ISIMA - Campus Universitaire des Cezeaux
1, Rue de la Chebarde
TSA 60125, CS 60026
63178 Aubière CEDEX

CERN
Route de Meyrin 385
Meyrin, Switzerland

Engineer Report

3rd Year Internship

LHCb Performance and Regression Development

Presented by: **Amine Ben hammou**

CERN Supervisor:
Benjamin Couturier

Internship duration:
5 months

ISIMA Supervisor:
MESNARD Emmanuel

Date of presentation:
August 28th 2015

Acknowledgments

Abstract

Contents Table

Acknowledgments i

Abstract ii

Contents Table iii

Abbreviations Table v

Introduction 1

1. Presentation of CERN and LHCb 2

1.1. Presentation of CERN 2

1.1.1. Scientific achievements 2

1.2. Presentation of LHCb 3

2. The LHCbPR project 3

2.1. What is LHCbPR 3

2.2. Why LHCbPR 3

2.3. Users of LHCbPR 4

2.4. The old LHCbPR application 4

3. New LHCbPR application 5

3.1. Architecture 5

3.2. Data Storage Layer: Database 5

3.3. Application Logic Layer: REST API 6

3.4. Frontend 7

4. Frontend Development 8

4.1. Used Tools 8

4.1.1. AngularJS 8

4.1.2. jQuery 8

4.1.3. npm 8

4.1.4. Bower 9

4.1.5. Gulp 9

4.1.6. jade 9

4.1.7. less 9

4.2. The starting template 10

4.2.1. Setup steps 10

4.2.2. Files structure 10

4.2.3. Included Libraries 10

4.2.4. Lazy Loading 10

4.2.5. Predefined Directives 10

4.2.6. Gulp tasks 10

4.3. Modifications made to the template 11

4.3.1. LHCbPR service 11

4.3.2. Separating modules 11

4.3.3. Custom directives 11

4.3.3.1. Search Jobs 11

4.3.3.2. Select Jobs 11

4.3.4. Responsive Charts and DataTables 11

4.3.5. Loading Icon 11

5. Analysis Modules **12**

5.1. Jobs Module 12

5.2. Trends Module 12

5.3. Histograms Module 12

Conclusion **13**

Abbreviations Table

3TP	Three Tiered Programming
Ajax	Asynchronous JavaScript and XML
CERN	The European Organization for Nuclear Research
CSS	Cascading Style Sheets
DOM	Document Object Model
HTML	HyperText Markup Language
ISIMA	Institut Supérieur d'Informatique, de Modélisation et de leurs Applications
JSON	JavaScript Object Notation
LHCb	Large Hadron Collider beauty
LHCb PR	LHCb Performance and Regression
npm	Node Package Manager
REST	Representational State Transfer

Introduction

As part of my third year at ISIMA, I did my final internship at CERN. More specifically, I worked on the LHCb experiment and participated on the development of LHCb Performance and Regression framework.

The goal of the LHCb PR project is to provide developers with a profiling framework helping them to evaluate their recent changes by runing analysis modules and comparing results. The results can be used as indicator for worsen algorithms due to run-time or memory consumption and thus as knowledge basis for warning mechanism or for project leaders to track the development process from the optimization point of view.

During my internship, I worked on the development of the new version of LHCbPR. My task was to upgrade the web application using AngularJs framework. The new version should provide the same functionalities as the old one and provide a more easy to use user interface. The application should also give developers the ability to add new modules easily and without having to change the code of the core application.

This report will present the different task I have done during my five months internship and the results I obtained. After presenting CERN and LHCb experiment, I will explain the idea of the LCHbPR project and describe the old version of its web application. Then I will present the different parts of the new version. After this I will explain in details how the new frontend web application was made. Finaly I will present the analysis modules made in the new version.

1. Presentation of CERN and LHCb

1.1. Presentation of CERN

The European Organization for Nuclear Research (French: Organisation européenne pour la recherche nucléaire), known as CERN (derived from the name "Conseil Européen pour la Recherche Nucléaire") is a European research organization that operates the largest particle physics laboratory in the world. Established in 1954, the organization is based in a northwest suburb of Geneva on the Franco–Swiss border.

The term CERN is also used to refer to the laboratory, which in 2013 had 2,513 staff members, and hosted some 12,313 fellows, associates, apprentices as well as visiting scientists and engineers representing 608 universities and research facilities.

CERN's main function is to provide the particle accelerators and other infrastructure needed for high-energy physics research – as a result, numerous experiments have been constructed at CERN as a result of international collaborations.

CERN is also the birthplace of the World Wide Web. The main site at Meyrin has a large computer facility containing powerful data processing facilities, primarily for experimental-data analysis; because of the need to make these facilities available to researchers elsewhere, it has historically been a major wide area networking hub.

1.1.1. Scientific achievements

Several important achievements in particle physics have been made through experiments at CERN. They include:

1973: The discovery of neutral currents in the Gargamelle bubble chamber

1983: The discovery of W and Z bosons in the UA1 and UA2 experiments

1989: The determination of the number of light neutrino families at the Large Electron–Positron Collider (LEP) operating on the Z boson peak

1995: The first creation of antihydrogen atoms in the PS210 experiment

1999: The discovery of direct CP violation in the NA48 experiment

2010: The isolation of 38 atoms of antihydrogen

2011: Maintaining antihydrogen for over 15 minutes

2012: A boson with mass around 125 GeV/c2 consistent with long-sought Higgs boson

1.2. Presentation of LHCb

The LHCb (standing for "Large Hadron Collider beauty") experiment is one of seven particle physics detector experiments collecting data at the Large Hadron Collider accelerator at CERN. LHCb is a specialized b-physics experiment, that is measuring the parameters of CP violation in the interactions of b-hadrons (heavy particles containing a bottom quark). Such studies can help to explain the Matter-Antimatter asymmetry of the Universe. The detector is also able to perform measurements of production cross sections and electroweak physics in the forward region. Approximately 840 people from 60 scientific institutes, representing 16 countries, form the collaboration who built and operate the detector. As of 2014, the spokesperson for the collaboration is Guy Wilkinson. The experiment is located at point 8 on the LHC tunnel close to Ferney-Voltaire, France just over the border from Geneva.

2. The LHCbPR project

2.1. What is LHCbPR

LHCbPR is a service designed to record important measurements about releases of the LHCb applications. These applications receive input in the form of configuration files and produce, as an output, various information. LHCbPR is not intended to actually run the jobs, but instead to manage and track the bulk of information produced by storing them into the database. The LHCbPR is a framework that allows LHCb software developer to push information for a run of their code(job charactetistics, results, performance measures, files) to a central database. Developers are also able perform analysis of the data across versions, runing options and plateforms.

2.2. Why LHCbPR

In the past, in order for a user working group to run a job, they had to execute a job for a specific application -using a configuration file-, gather the results and then run their own analysis on the produced sample. Finally, when such a process was complete, they needed to find a way to publish their results. Often ending up with serving static documents such as HTML, CSV or e-mail.

LHCbPR was conceived as a tool to reliably organize the process of configuring and monitoring a job execution. The framework, solves the problem of gathering the results of a job execution, by providing the user a complete script that, when executed, can produce the desired output, according to a configuration file. This script, also includes a list of handlers that collect the defined aspects of the job results and push them to the database in a uniformed way.

By collecting the results in such an organized manner, it provides the solution to the second problem, mentioned above; the running of the analysis.

Since, all the data are stored in the database, the framework provides an abstract and easy way to deploy algorithms and functions which perform analysis on the saved data.

Finally, LHCbPR handles the presentation of the collected data by providing a set of templates for the creation of web pages specific to each analysis and customized to the preferences of each user.

2.3. Users of LHCbPR

The LHCbPR users can be divided into three categories: Administrators, Application developers and End users.

The administrators group is responsible for the integrity of the collected data and the efficiency of the service. They maintain and support the system, making sure the application is functional and available to the end users.

Application developers are the users who actively design and develop modules for the framework, thus extending the functionality of the application.

Finally, the end users are the main body of the users of the service. They put the tools provided by the application developers to practical use and their job results are populating the database.

2.4. The old LHCbPR application

As my task is to upgrade the LHCbPR web application. I first started by reviewing the old version.

The LHCbPR follows the architectural model of three tiered programming. 3TP provides a way to theoretically categorize the components of an application, providing abstract modularity. In essence the 3TP model is a client-server architecture which is composed of three layers. Namely, the presentation tier, the functional process logic tier and the data storage tier. These tiers communicate with one another in a strictly defined way, which allows the developer to independently maintain each tier. In most practical applications of the model the presentation layer includes all forms of user interaction with the service. The process logic tier encompasses the whole of the application functionality, while the data storage tier handles the flow of the information from and to the data source (in most cases a database).

The Django framework was used to create the LHCbPR web application and the 3TP model was applied as follows:

Presentation Layer: A set of web pages using the jQuery library.

Application Logic Layer: A set of python modules made using the Django framework.

Data Storage Layer: An Oracle database storing all the applications and jobs informations and results.

3. New LHCbPR application

Instead of trying to improve the old version of the LHCbPR application, we started the new version from scratch to be able to use the new technologies of web development which was not used in the old version like Gulp task runner and the AngularJS framework.

3.1. Architecture

The new version follows the 3TP model too, but in a more modern way. In particular, the presentation layer and the application logic layer are no longer in the same application. They are now two totaly separated applications communicating using a REST API.

3.2. Data Storage Layer: Database

We kept the same database as the old version but made some little changes to it. The new database entity relation schema is presented on figure in the next page. The main entities are the following:

Application: an application is described by a name and has many versions

Application Version: a version belongs to an application. It can be a nightly version or not and can have multiple slots.

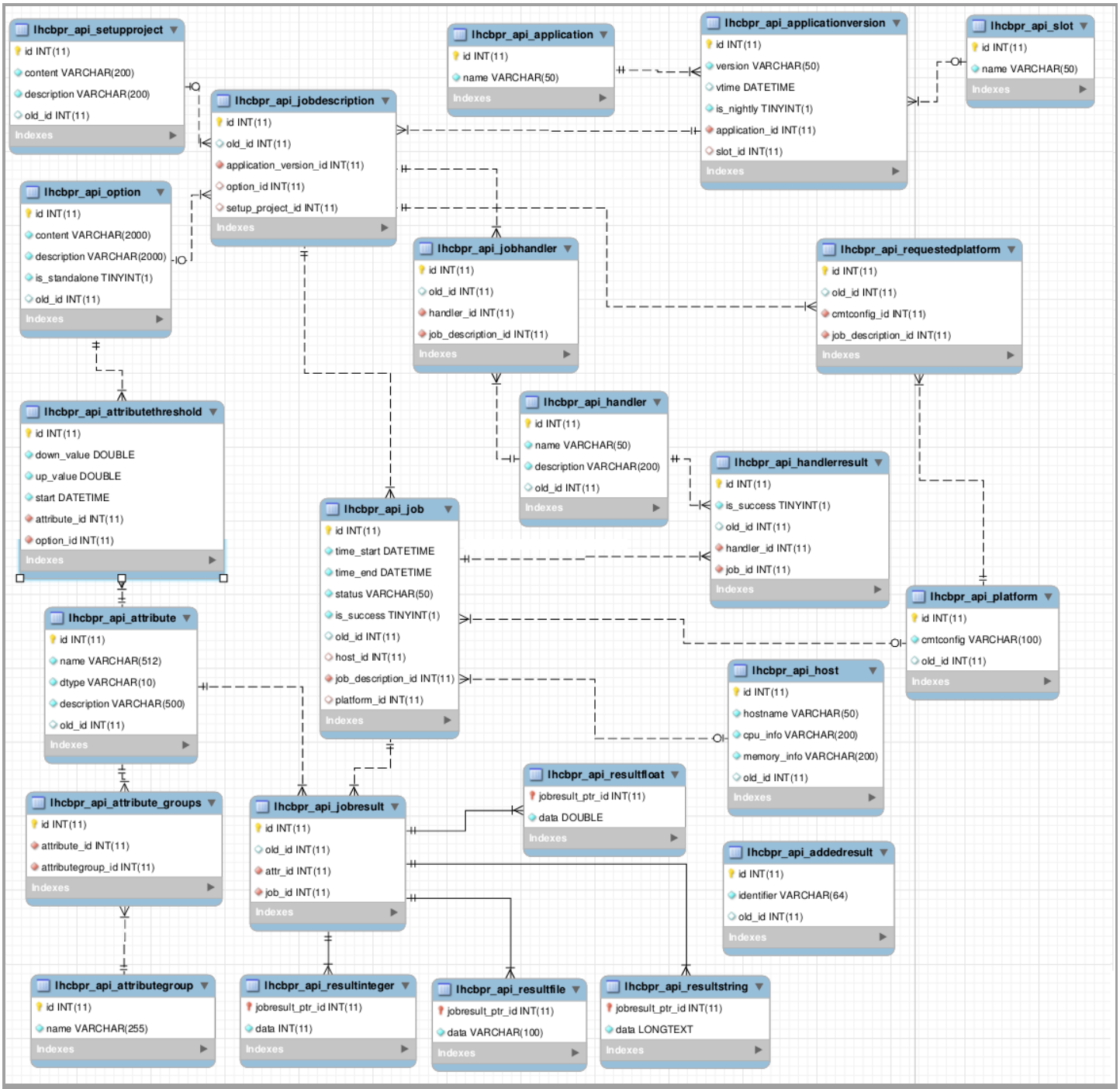
Job Description: This entity describes a group of jobs by specifying the application version, the project setup, the runing option, the plateforms in which jobs will be executed and the handlers that will collect results.

Handler: a handler has a name and a description. It is linked to multiple job descriptions via the "job handler" pivot table. It is also linked to many jobs via the "handler result" pivot table.

Option: describes the command-line options to pass when runing the jobs. I has a many to many relation with the attribute entity.

Attribute: an attribute can belong to multiple attribute groups. It also has a many to many relation with the job entity via the jobresult. Job results can have different types (integer, float, file, string).

Job: This is the main entity that describes a job. A job has a start time, an end time, and a status. It belongs to a job description and is run in a specifique plateform and host. Their results can be collected by multiple handlers and it can have multiple results. Every job result concerns an attribute.



Database Entity Relation Schema

3.3. Application Logic Layer: REST API

This layer is made using the **Django REST framework** which is a powerful and flexible toolkit that makes it easy to build Web APIs using Python language. Making the application logic layer as an API is the best choice to keep layers totaly independent. This way we can build multiple frontend applications (one for the web, an other for mobiles for example) based on the same API.

REST stands for **Representational State Transfer**. (It is sometimes spelled "ReST") It relies on a stateless, client-server, cacheable communications protocol -- and in virtually all cases, the HTTP protocol is used. REST is an architecture style for designing networked applications.

In our case, the frontend application send requests to the API and the API responds without the requested data in JSON (JavaScript Object Notation) format which 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 which make it easy to parse it as our frontend is written using JavaScript.

3.4. Frontend

The frontend application is made using the AngularJS javascript framework which add many features to HTML to make a new generation of web applications. Instead of starting the web application from scratch, we have used a template called **Angle** which offers a collection of ready to use stylings and tools based on AngularJs and many javascript libraries. More detailled description of the Angle features is described in the next section.

4. Frontend Development

4.1. Used Tools

We have used the Angle template as a starting point for the development of the frontend application. This template includes AngularJS, jQuery and many other libraries. It offers also some usefull predefined Angular directives and services. In order to understand the structure of this template and be able to customize it, a good understanding of AngularJS architecture and terms is essential. The most important tools used to build the frontend application are the following:

4.1.1. AngularJS

AngularJS is an open-source web application framework maintained by Google and by a community of individual developers and corporations to address many of the challenges encountered in developing single-page applications. It aims to simplify both the development and the testing of such applications by providing a framework for client-side web application which is able to add new reusable features to HTML.

4.1.2. jQuery

The usage of AngularJS does not eliminate jQuery which is the most used javascript library in web applications today. It is designed to make it easier to navigate a document, select DOM (Document Object Model) elements, create animations, handle events, and develop Ajax (Asynchronous JavaScript and XML). jQuery also provides capabilities for developers to create plug-ins on top of the JavaScript library. This enables developers to create abstractions for low-level interaction and animation, advanced effects and high-level, theme-able widgets.

4.1.3. npm

npm is package manager for JavaScript, and is the default one for NodeJS packages. NodeJS gives the possibility to run the javascript on the server side making it possible to write javascript files and run them from the terminal.

After the release of NodeJS on 2009, developers started making reusable node modules and the **node package manager** (npm) was created on 2011 to make it easy to share and reuse node modules. But now it is used to manage all types of javascript modules and packages.

npm is used in our application to manage packages such as Bower and Gulp.

4.1.4. Bower

Bower is a package manager for frontend components such as CSS and javascript libraries. It is installed using npm as it is a node module. Bower does not replace npm. npm manages server side packages will bower manages frontend packages.

4.1.5. Gulp

Gulp is a javascript task runner or build tool. It can run several tasks automatically to improve the developer workflow. Tasks are written in javascript and can do various things from concatenating multiple files into one to compiling a less code (the less language is presented below) and compressing the resulting CSS code and storing it in a destination file.

Gulp uses Steams (the NodeJS objects representing a buffer of data) during the task operations. So there is no need to store intermediate results in temporary files. It also runs the tasks in parallel which is more efficient. Dependencies between tasks can be declared to force a task not to start before the ending of an other one. Gulp can also watch files and run corresponding tasks when a file is changed.

4.1.6. jade

Jade is a templating language which produces HTML code from a less verbose syntax. The Jade syntax is more easy to write, read and maintain. More than that, the Jade language has many features which make the code dynamic and reusable.

All the HTML files in our frontend application are written using Jade and compiled by Gulp.

4.1.7. less

Less is a CSS pre-processor, meaning that it extends the CSS language, adding features that allow variables, mixins, functions and many other techniques that allow the developer to make CSS that is more maintainable, themable and extendable.

There are also many 3rd party tools that help to compile less files and watch for changes. Gulp is one of these tools.

4.2. The starting template

4.2.1. Setup steps

4.2.2. Files structure

4.2.3. Included Libraries

4.2.4. Lazy Loading

4.2.5. Predefined Directives

4.2.6. Gulp tasks

4.3. Modifications made to the template

4.3.1. LHCbPR service

4.3.2. Separating modules

See the modules development guide appendix for the modules development documentation.

4.3.3. Custom directives

4.3.3.1. Search Jobs

4.3.3.2. Select Jobs

4.3.4. Responsive Charts and DataTables

4.3.5. Loading Icon

5. Analysis Modules

5.1. Jobs Module

5.2. Trends Module

5.3. Histograms Module

Conclusion