this category the team did a research for each framework to determine which framework provides the best resourceful for our project.

## Table 4. Front-end Framework Learning Curve

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Front-end Framework | Learning Curve | Adaptability | Resources | Total |
| ReactJS v17.0.1 | 10 | 10 | 8 | **28** |
| AngularJS v1.8.2 | 8 | 8 | 6 | **22** |
| Django v3.0.12 | 2 | 4 | 8 | **14** |

## Table 5. Back-end Framework Learning Curve

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Back-end Framework | Learning Curve | Adaptability | Resource | Total |
| NodeJS v15.6.0 | 10 | 10 | 10 | **30** |
| Django v3.0.12 | 2 | 2 | 8 | **12** |
| PHP v8.0 | 2 | 4 | 4 | **10** |

## Table 6. Software Components

|  |  |
| --- | --- |
| Component | Usage |
| GitHub | As a team repository for the entire project. |
| Docker | As a container transfer of the project as an instance. |
| AWS (EC2) | As a virtual computer to run the server. |
| Apache | As a forwarder of the traffic of the IP. |

# Security

In order to protect the server and the webpage, each framework will search once every two (2) hours for new updates available. In case that an update was found, this will trigger an alert in the system that a new update was found. Once one of the developers get that alert, the team will proceed to make a research of the new update to evaluate if all the components the current framework has are compatible with the new update that was launched.

That research will conclude with a final discussion between the developers to conclude if the updates are suitable at this time for implementation on the project. If it is determined that the update is not suitable for the framework, the update will be rescheduled, until a new one comes out, or at least a patch. Then another research would take place and another discussion for no later than thirty (30) days.

# Software Architecture

* The whole architecture of the Application will be covered inside React.js. This is a JavaScript library that is used for building the user interface and it is the right choice for building the Covid-19 application. It was chosen because the capabilities for building an efficient user interface without removing the interactive part with the user. Components inside this library such as render will be very useful for taking the input data that the APIs bring and displayed in a graphic. Moreover, the use of states will help to develop different cases and times that the data will be presented when retrieving the data. Getting this components together plus the APIs will make an efficient construction of the code.

# Detailed Software Architecture

Provide more than one perspective of the Software Architecture being adopted, detailing not only Software Components and their interconnection through Software Connectors, but also scenarios for deployment of software modules or alternatives.

# Perspectives

* Layered Architecture: With this type of pattern, we will have the Presentation Layer first, which will be where the UI (User Interface) components will be held. The UI components will be used in only one page, which will be the Home Page. Then, the Presentation Layer will be communicating to the Logic Layer to receive the data to be displayed in the graphics. In the Logic Layer it will be determined how the data can be created and modified. The validation of everything in the API, will be tested constantly for response using the command line interface. Finally, in the Data Access Layer a simplified access to the API’s data will be created.
* Model-View-Controller Architecture: Another way of looking at the system, would be with this type of pattern, in which we will have the View, which will hold the UI components that will be used as Controllers through the user’s interactions. These Controllers will be communicating to the Model, letting it know what kind of data needs to be altered. Then, in the Model, it will be determined how the data will be created and changed for validating everything; this way we can communicate back to the View to alter the data for the user.

# Software Components

* Front-End: This is the phase of the software that will be presented to the user, where it will be able to interact with the website. The customization of the webpage will be in this phase along with the structure, as it will be presented to the user. Here the user will find a summary of statistics, all the graphs available and information about the graphs like where the data came from. Moreover, the user will be able to select different countries and see how the Covid-19 statics of the desired country are at the moment. Furthermore, the user will be able to change the desired view of the graphic between plot and bar graph .The user, for example, can select the graph to be displayed, interact with those graphs, and decide if the data is presented or not by clicking a button in the graph. Users will also be able to interact with a graph that can display the covid -19 cases, deaths and recovered people of all countries. As well the users can re. In which they can sort by any of the previous categories mentioned before .This phase will be constantly communicating with the back-end to receive all the information that it needs to show. In the development part, the front-end will oversee handling some important tasks. We will start by creating the React app to start coding the project, and tackling the header, container, and footer components. After that, we will make three components, the summary, references and report components. Then moving to work on the colorblind button component and the different graphs like the bar and linear graph, that will be available in each one of the plots presented for the user.
* Back-End: In this phase all the logical functions (calculations and modifications of data) and other connections of the webpage will be coded, which only the developers will have access to. This phase will contain all the connections the webpage will need such as server connection and user connection. This is also the only phase that will be communicating with the other phases at the same time since it will need to pass the data it receives from the APIs to the front-end. In the development part, we will start by setting up the NodeJS to start working in the rest of the back-end and configuring with the different APIs required for the information that will be presented. This will be possible by working on API methods, and how to filter the data received from the APIs. Finally, we will be setting up a function to update the data from the APIs in a daily basis so that we do not have to do it manually, making the page more self-sufficient.

# Software Connectors

* GitHub: This platform will be used as a repository to save the code and implement/update the main project. This repository will be keeping a copy of all the versions for future reference so nothing would be really lost. Every member will have access to the project, but it would be required to never work on the master branch. To implement changes into the code, a branch will have to be made to test and run that piece of code. When that branch is tested and confirmed that it truly runs, then it will be pushed to the master branch to apply the changes.