

BELLE II EVENT DISPLAY WITH PHOENIX

Contributor: **Hieu Le Cong Minh**

Mentors: **Thomas Kuhr, Giacomo De Pietro**

Google Summer of Code 2023 @ CERN-HSF

PROJECT BACKGROUND

The Belle II event display relies on ROOT TEve.

- Requires the installation of the full Belle II software on the local machine.
- Not suitable for modern client-server-based systems and requires significant effort from users.

Solution implemented in GSoC project

Harnessing the display capabilities of today's web browsers for event display

- Developing a web application using **Phoenix**, a TypeScript-based event display framework.

Technologies

- Typescript, Angular
- Phoenix
- JSROOT



PROJECT PROCESSES

Preparation – Community Bonding Period

- Familiarizing myself with the organization and technologies, and then creating a dummy version of a Phoenix event display.

Data Extraction and Visualization – Official Coding Phase Begins!

- Utilizing JSROOT to extract information from .root files and adapting it for web-based event display.
- Using Phoenix to display the extracted data.

User Experience Improvement

- Refining UI tools to elevate user experience.

User Feedback Collection

- Gathering feedback from trial users to enhance the application accordingly.

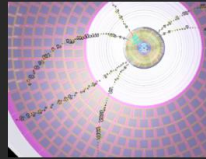
Documentation

- Writing comprehensive documentation for both users and developers.

DELIVERABLES – WEB APPLICATION



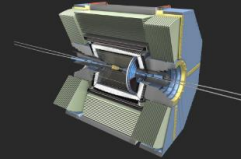
Welcome to the Belle II Experiment



Event Display

Display the event with the simplified detector.

Show



Detector

Show the Belle II detector with full details.

Show

Google Summer of Code 2023

[Github - Documentation](#)

<https://gsoc2023-belle2-display.netlify.app/#/>

DELIVERABLES – WEB APPLICATION

The screenshot displays the Belle II Event Display web application. The main area shows a 3D visualization of a particle collision event, with a central yellow cylinder representing the interaction point and various colored tracks (red, blue, green) extending outwards. The Belle II logo is visible at the top center of the visualization.

Le Cong Minh Hieu
Belle2 Event Display
GSoC 2023

Experiment Info
Experiment 0 / Run 0 / Event 2
Date: 2021-10-12 17:16:19

Display Options

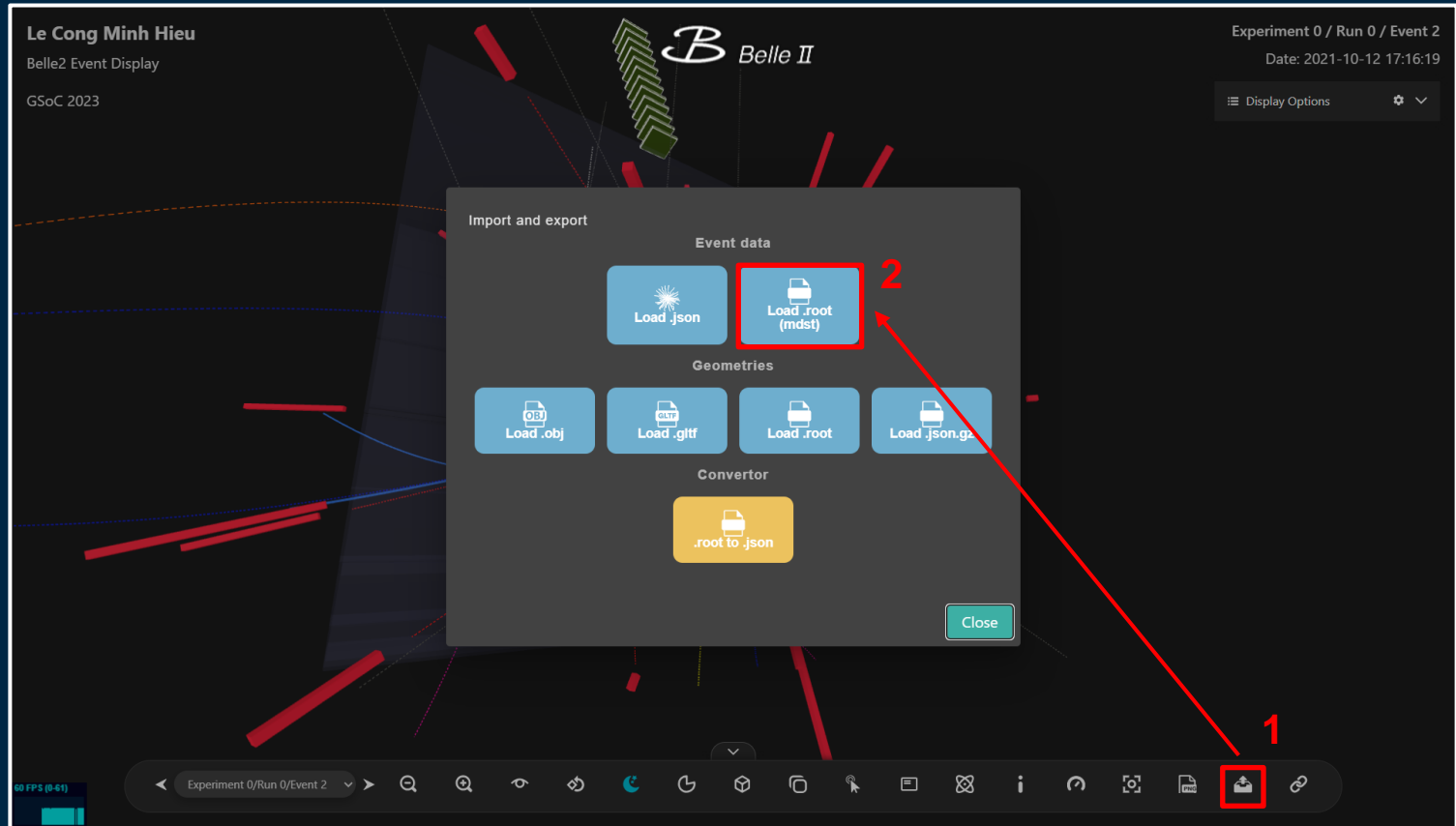
UI Menu

60 FPS (58-60)

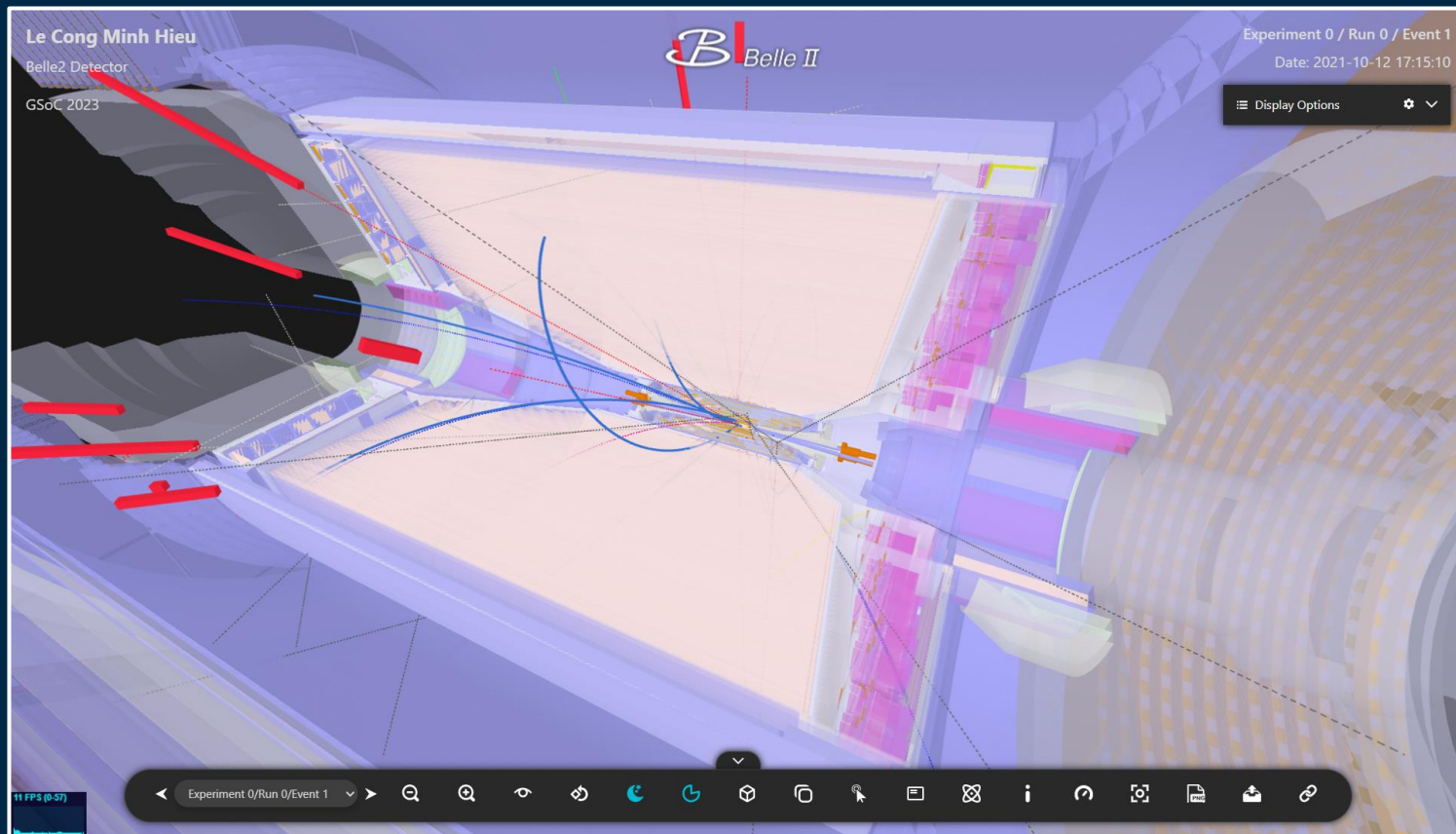
Experiment 0/Run 0/Event 2

The interface includes a top-left header with the user's name and project details, a top-right 'Experiment Info' box with event details, a 'Display Options' menu, and a bottom 'UI Menu' with various navigation and control icons. The main visualization area is a 3D event display showing particle tracks and detector components.

DELIVERABLES – WEB APPLICATION



DELIVERABLES – WEB APPLICATION



<https://gsoc2023-belle2-display.netlify.app/#/detector>

DELIVERABLES – DOCUMENTATION

🏠 Belle II Event Display

Search docs

CONTENTS:

- User Documentation
- Developer Documentation

🏠 / Belle II Event Display [View page source](#)

Belle II Event Display

Belle II Event Display with Phoenix is a project conducted during the Google Summer of Code 2023, aimed at providing a web-based tool for event display. The project's main objective is to enhance the usability and accessibility of Belle II results by allowing users to upload the `mdst.root` file and display the events on their browser. Both user documentation and developer documentation are included on this page.

Contents:

- User Documentation
 - Installation
 - Usage
- Developer Documentation
 - Installation
 - Framework and Tools
 - Event Display
 - Server Integration

User Documentation

Developer Documentation

Check out the [User Documentation](#) section for further information, including how to [manipulate](#) the application.

[Next](#) ↗

© Copyright 2023, Hieu Le Cong Minh.
Built with [Sphinx](#) using a [theme](#) provided by [Read the Docs](#).

FURTHER DEVELOPMENT

Advancing widespread application

- Server implementation (a simple instruction included in the developer documentation).
- Integrating this application into the Belle II software ecosystem.

Exploring additional potentials

- Continuous frontend enhancements: Enhancing user-friendliness and accommodating diverse user requirements.
- Frequent maintenance: Ensuring ongoing reliability and performance.

THANK YOU FOR LISTENING!

Do you have any comments/questions?

You can contact me at hieu.lecongminh@gmail.com