# Google Summer of Code 2019 Proposal

## CERN-HSF - DIRAC

Pujan Mehta

## 1. Basic Details

| | |
|---|---|
| **Full Name** | Pujan Rajesh Mehta |
| **Institute** | A 3rd Year B.E. Computer Engineering student at D.J. Sanghvi College of Engineering, Mumbai. |
| **Email** | pujanrmehta@gmail.com |
| **Phone** | +91-9930437371 |
| **GitHub** | github.com/pujanm |
| **LinkedIn** | linkedin.com/in/pujanm/ |
| **Location & Time Zone** | Mumbai, India.<br>Indian Standard Time (GMT +0530) |
| **Address** | 2/B Manisha Building, Bhardawadi Road, opp. Oscar Diamond Building, Bhardawadi Road, Andheri West, Mumbai - 400058. |
| **Resume** | Pujan Mehta Resume |

## 2. About Me

I am Pujan Mehta, a 3rd-year Computer Engineering student at D. J. Sanghvi College of Engineering. I have fluency in programming languages like Python, Java, and C. I am also a full-stack Web developer.

Most of the projects that I have worked on until now are based on Python and I also started my programming career in 2016 with Python with which I realized the true power of Python and also found that many of the big organizations around the world use it heavily. Having knowledge in Software Development helped me get an internship at a

Silicon-Valley based startup named Falkonry where I used Python, Java, Node.js, and Git and learned Docker, Kubernetes, and ELK Stack and also got to work on AWS. Recently, I also completed MOOCs on Machine Learning and Deep Learning during which I used tools like NumPy, Pandas, Matplotlib, Scikit-Learn, and PyTorch and also got an internship at VoyaGenius Labs. My background knowledge comprises of Algorithms, Operating Systems, and Machine Learning.

And apart from these, I am also a student mentor at DJ Unicode a college level open-source organization where I guide juniors on their back-end development projects and I also like Competitive Programming and I take part in competitions held on Codechef and HackerRank.

**Why GSOC with CERN-HSF?**

I was always fascinated by the work going at CERN and the heavy use of computing along with physics. I also wanted to be associated with CERN in some or the other way and I found GSOC the best way to achieve it and here I will also get a chance to enhance my practical skills as there are varied types of tasks each one of them requiring a set of different concepts.

**How much time will I be able to commit to the project?**

I will be able to contribute 6-8 hours per day to the project. I will be able to work from 9.00 AM (Central European Timezone) till 5.00 PM (Central European Timezone) and I will be working full-time only for GSOC throughout the summer, as I have no other work commitments.

## 3. Synopsis

**DIRAC** is a highly-scalable software used for accessing distributed resources from various distributed systems. **DIRAC's** main contributor is **LHCb** and also its initiator. **LHCb** uses the different type of computing technologies in order to distribute and process the collected physics data and **DIRAC** is one the software which is used because of its scalability and the level of orchestration and monitoring it provides for the distributed resources which is the main requirement of the **LHCb** collaboration.
Further, my task will be to upgrade **DIRAC's** monitoring system further by assuring high-scalability as when the **LHCb** gets upgraded there can be an unpredictable type of data which needs to be molded easily by **DIRAC** for which we will use **ElasticSearch** which is one of the widely used NoSQL technology.

# 4. Project Goals

## Objectives

- To setup **DIRAC** and **WebAppDIRAC** locally.
- To fix bugs in the Monitoring plots.
- To document the working of the **Framework/Monitoring** service.
- To implement various use cases of **gMonitor** using **MonitoringReporter**.
- To create new monitoring plotter to visualize data based on ES.
- To write unit, performance, integration and system tests.
- To write Python 3 compatible code.
- To document all the completed work.

## Tasks

### 1. Setup details:

To follow the developer guide of DIRAC and complete the local setup either Docker or UNIX based.
Then after installation to run components like Service, Agent, etc. To write some basic "Hello World" service and to connect it with an agent.
To write unit tests for service and agent and too make them **pylint** verifiable i.e. to use **autopep8** to format the code properly.
To install MySQL locally and run the "Framework/Monitoring" service and to check its working along with the "Hello World" service too i.e. to ensure that it is invoked.
To install ElasticSearch locally and then to run and setup the **Monitoring/Monitoring** and **Framework/SystemAdministrator** services.
Then to setup **WebAppDIRAC** to verify various monitoring plots locally.

### 2. Implementation flow:

I will divide the above objectives into a set of tasks for better implementation flow.

**Task 1:**

After completing all the required setup I will compare the monitoring plots and based on their output I will solve the possible bugs.
Then, I will start writing the documentation on the working of the **Framework/ComponentMonitoring** service which can be found here. I will make a note on working of each method and how the information is currently stored in the DB.

**Task 2:**

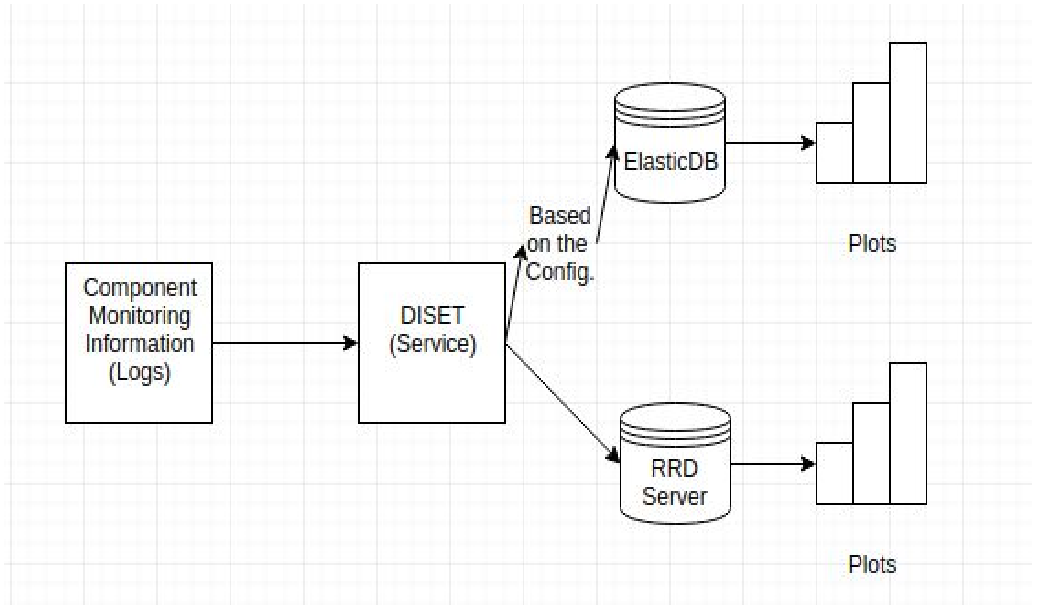
Then I will have to define an ElasticSearch type using **MonitoringReporter**

inside **Core/DISET/private/Service.py** in order to send the monitoring info as it is currently done by the **gMonitor** object which sends info to the **RRD** ( Round-Robin Database tool ).
server using **gMonitoringFlusher** i.e. wherever we use the **registerActivity** and **addMark** functions we will send that data to the ElasticDB in NoSQL form using the **MonitoringReporter**.
Further, as we need to keep the ElasticDB optional we will add a flag which will be configurable inside the dirac.cfg file and based on its value which we can get using the DIRAC's gConfig object we will decide to send data to ES backend or not.
Then I will like to document all the development and based on the requirement and discussion with mentors I will write unit tests accordingly.

Then further once this data is being added to the **ElasticDB**, I will have to create a new monitoring plotter to visualize data based on the **ElasticDB** which will be created inside **DIRAC/Monitoring/private/Plotters**.
Then to ensure proper functionality of the plotter I will have to write a regression test based on this given model.

Then based on the outputs of the plots i.e RRD vs ElasticSearch I will note down all the differences, advantages and drawbacks and note them down in a document.

The workflow will be as shown below in the diagram:



As our main task was to improve the scalability of the system w.r.t. the later one I will have to create some performance and load tests in python using

Multi-Mechanize which will help me to generate load (queries ) which can be further passed on using the DIRAC’s REST interface. Here I will have to document all the performance difference’s between RRD and ElasticSearch like the improvements, drawbacks, etc.

**Task 3:**

Then in this task, I will have to start implementation of this use case by inputs from my mentors where we have to send the current logs to the Elastic backend which are currently being reported nowhere.
Then I will have to migrate **Core/base/AgentModule.py** such that we can send logs to the Elastic backend wherever we use the **gMonitor** object using the **MonitoringReporter**.
After, that the next task will be report the cycle duration(duration of time taken by the agent in a single loop) and send them to the Elastic backend.
Similarly, as to be done in **Task 1** we have to develop use cases of **gMonitor** object in ElasticSearch. I will have to develop the similar use case for **RequestManagementSystem** and **DataManagementSystem/Agent/RequestOperations/**. For this, I will have to document the proper places in the code base where exactly I will have to send monitoring info to ElasticSearch and note them down.
Further, I will think over an Elastic type which covers both of these use cases and then send info to the Elastic backend accordingly.
Once, this implementation is completed I will have to work on the required unit tests and will have to check for any errors and issues in the code.
Then after completing all these three tasks, I will have to work on integration and system testing for checking the performance of the overall system after making changes to the code base to ensure everything else works fine within DIRAC.

## 5. Timeline

| Duration | Task |
|---|---|
| April 9 | **Deadline for Student proposals.** |
| April 9 - May 6 | Reading the **DIRAC** documentation and try to experiment more with it. |
| May 6 - May 27 | **Official Community Bonding Period:** Trying to know more about **DIRAC** and learning about its usage in the **LHCb.** Also, to attend **DIRAC** workshops if conducted.<br><br>**Begin with the Setup:**<br>Complete the setup as described in the **Setup details** in the proposal. |
| May 27 - June 10 | **Official Coding Period Starts:**<br>● Begin with the implementation of the **Task 1.**<br>● To fix bugs in the plots and to complete the documentation as described in the **implementation details** section.<br>● To send the **1st Pull Request** on the official **DIRAC** code base.<br>● This will complete the **Task 1.**<br>● Begin with implementation of **Task 2.**<br>● Start making notes on the current use cases of **gMonitor** object as described in **Task 2** implementation details. |
| June 10 - June 24 | ● Complete all the notes and complete coding the parts where need to send logs to the Elastic backend in **Core/DISET/private/Service.py** as described in the implementation details**.**<br>● Make the Elastic backend as configurable i.e. user can decide to send logs to Elastic backend or not. |

| | |
|---|---|
| | ● To start documenting all the work till now and also complete unit tests wherever required for the **Phase 1 Evaluations**.<br>● To send the **2nd Pull Request** on the official **DIRAC** code base. |
| June 24 - June 28 | **Official Phase 1 Evaluations.**<br>To complete all the documentation work and summarize about the 2 **PRs** sent on the official **DIRAC** code base until now. |
| June 28 - July 12 | ● To continue further working on the **Task 2.**<br>● Further, to start working on building the plotter as mentioned in the implementation details from the work done till now.<br>● Complete coding the Regression tests for the new plotter.<br>● Make notes and documentation on the comparison between RRD and ES plots.<br>● To send the **3rd Pull Request** on the official **DIRAC** code base. |
| July 12 - July 22 | ● Start writing the performance tests for the updates system using multi-mechanize.<br>● Make documentation on the performance between the RRD and ES systems and mention about the queries performed.<br>● To send the **4th Pull Request** on the official **DIRAC** code base.<br>● This will complete the **Task 2.** |
| July 22 - July 26 | **Official Phase 2 Evaluations.**<br>● Summarize all the work done till now and the details of the 2 **PRs** sent on the official **DIRAC** code base. |
| July 26 - August 9 | ● Begin with the implementation of the **Task 3.**<br>● To complete the use case mentioned in the implementation.<br>● To start migrating **Core/Base/AgentModule** to |

| | |
|---|---|
| | **MonitoringReporter**.<br>● To make documentation first on which places to migrate to the Elastic backend.<br>● To complete the migration work.<br>● To also report cycle duration in accordance with the implementation details.<br>● To document all the work done on **Task 3** till now and also write unit tests wherever necessary.<br>● To send the **5th Pull Request** on the official **DIRAC** code base.<br>● To make notes on the use cases of **gMonitor** object in **RequestManagementSystem** and **DataManagementSystem/Agent/RequestOperations/.**<br>● To code this use case as described in **Task 3** inside the implementation details.<br>● To write documentation of work done and unit tests wherever necessary.<br>● To send the **6th Pull Request** on the official **DIRAC** code base.<br>● This will complete the **Task 3.** |
| August 9 - August 19 | ● Then too work on various integration and system tests with the inputs from my mentors to ensure proper working of the entire system.<br>● To document all the tests and send the **7th Pull Request** to the DIRAC code base.<br>● To keep some buffer time for any unexpected issues and start preparing for the final evaluations. |
| August 19 - August 26 | **Official Final Evaluations.**<br>● Summarize all the work done till now in the entire project and the details of the 3 **PRs** sent on the official **DIRAC** code base.<br>● All so prepare a final project report. |

## 6. Deliverables

- To provide working implementation of **DIRAC** component monitoring using ElasticSearch backend, completed with tests and documentation.
- To improve the overall system performance and scalability.
- To provide code compatible in Python 3.

### Future Goals

- To improve the overall Docker setup of the **DIRAC** system and to also extend support of Kubernetes to **DIRAC**.
- To continue working on related problems on the **DIRAC** GitHub repository and help the **DIRAC** collaborators to migrate their code base in Python 3.
- To further help in building **DIRAC**.