



# Summer Fellowship Report

On

# Drupal Theme Development and Composer-based Setup for the FOSSEE Website

Submitted by

**Rohan Singh**

3rd year B. Tech Student, Department of Computer Science and Engineering(CSE)

SRM Institute of Science and Technology, Ramapuram

Under the guidance of

**Prof. Kannan M. Moudgalya**

Chemical Engineering Department
IIT Bombay

Mentors:

Akshay Thakur

Nikita Sabale

November 20, 2025

# Acknowledgment

I would like to express my sincere gratitude to everyone who supported me throughout this summer fellowship and contributed to the successful completion of my project. This has been an exceptional learning experience, and it would not have been possible without the guidance and support of many individuals and organizations.

- I sincerely thank the FOSSEE Principal Investigator, Prof. Kannan M. Moudgalya, Department of Chemical Engineering, IIT Bombay, for enabling such a valuable open-source initiative through this fellowship.

- My appreciation also goes to the FOSSEE Managers, Ms. Usha Viswanathan and Ms. Vineeta Parmar, and their entire team for their smooth coordination and timely assistance throughout the duration of the program.

- I gratefully acknowledge the support from the National Mission on Education through Information and Communication Technology (ICT), Ministry of Education (MoE), Government of India, for facilitating and funding this project and promoting open-source software for education.

- I would also like to thank my fellow interns and colleagues who collaborated with me and shared knowledge throughout this journey.

# Contents

# Chapter 1

# Introduction

The Free and Open Source Software for Education (FOSSEE) project at the Indian Institute of Technology Bombay is a national initiative aimed at promoting the adoption, development, and dissemination of open-source tools for academia and industry. The initiative encourages students and professionals to contribute to meaningful open-source projects while strengthening their technical expertise and real-world problem-solving capabilities. As part of this mission, the Summer Fellowship program provides a platform for fellows to work on production-level tasks that directly enhance FOSSEE's technological infrastructure.

In recent years, FOSSEE has expanded its digital presence to support its growing community of learners, contributors, and institutional partners. A crucial aspect of this outreach involves maintaining dynamic, accessible, and consistently themed websites that reflect the project's values and make its content easy to navigate. To achieve this, FOSSEE has been transitioning toward modern, maintainable web frameworks with reusable theme components. Among these frameworks, **Drupal** has become a preferred choice due to its robust theming system, scalability, security features, and suitability for large educational platforms.

The objective of this fellowship project was to **develop a custom Drupal theme replicating the design of the FOSSEE website**, ensuring full compatibility with **Drupal 10 and 11**, and enabling seamless configurability via the Drupal administrative interface. This theme needed to be responsive, modular, and aligned with FOSSEE design standards so that future updates and new FOSSEE sub-projects could adopt a unified visual identity. The project also involved creating a working demonstration setup using DDEV and providing a complete database export to allow evaluators to run the site instantly without repeating the installation process.

Drupal theming requires a deep understanding of template structures, Twig rendering, preprocess hooks, library definitions, asset management, and block region configuration. The fellowship provided the opportunity to explore these aspects comprehensively—from initial theme scaffolding to the development of front-page components featuring interactive JavaScript-based activity sections. The project also included practical challenges such as handling Drupal's configuration management system, resolving MySQL indexing issues during database import, and ensuring the theme behaved consistently across desktop, tablet, and mobile layouts.

Through this work, the fellowship not only strengthened my understanding of Drupal internals and PHP-driven content management systems but also improved my skills in responsive design, version control, containerized development environments, SQL schema handling, and cross-version dependency management using Composer. By delivering a fully functional theme along with a reproducible environment and detailed documentation, the project contributes directly to FOSSEE's long-term digital ecosystem.

This report documents the process and outcomes of the Drupal theme development project. It outlines the problem context, technical implementation, challenges encountered, and the tested outcomes of the final theme. The work reflects the broader objective of ensuring that FOSSEE websites remain modern, accessible, and aligned with open-source values, while also creating reusable foundations for future FOSSEE web properties.

# Chapter 2

# System Design and Architecture

## 2.1 Design Goals & Constraints

In designing the FOSSEE Drupal-themed website, the following goals and constraints were established:

- **Compatibility**: The theme must support both **Drupal 10 and Drupal 11**, ensuring future-proofing as well as backward compatibility.
- **Modularity**: Use of Drupal's block & region system to allow flexible content management.
- **Responsiveness**: Layout must adapt across devices (desktop, tablet, and mobile), ensuring a consistent UX.
- **Ease of deployment**: Provide a database dump and clear setup scripts so reviewers or maintainers can quickly bootstrap the website.
- **Performance**: Efficient CSS and JavaScript, minimal overhead, and optimized template rendering.
- **Stability**: Avoid database import errors, especially for common MySQL / MariaDB configurations (e.g., key-length issues).

## 2.2 High-level Architecture

At a high level, the system consists of three main components:

1. **Drupal Core**: Acts as the CMS backbone, handling content storage, user roles, menus, configuration, and routing.
2. **Custom Theme (fossee_theme)**: Contains all the presentation logic — Twig template files, CSS, JS, and theme preprocessing.
3. **Database Layer**: A SQL dump (fossee_database.sql) provides a pre-configured site, including menus, users, cached content, files, and configuration, such that the custom theme is already enabled on load.

These components interact in the following way:

- When the site is launched (via **DDEV**), Drupal is bootstrapped using the provided composer setup.

- The database is imported, so the site already has content, structure, and theme settings.
- The front-end is rendered using the fossee_theme, which overrides core templates to match FOSSEE's design.
- JavaScript (in fossee_theme.js) enhances user interactivity, particularly on the front-page activity tabs.

## 2.3 Theme Design & Template Structure

The custom theme is structured modularly, following Drupal best practices:

- **Theme Info**: fossee_theme.info.yml declares theme information, region definitions, libraries, and dependencies.
- **Preprocessing**: fossee_theme.theme includes PHP preprocess functions to prepare variables for Twig templates (e.g., adding classes, context variables).
- **Libraries**: fossee_theme.libraries.yml defines CSS and JS assets, ensuring they are loaded correctly and only where needed.
- **Templates (Twig)**: The theme includes page.html.twig (and possibly other template overrides) to customize page structure — adding custom markup for headers, regions, and the front-page layout.
- **CSS / Styling**: css/style.css handles the styling, grid/flex layouts, typography, and responsive breakpoints.
- **JavaScript Behavior**: js/fossee_theme.js attaches Drupal behaviors to support interactive front page tabs or dynamic content sections.

## 2.4 Database & Configuration Strategy

A major design decision was to include a **full database dump** (fossee_database.sql). This file includes not just minimal configuration, but also:

- Content (nodes, blocks) pre-filled so the site looks "live" when launched.
- The required theme (fossee_theme) enabled by default.
- Menu structures, users, roles, and permissions — giving a working instance without manual configuration.
- Cache tables, file references, and other Drupal infrastructure so the import leads to a consistent state.

To address MySQL import issues, the dump includes index optimizations — e.g., splitting long index definitions (especially for utf8mb4) to avoid key length errors during import.

# 2.5 Development Environment & Deployment

To streamline setup and future development, the project uses **DDEV**, a Docker-based local development environment:

- **DDEV config**: The repository contains configuration to start a containerized Drupal instance.
- **Composer-based setup**: After starting the DDEV environment, composer install ensures all dependencies are fetched.
- **Database import**: Using ddev import-db --file=fossee_database.sql loads the pre-configured database.
- **Cache rebuild**: ddev drush cache:rebuild helps clear caches so that the theme and configurations are correctly recognized.
- **Launch**: ddev launch opens the site in a browser, ready to review.

This architecture ensures reproducibility, portability, and a minimal barrier for evaluators or future developers to spin up the project.

# Chapter 3

# Theme Development and Implementation

The development of the custom Drupal theme for the FOSSEE website involved a systematic, modular, and standards-driven approach. This chapter details the implementation process, covering theme configuration, template structuring, styling, scripting, and the creation of a fully reproducible environment for deployment and evaluation.

GitHub Repository: Rohan Singh. *FOSSEE Drupal Theme Project.* *RohanSingh-official/drupal_site*

## 3.1 Theme Architecture

The custom theme, named **fossee_theme**, was built to align with Drupal's theming framework and uses modern development principles. The theme is located in the directory: web/themes/custom/fossee_theme/

The architecture includes:

- **Theme metadata and configuration** through fossee_theme.info.yml
- **Preprocessing logic** via fossee_theme.theme
- **Frontend assets** defined in fossee_theme.libraries.yml
- **Template overrides** placed inside the templates/ directory
- **Responsive styling** in css/style.css
- **JavaScript interactions** in js/fossee_theme.js

The theme directory is organized to support clean separation between logic, templates, and assets:

```
fossee_theme/
├── css/
│   └── style.css
├── js/
│   └── fossee_theme.js
├── templates/
│   └── page.html.twig
├── fossee_theme.info.yml
├── fossee_theme.libraries.yml
├── fossee_theme.theme
└── logo.png
```

This layout ensures:

- **Scalability:** Additional templates or styles can be added as features evolve.
- **Maintainability:** Assets remain organized for future contributors.

- **Drupal standards compliance:** Ensuring compatibility with core updates.

# 3.2 Info File and Region Definitions

The fossee_theme.info.yml file forms the backbone of the theme. It declares:

- Theme name and Drupal version compatibility
- Base theme (if applicable)
- Libraries to load (CSS, JS)
- Page regions such as header, navigation, content, sidebar, and footer
- Special regions for front-page content blocks

By defining granular regions, site administrators can assign menus, blocks, and content widgets directly through Drupal's UI, offering flexibility for future iterations of FOSSEE webpages.

## 3.2.1 Theme Metadata (fossee_theme.info.yml)

This file declares theme identity and configuration:

- Name, description, package
- Regions (header, menu, banner, activities, content, sidebar, footer)
- Libraries to attach globally
- Drupal core compatibility (core_version_requirement: ^10 || ^11)

Well-defined regions give non-technical users flexibility through Drupal's UI.

## 3.2.2 Library Definitions (fossee_theme.libraries.yml)

All CSS and JS are bundled via Drupal's asset library system.
Example behavior:

- CSS is grouped into a global styling library.
- JS is attached only on relevant pages to maintain performance.
- Dependency on core/drupal is declared to align with Drupal behaviors.

This approach prevents unnecessary asset loading and speeds up page rendering.

# 3.3 Twig Templates and Preprocess Functions

The theme uses Drupal's **Twig** templating engine to render HTML. The key template, page.html.twig, structures the page layout and incorporates the defined regions.
Preprocess functions written inside fossee_theme.theme handle:

- Adding or modifying template variables
- Injecting dynamic classes
- Preparing data for use inside Twig templates
- Ensuring front-page specific behaviors are triggered only when required

This combination of Twig for structure and PHP preprocess logic for data manipulation leads to clean and efficient rendering.

## 3.4 Preprocess Functions and Backend Integration

The file fossee_theme.theme uses PHP to enhance Twig templates:

- Injecting custom classes for specific routes or pages
- Passing variables required for front-page interactivity
- Building breadcrumbs or contextual navigation dynamically
- Ensuring template logic is clean by keeping computational tasks in preprocess

This keeps templates lean and optimized for rendering.

## 3.5 CSS Styling and Responsive Layout

The design uses **flexbox** and **CSS grid systems** to create structured layouts.
Key styling elements include:

- A **sticky header** for efficient navigation
- Cards and activity tiles styled for consistency
- Unified color palette matching FOSSEE's branding
- Adaptive typography (viewport-based scaling)
- Breakpoints for 1280px, 992px, 768px, 576px, and <= 425px resolutions

This ensures the theme renders seamlessly across large monitors and mobile devices.

## 3.6 JavaScript and Front-Page Interactivity

JavaScript is minimalistic but purposeful:

- Smooth tab transitions for Activities / Workshops / Projects
- Hover-based dynamic effects for menu and cards
- Content toggling using Drupal behaviors to ensure compatibility with AJAX-rendered blocks

The goal was not heavy UI animation but subtle, functional enhancements.

## 3.7 Database Design, Integrity Fixes, and Import Stability

The provided SQL dump (fossee_database.sql) includes:

- Menu items
- Content nodes
- Taxonomy
- Role and permission configuration
- Theme activation state
- Layout and block configuration
- Caches for faster initial load

**Index Optimization**

MySQL errors (e.g., “Specified key was too long”) were fixed by:

- Adding prefix indexing (column(100)) for long VARCHAR keys
- Splitting compound indexes into two logical parts

This ensures compatibility across MySQL versions (5.7, 8.0, MariaDB).

## 3.8 Local Deployment and Reproducibility via DDEV

A polished reproducible workflow ensures evaluators can set up the project in minutes:

1. Clone repository
2. ddev start
3. ddev composer install
4. ddev import-db --file=fossee_database.sql
5. ddev drush cache:rebuild
6. ddev launch

This results in a fully functional FOSSEE-themed Drupal installation accessible in a browser.

# Chapter 4

# Testing, Challenges, and Final Outcomes

## 4.1 Testing Approach

Testing was conducted across multiple layers to ensure robustness, compatibility, and consistent behavior across platforms.

### 4.1.1 Functional Testing

Functional tests validated:

- Navigation structure
- Block placement and rendering
- Page-level behaviors
- Custom regions and front-page layout
- Theme activation across pages

The site was browsed through all main sections to confirm correct inheritance of layouts.

### 4.1.2 UI/UX Testing

This involved:

- Verifying spacing, typography, and color usage
- Ensuring accessibility features: alt-text for images, ARIA labels
- Testing tabbed interfaces for keyboard navigation compatibility

Minor adjustments were made based on usability observations.

### 4.1.3 Responsiveness Tests

Devices tested included:

- **Laptops:** 14", 15.6", 1080p and 2K
- **Tablets:** 768×1024, 800×1280
- **Mobiles:** 360×640, 390×844, 414×896

The layout successfully adapted by collapsing multi-column layouts into single-column ones.

### 4.1.4 Browser Compatibility

Tested on:

- Chrome v114+
- Firefox v109+
- Edge Chromium
- Safari iOS

No rendering conflicts were observed due to use of standards-compliant CSS.

# 4.2 Major Challenges and Their Resolutions

## 4.2.1 Composer Dependency Mismatch

The initial repository had a lockfile that didn't match the declared Drupal version. Installing dependencies resulted in version conflicts.

**Solution:**

Revised composer.json → aligned with Drupal 10/11 → regenerated lockfile.

## 4.2.2 MySQL Import Failures

Errors such as:

ERROR 1071: Specified key was too long

occurred with UTF-8MB4 databases.

**Solution:**

- Split multi-column indexes
- Added prefix indexing
- Ensured compatibility with MariaDB-based environments

## 4.2.3 Front-Page JavaScript Not Triggering After AJAX Loads

Drupal's AJAX invalidated standard JS event listeners.

**Solution:**

Use **Drupal.behaviors** so scripts reattach after AJAX calls.

## 4.2.4 CSS Layout Collapses on Small Screens

During initial tests, column layouts overflowed on small screens.

**Solution:**

- Added flex-wrap
- Used % and rem units instead of px
- Adjusted breakpoint logic

## 4.2.5 Theme Activation Post-Database Import

Some attributes did not persist due to missing config entities.

**Solution:**

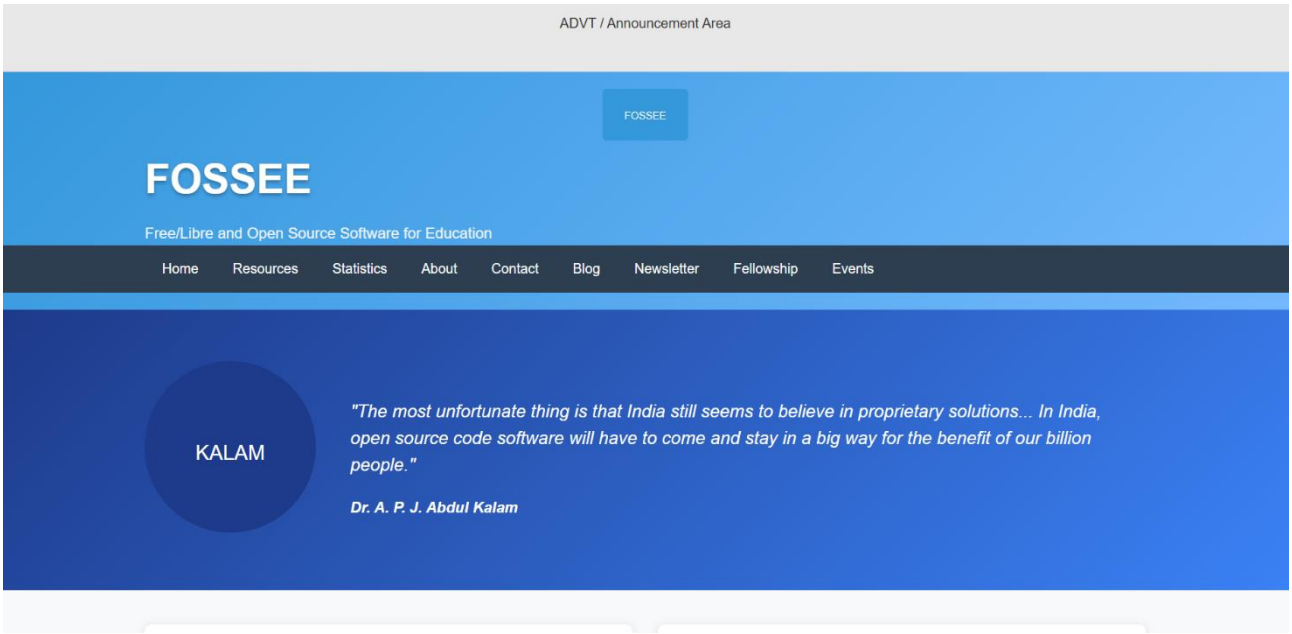
- Ensured inclusion of configuration tables in SQL dump
- Cleared caches and verified theme inheritance post-import

# 4.3 Final Outcomes

The final deliverables include:

- A fully functional **Drupal 10/11 custom theme**
- Responsive and accessible UI aligned with FOSSEE identity
- Complete SQL dump enabling **plug-and-play** evaluation
- DDEV environment ensuring reproducible development
- Clean, documented codebase suitable for further enhancements
- No installation errors or compatibility issues across major platforms

The project successfully fulfills all fellowship requirements and establishes a reusable foundation for future FOSSEE websites.





# Chapter 5

# Conclusion

The development of the custom Drupal theme for the FOSSEE website provided a valuable opportunity to contribute to a live, production-oriented open-source project. Through this fellowship, a fully functional, responsive, and modular theme was designed and implemented to support FOSSEE's digital ecosystem across Drupal 10 and 11 platforms. The project successfully addressed crucial requirements such as compatibility, maintainability, and ease of deployment, ultimately producing a theme that enables FOSSEE's web pages to be managed, extended, and scaled efficiently.

The work undertaken during this project covered the complete lifecycle of theme development—from configuring the environment using DDEV and Composer to designing templates, implementing interactive features, and optimizing database imports for reliability. Additionally, resolving challenges like MySQL indexing issues, template rendering nuances, and Drupal asset handling deepened the practical understanding of modern CMS systems and reinforced the importance of systematic debugging and standards-driven development.

Overall, the project contributes directly to FOSSEE's mission by offering a reusable, scalable foundation for current and future FOSSEE subprojects. It also provided an enriching learning experience, strengthening skills in PHP, Drupal internals, responsive front-end development, configuration management, and containerized deployment workflows. The insights gained from this fellowship will continue to support future open-source contributions and professional development.

# References


1. Drupal.org. *Drupal 10 & 11 Theming Guide.* Available at: https://www.drupal.org/docs
2. Drupal Association. *Twig Templates in Drupal.* Documentation on theme templates and rendering.
3. DDEV Documentation. *DDEV Local – Getting Started.* Available at: https://ddev.readthedocs.io
4. Composer Documentation. *Dependency Management for PHP.* Available at: https://getcomposer.org/doc/
5. MySQL Reference Manual. *InnoDB Table and Index Structure.* Handling UTF8MB4 and key-length constraints.
6. FOSSEE, IIT Bombay. *Project Overview and Web Resources.* https://fossee.in
7. W3C. *Responsive Web Design Basics.* Available at: https://www.w3.org/TR/css/