CERN
Summer Student Programme 2021
รายงานการเข้าร่วมโครงการนักศึกษาภาคฤดูร้อน

ระหว่างวันที่ 5 กรกฎาคม - 27 สิงหาคม 2564

นายชาญชนะ วิชา

ภาควิชาวิศวกรรมคอมพิวเตอร์

คณะวิศวศาสตร์ มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าธนบุรี

PREFACE

This report is the outcome of the CERN Summer Student Programme 2021, organized virtually by CERN starting from July 5 to August 27, 2021. The objectives of this report are to provide an introduction of my attended program, to express my experience throughout the program and to explain the technical detail of my project.

I highly hope that this report will give beneficial and inspirational information to readers. Though I tried my best to keep this report error free, I apologize if any error was found which was not deliberately made. Please do not hesitate to contact me if any questions arise.

Chanchana Wicha

ACKNOWLEDGEMENTS

The author is appreciatively aware of his royal grace in Her Royal Highness Princess Maha Chakri Sirindhorn bestowed the opportunity to be a representative of Thai students participating in the CERN Summer Student Programme 2021.

I would like to take this opportunity to give my special thanks to Dr. Unchalisa Taetragool, my advisor of the department of computer engineering, King Mongkut's University of Technology Thonburi, for giving me valuable suggestions and knowledge and always supporting me throughout the program.

Beside my university's advisor, I am extremely thankful to my CERN Summer Student Programme supervisors, Justinas Rumsevicius and Chayanit Asawatangtrakuldee who gave me a golden opportunity to do this project as well as gave me valuable comments and suggestions that led to the completeness of this project.

Also, this summer program could not be completed without Dr. Norraphat Srimanobhas and Mrs. Umaratchani Kaewbutta, who coordinate and provide precise information throughout the program.

I also would like to thank the Synchrotron Light Research Institute for holding an informative preparation training session which provided valuable knowledge. Lastly, I gratefully appreciate all the hard work of related coordinators and institutes that support this program.

TABLE OF CONTENTS

PREFACE	**I**
ACKNOWLEDGEMENTS	**II**
TABLE OF CONTENTS	**III**
TABLE OF FIGURES	**IV**
Chapter 1 Introduction	**1**
Chapter 2 Experiences	**2**
Chapter 3 Technical Report	**3**
3.1 Abstract	3
3.2 Introduction	3
3.3 Acknowledgements	3
3.4 Technologies	4
3.5 Methodology	4
3.6 Result	8
3.7 Conclusion	10

TABLE OF FIGURES

Figure 1: Use-case Diagram 5
Figure 2: System Architecture 6
Figure 3: ORM Usage Example 6
Figure 4: All Campaign Page User Interface 8
Figure 5: Report Editor User Interface 8
Figure 6: Report Comment Section User Interface 9
Figure 7: User Permission Management Page User Interface 9

Chapter 1
Introduction

CERN Summer Student Programme 2021 is a summer internship program that gives an opportunity to physics and computer engineering students who want to contribute their passion with one of the world's largest and most respected centers for scientific research. Due to the COVID-19 situation, the program is held virtually for this year. However, there are many interesting virtual activities and lectures packed throughout the program.

When I got an opportunity to participate in this year's program, which took place from July 5 to August 27, I was extremely excited and grateful. I was assigned as a part of the PdmV team to develop a new version of one of their software called ValDB.

ValDB is a website for tracking software packages that are deployed on the CMS system to ensure the correctness and quality of the software. However, the previous version has a problem with the overall user experience, which makes users frustrated when using this tool. So, my main objective is to implement the new version of this software that enhances the previous version’s features as well as introduce new essential features.

With my knowledge and experience of software engineering, I am confident in the completeness of this project. Together with modern software development technologies such as Python, TypeScript, and React, which I am very familiar with, it guarantees that this software is robust, and the codebase is highly maintainable.

Nevertheless, CI/CD eases the development and deployment process of this project. It provides an automation pipeline which reduces repetitive manual processes. It also consistently ensures software quality. This is the main key that accelerates the software delivery process to users and reduces complexity for developers.

Chapter 2
Experiences

Having this summer student program is one of the best opportunities and an impressive experience for me. It gave me a chance to work with one of the largest scientific research organizations in the world. Working with amazing people internationally such as my supervisors is a golden opportunity and experience that cannot easily be found elsewhere.

Throughout my program, I got so many valuable suggestions and insightful comments from my supervisors and my team members, which made me improve my technical skills. Working with an international organization helps me improve my soft skills as well. It opens my perspective, and it makes me learn about other cultures as well.

Because my assigned project is a small-scale project, I got an opportunity to explore and experiment with many interesting ideas, including different implementation approaches, interesting external software libraries, and implementing my own core libraries. So, I did not have to worry about trying new things. I think this is one of the best experiences I have had because it makes me think outside the box, and reminds me not to get stuck with the same kind of ideas.

Although this is my first virtual internship that I have not had before, I think the program was perfectly executed and very impressive. Their virtual activities and lecture sessions are flawlessly organized. Lastly, I highly recommend this program to everyone who wants to take a chance to become a part of CERN that may change our future.

Chapter 3
Technical Report

3.1 Abstract

ValDB is a validation database website. The previous version has a problem with the overall user experience. By developing a new version of ValDB, it can help improve user experience and introduce essential features to the system. Together with modern web development technologies such as Python, Flask, TypeScript and React, they ease the development process and make the codebase easily maintainable. A custom ORM takes advantage of native type hinting introduced in Python with simple syntax. CI/CD helps automate processes, ensure system quality, and quickly deliver new software changes to users.

3.2 Introduction

I got an opportunity to be selected as a CERN Summer Student Programme 2021 for 8 weeks. During the program, I was assigned as a part of the PmdV team with my supervisors: Justinas Rumsevicius and Chayanit Asawatangtrakuldee. My assignment was to work on developing a new ValDB system.

ValDB is a validation database website for tracking software packages' quality that are deployed on the CMS system. Currently, this system has some problems that affect the user experience, including a complicated user interface, missing information in report content, and a complicated user management system. The objectives of developing a new version of ValDB are to improve the overall user experience and implement essential features.

3.3 Acknowledgements

I would like to express my special thanks to my supervisors: Justinas Rumsevicius and Chayanit Asawatangtrakuldee, and the PdmV team members who gave me a golden opportunity to do this project as well as gave me valuable comments and suggestions which led to the completion of this project.

Furthermore, I would like to thank everyone who organized and coordinated CERN Summer Student Programme 2021. Although it was an online summer student program, it was an amazing and memorable experience for me, and everything was perfectly organized.

3.4 Technologies

Unlike the previous version of ValDB that used conventional technologies such as plain HTML, which made it hard to develop advanced features and maintain, the new version of ValDB uses modern technologies which are popular, powerful and maintainable. Technologies applied in this system can be divided into three main components: backend, frontend and database.

The backend is implemented with Python together with Flask to serve the requests from clients. The frontend is implemented with TypeScript and React. TypeScript is an enhanced version of JavaScript which supports static typing, so the frontend codebase is easy to maintain. For the database, MongoDB is used here. MongoDB is a document-based NoSQL database that supports dynamic schemas. It is a powerful database that can handle large amounts of various types of data.

3.5 Methodology

3.5.1 System Requirements

- Administrators can manage campaigns in the system.
- Administrators can manage users' permission and role.
- Administrators and validators can view campaigns and reports.
- Validators can only write reports based on their permission group.
- Validators can see their assigned reports based on their permission group.
- Anyone can comment on reports.
- A report editor needs to support various types of information, including tables, hyperlinks, and attachments.
- Emails need to be sent on certain events in the system, such as campaign or report modification, users' permission change, and new comments on reports.
- Data from the previous version needs to be migrated to the new version.

3.5.2 Use-case Diagram

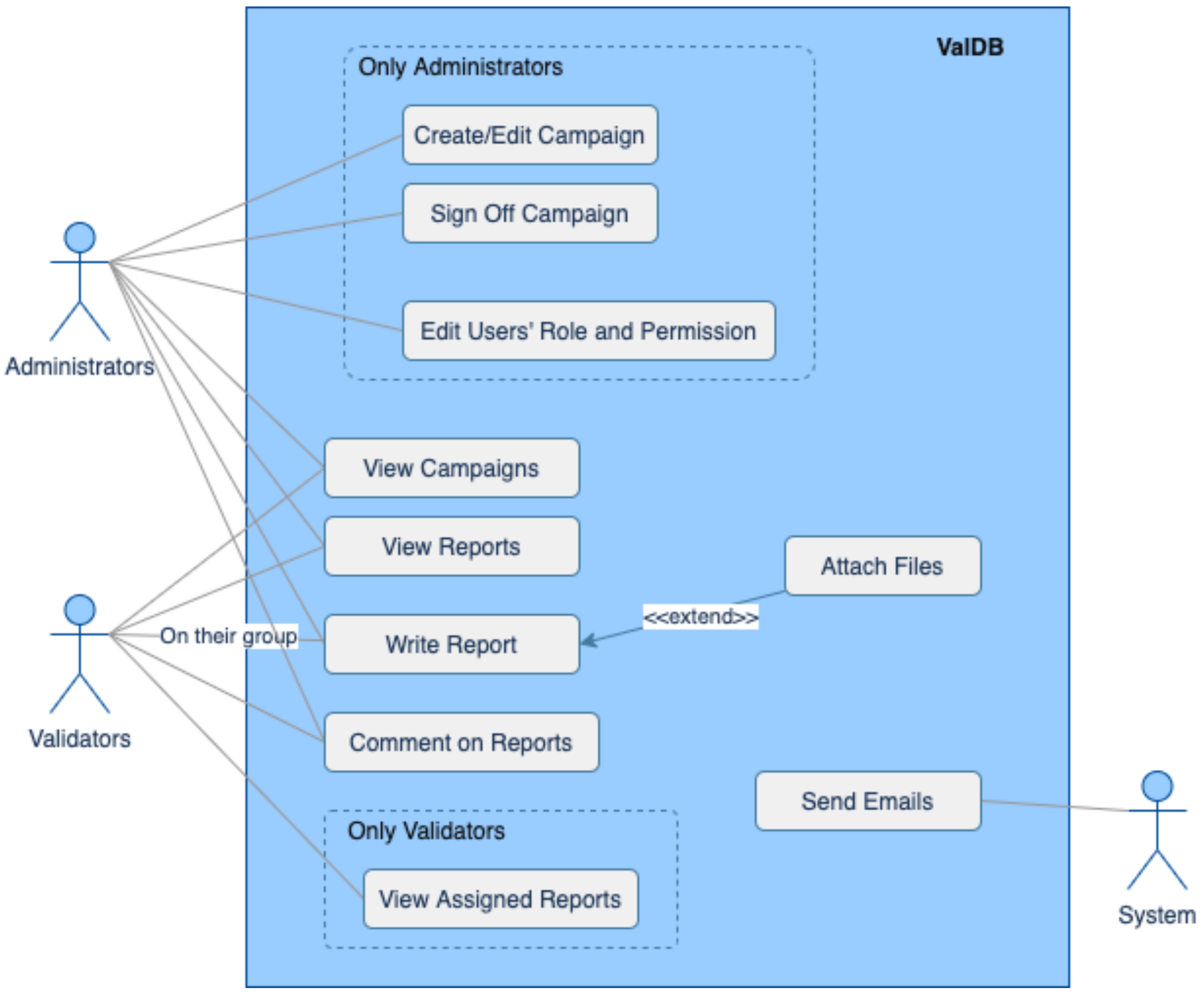


Figure 1: Use-case Diagram

Figure 1 shows the overall use-case of the new system. Users in the system can be divided into two roles: administrators and validators. Administrators are responsible for campaign and user management. They can create, edit and sign off campaigns and edit users' role and permission. On the other side, validators are responsible for writing reports based on their permission groups. They can add attachments to the reports. However, they can view campaigns and reports from another group, but they do not have permission to modify them. Any user can comment on any report in the system. Validators can also view their assigned report which they can have quick access to write reports. Lastly, the system can automatically send emails to users on certain events.

3.5.3 System Architecture

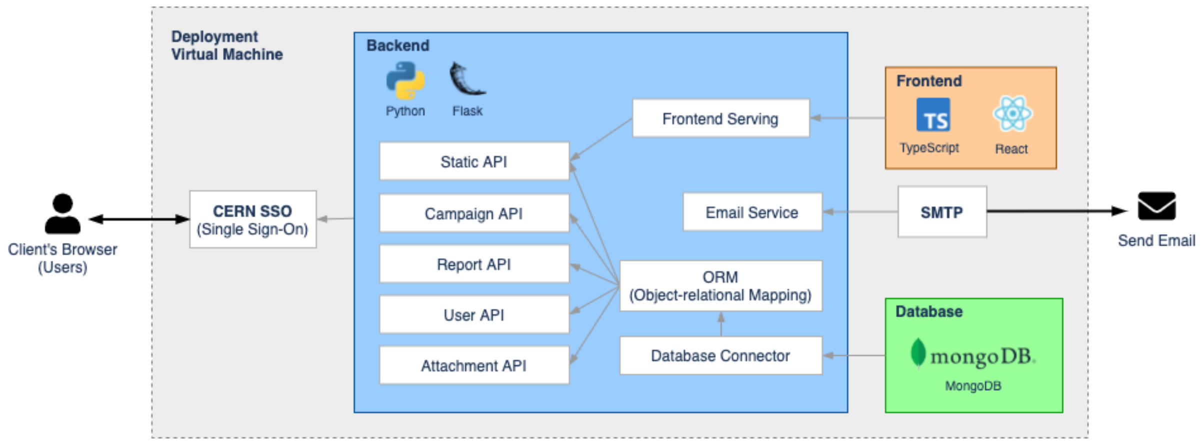
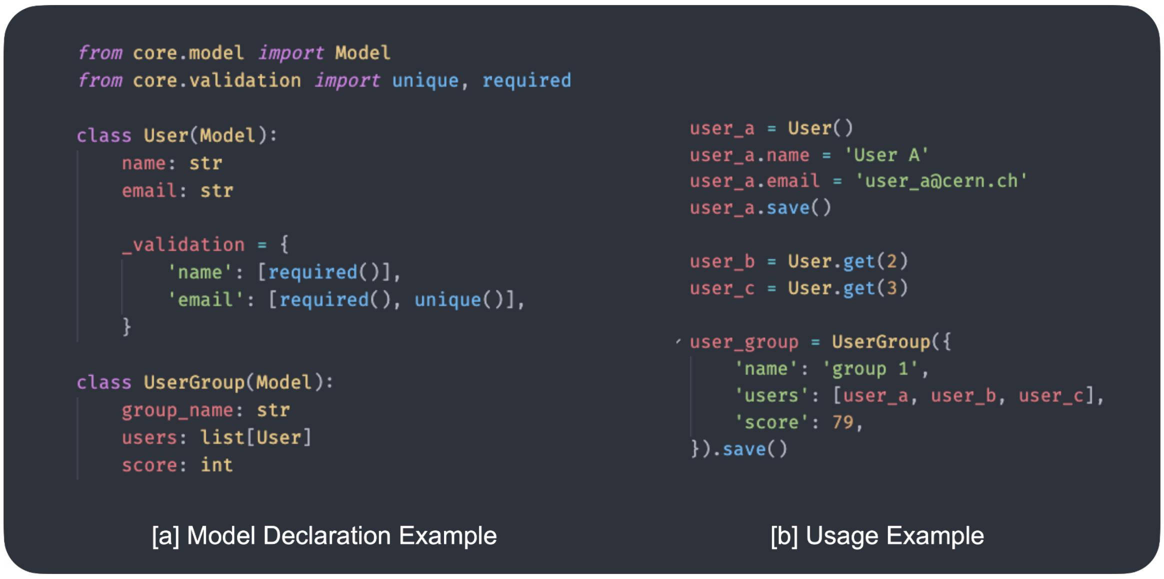


Figure 2: System Architecture

Figure 2 shows the architecture of the system. Every request sent from the client's browsers will be authenticated by the CERN SSO (single sign-on) Service before passing to the backend of the system. The backend is the main component in the system that handles all the logic. Data will be mapped into native Python objects using ORM (Object-relational Mapping) so that the backend can manipulate data in the database using its objects with a simple interface. ORM handles database operations via the database connector which connects to the MongoDB database. For the frontend component, it is built into a static site that can serve clients via a backend static API. For email service, it uses pre-configured SMTP on their deployment virtual machine to send email to users.

3.5.4 Custom ORM



```
from core.model import Model
from core.validation import unique, required

class User(Model):
    name: str
    email: str

    _validation = {
        'name': [required()],
        'email': [required(), unique()],
    }

class UserGroup(Model):
    group_name: str
    users: list[User]
    score: int
```

[a] Model Declaration Example

```
user_a = User()
user_a.name = 'User A'
user_a.email = 'user_a@cern.ch'
user_a.save()

user_b = User.get(2)
user_c = User.get(3)

user_group = UserGroup({
    'name': 'group 1',
    'users': [user_a, user_b, user_c],
    'score': 79,
}).save()
```

[b] Usage Example

Figure 3: ORM Usage Example

Because Python is a dynamic type programming language, dealing with data that does not declare specific types or schemas can be frustrating and hard to maintain. However, Python 3.9 introduces native generic type hinting. A custom ORM implemented in this system aims to take advantage of native type hinting to makes the codebase more maintainable but still keep the simple syntax style of Python.

Figure 3 shows the example usage of custom ORM in the system. For data model declaration, it supports native type annotation such as string, integer, floating point, boolean and enumeration. It supports data relationships such as one-to-many and many-to-one relationships. It provides a simple interface which resembles the native Python coding style without strict schema or complex declarations. For example, on Figure 3.a, the user group contains many users which can be declared by a list of user objects using native type hinting, and the ORM will automatically create a data relationship. Moreover, it supports field validation that validates each field by specified rules before committing a transaction in the database. The ORM provides all essential database operation interfaces such as create, update, query, and delete with simple usage.

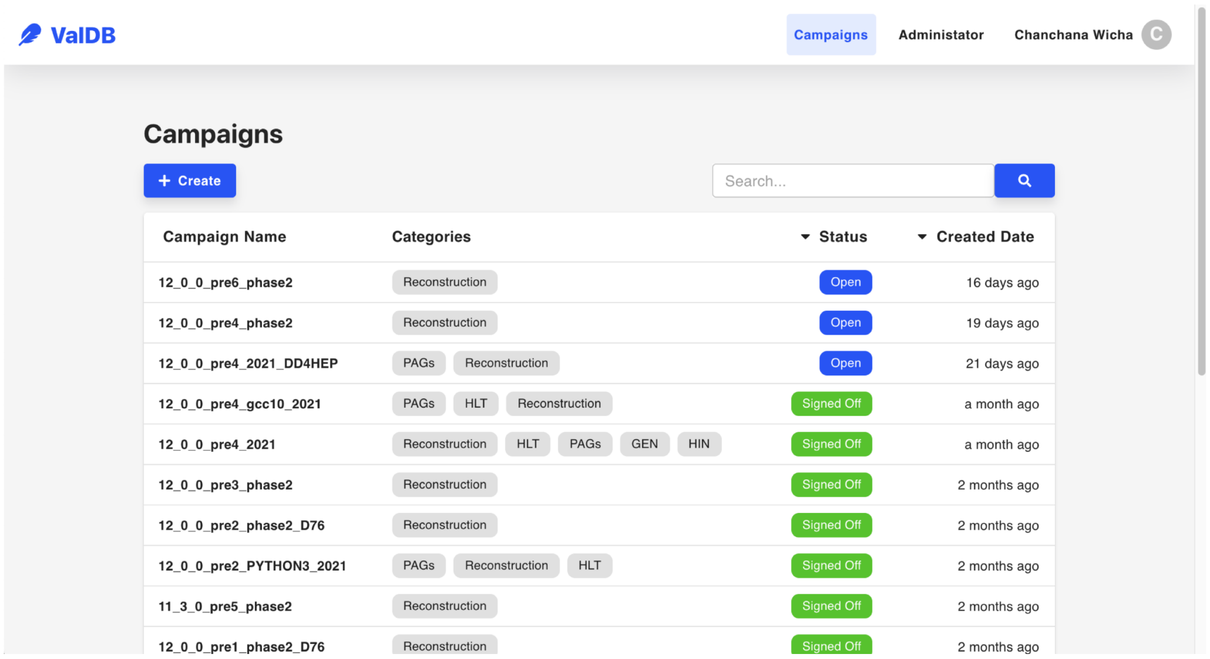
3.5.5 CI/CD

CI/CD (continuous integration and continuous delivery) helps developers integrate their features and deliver them easily. Since Github is used to store the source code of this project, Github Actions takes care of CI/CD operations in this project. The operations are separated into two parts: pull requests and deployment pipelines.

For the pull requests pipeline, the pipeline runs on every pull request. The main action for this pipeline is linting, which includes pylint for Python and eslint and tslint for TypeScript. Linting helps keep code clean and conform to coding style standard rules. Unit tests are implemented in the project to check the correct functionality and behavior of the system. However, at the current state, unit tests need to be run manually because the pipeline is not configured to run them yet.

For the deployment pipeline, this pipeline runs every day at midnight. It keeps track of the main branch of the project's codebase for any new modifications. If there are changes, it runs these actions: build and release. For build action, it builds the frontend source into a static site and removes the frontend source and unit test folders since they are not needed for production. For release action, it makes an artifact file that can be executed on a production virtual machine.

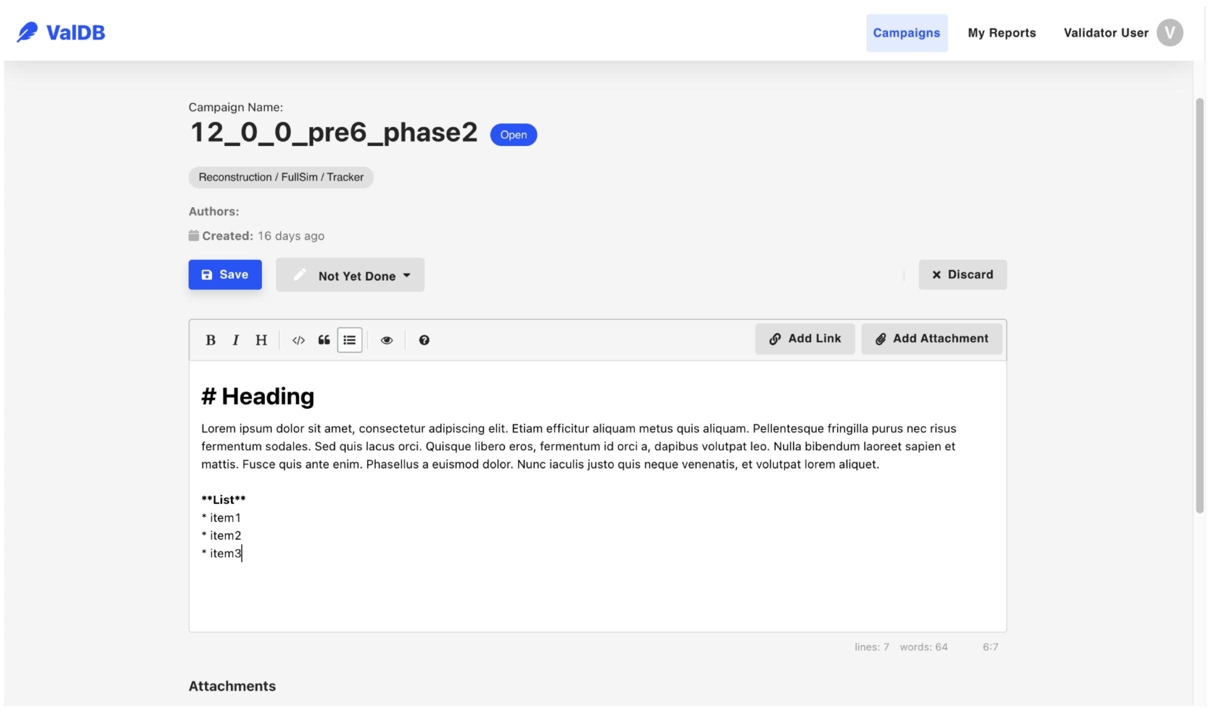
3.6 Result



Campaigns

Campaign Name	Categories	Status	Created Date
12_0_0_pre6_phase2	Reconstruction	Open	16 days ago
12_0_0_pre4_phase2	Reconstruction	Open	19 days ago
12_0_0_pre4_2021_DD4HEP	PAGs Reconstruction	Open	21 days ago
12_0_0_pre4_gcc10_2021	PAGs HLT Reconstruction	Signed Off	a month ago
12_0_0_pre4_2021	Reconstruction HLT PAGs GEN HIN	Signed Off	a month ago
12_0_0_pre3_phase2	Reconstruction	Signed Off	2 months ago
12_0_0_pre2_phase2_D76	Reconstruction	Signed Off	2 months ago
12_0_0_pre2_PYTHON3_2021	PAGs Reconstruction HLT	Signed Off	2 months ago
11_3_0_pre5_phase2	Reconstruction	Signed Off	2 months ago
12_0_0_pre1_phase2_D76	Reconstruction	Signed Off	2 months ago

Figure 4: All Campaign Page User Interface



Campaign Name:

12_0_0_pre6_phase2 Open

Reconstruction / FullSim / Tracker

Authors:

Created: 16 days ago

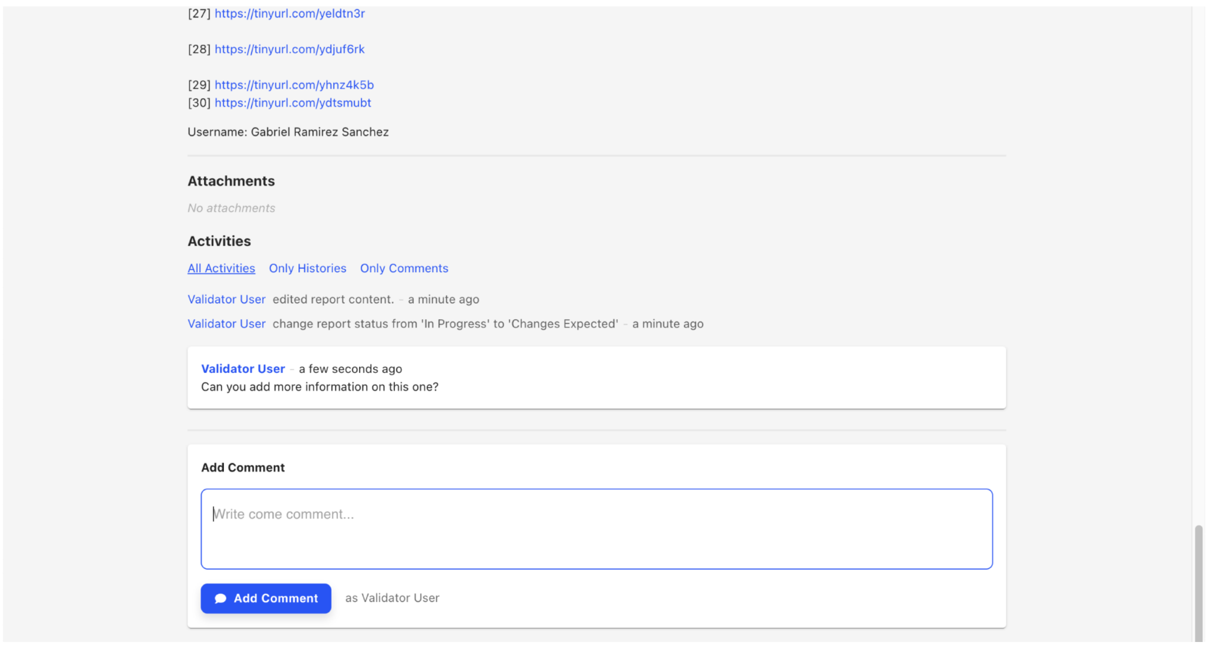
Heading

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Etiam efficitur aliquam metus quis aliquam. Pellentesque fringilla purus nec risus fermentum sodales. Sed quis lacus orci. Quisque libero eros, fermentum id orci a, dapibus volutpat leo. Nulla bibendum laoreet sapien et mattis. Fusce quis ante enim. Phasellus a euismod dolor. Nunc iaculis justo quis neque venenatis, et volutpat lorem aliquet.

```
**List**
* item1
* item2
* item3
```

Attachments

Figure 5: Report Editor User Interface


[27] https://tinyurl.com/yeldtn3r

[28] https://tinyurl.com/ydjuf6rk

[29] https://tinyurl.com/yhnz4k5b
[30] https://tinyurl.com/ydtsmubt

Username: Gabriel Ramirez Sanchez

Attachments

No attachments

Activities

All Activities Only Histories Only Comments

Validator User edited report content. - a minute ago

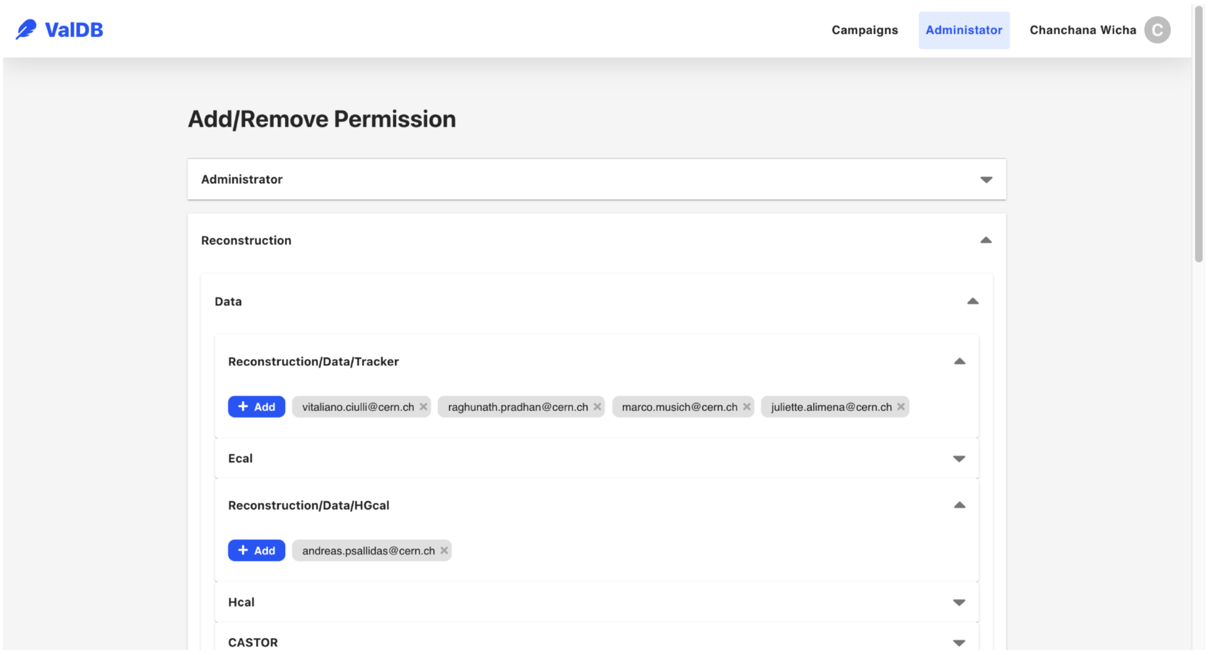
Validator User change report status from 'In Progress' to 'Changes Expected' - a minute ago

Validator User - a few seconds ago
Can you add more information on this one?

Add Comment

Write come comment...

Figure 6: Report Comment Section User Interface



Add/Remove Permission

Administrator

Reconstruction

Data

Reconstruction/Data/Tracker

vitaliano.ciulli@cern.ch raghunath.pradhan@cern.ch marco.musich@cern.ch juliette.alimena@cern.ch

Ecal

Reconstruction/Data/HGcal

andreas.psallidas@cern.ch

Hcal

CASTOR

Figure 7: User Permission Management Page User Interface

All of the components are developed and deployed in a virtual machine. The following figures: Figure 4, Figure 5, Figure 6 and Figure 7, show the examples of the new system’s user interfaces.

Figure 4 shows the campaign list page. Users can see the list of all campaigns, search by keyword, and sort by columns. Administrators can create new campaigns by clicking on the “Create” button.

Figure 5 shows the new report editor. This editor supports the markdown format, allowing users to add different types of content to the report, such as tables, hyperlinks, checkboxes, and item lists. Attachments can be added by dropping the file into an editing area or by clicking on the “Add Attachment” button.

Figure 6 shows the comment section in the report page. When a new comment is added to a report, an email will be automatically sent to authors and users commented before. This feature solves the problem that users need to contact authors by emails frequently when some information is missing, and they need to find authors’ email by themselves.

Figure 7 shows the user permission management page. This page is used by administrators to manage validators’ access rights. Administrators can add or remove each person from certain groups in the system. Unlike the previous version of ValDB that requires administrators to input both username and email when adding a new user, the new version requires only email and the system will automatically get users’ full name from their account.

3.7 Conclusion

The new version of ValDB improves the overall user experience from the previous version. It also solves the problems of the previous version by introducing new essential features. The overall user interface looks nicer and easier to understand. Modern technologies applied to the system make it easy to develop and maintain. The custom ORM supports all use-cases in the system. However, the ORM has a performance issue when querying a very large amount of data compared to directly querying with a database. CI/CD helps speed up deployment time, ensure the correctness of the system and reduce manual humen work. Lastly, the data from the previous version is migrated to the new one, so that users can continue using the new version and see historical data of the system.

CERN

ไทยเซิร์น Siam Photon สวทช NSTDA NARIT สทน TINT สสวท THEP CENTER กฟผ. POWER OF INNOVATION FOUNDATION