FOSSEE Summer Fellowship Report

On

Development of Struts In Trusses Module for Osdag

Submitted by

Mohit Rana

3rd Year B.Sc. IT Student, Department of Computer Science

Graphic Era Hill University

Dehradun

Under the Guidance of

Prof. Siddhartha Ghosh

Department of Civil Engineering

Indian Institute of Technology Bombay

Mentors:

Ajmal Babu M S

Parth Karia

Ajinkya Dahale

July 23, 2025

Acknowledgments

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 extend my heartfelt thanks to the project staff at the Osdag team, especially **Mr. Ajmal Babu M. S.**, **Mr. Ajinkya Dahale**, and **Mr. Parth Karia**, for their invaluable technical guidance, patience, and continuous support throughout the project.
- I am extremely grateful to the Osdag Principal Investigator (PI), **Prof. Siddhartha Ghosh**, Department of Civil Engineering at IIT Bombay, for his visionary leadership and support.
- 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.

- Lastly, I wish to acknowledge the support from my college, **Graphic Era Hill University**, the Department of Information Technology, my professors, and the administration for encouraging me to participate in this fellowship and supporting my learning endeavors.

Contents

1 Introduction **5**

1.1 National Mission in Education through ICT 5

1.1.1 ICT Initiatives of MoE 6

1.2 FOSSEE Project 7

1.2.1 Projects and Activities 7

1.2.2 Fellowships 7

1.3 Osdag Software 8

1.3.1 Osdag GUI 9

1.3.2 Features 9

2 Screening Task **10**

2.1 Problem Statement 10

2.2 Tasks Done 10

3 Task 1: Struts In Trusses Module Integration in Osdag Web Application 12

3.1 Task 1: Problem Statement 12

3.2 Task 1: Tasks Done 12

3.3 Task 1: Python and JavaScript Code 14

3.3.1 Description of the Code 14

3.3.2 JavaScript Code 14

3.3.3 Python Code 15

3.3.4 Explanation of the Code 15

3.4 Task 1: Documentation 17

3.4.1 Directory Structure 17

3.5 Summary of My Contribution 18

4 Task 2: Restructuring the Struts In Trusses Module (Frontend) **19**

4.1 Problem Statement 19

4.2 Tasks Done 19

4.3 Code and Structure 21

4.3.1 Directory Structure (After Refactor) 21

4.3.2 Code Snippets 22
4.4 Summary of My Contribution 23

5 Conclusions **24**
5.1 Tasks Accomplished 24
5.2 Skills Developed 25

A Appendix **28**
A.1 Work Reports 28

Bibliography **31**

Chapter 1

Introduction

1.1 National Mission in Education through ICT

The National Mission on Education through ICT (NMEICT) is a scheme under the Department of Higher Education, Ministry of Education, Government of India. It aims to leverage the potential of ICT to enhance teaching and learning in Higher Education Institutions in an anytime-anywhere mode.

The mission aligns with the three cardinal principles of the Education Policy—**access, equity, and quality**—by:

- Providing connectivity and affordable access devices for learners and institutions.
- Generating high-quality e-content free of cost.

NMEICT seeks to bridge the digital divide by empowering learners and teachers in urban and rural areas, fostering inclusivity in the knowledge economy. Key focus areas include:

- Development of e-learning pedagogies and virtual laboratories.
- Online testing, certification, and mentorship through accessible platforms like EduSAT and DTH.
- Training and empowering teachers to adopt ICT-based teaching methods.

For further details, visit the official website: www.nmeict.ac.in.

1.1.1 ICT Initiatives of MoE

The Ministry of Education (MoE) has launched several ICT initiatives aimed at students, researchers, and institutions. The table below summarizes the key details:

No.	Resource	For Students/Researchers	For Institutions
Audio-Video e-content			
1	SWAYAM	Earn credit via online courses	Develop and host courses; accept credits
2	SWAYAMPRABHA	Access 24x7 TV programs	Enable SWAYAMPRABHA viewing facilities
Digital Content Access			
3	National Digital Library	Access e-content in multiple disciplines	List e-content; form NDL Clubs
4	e-PG Pathshala	Access free books and e-content	Host e-books
5	Shodhganga	Access Indian research theses	List institutional theses
6	e-ShodhSindhu	Access full-text e-resources	Access e-resources for institutions
Hands-on Learning			
7	e-Yantra	Hands-on embedded systems training	Create e-Yantra labs with IIT Bombay
8	FOSSEE	Volunteer for open-source software	Run labs with open-source software
9	Spoken Tutorial	Learn IT skills via tutorials	Provide self-learning IT content
10	Virtual Labs	Perform online experiments	Develop curriculum-based experiments
E-Governance			
11	SAMARTH ERP	Manage student lifecycle digitally	Enable institutional e-governance
Tracking and Research Tools			
12	VIDWAN	Register and access experts	Monitor faculty research outcomes
13	Shodh Shuddhi	Ensure plagiarism-free work	Improve research quality and reputation
14	Academic Bank of Credits	Store and transfer credits	Facilitate credit redemption

Table 1.1: Summary of ICT Initiatives by the Ministry of Education

1.2 FOSSEE Project

The FOSSEE (Free/Libre and Open Source Software for Education) project promotes the use of FLOSS tools in academia and research. It is part of the National Mission on Education through Information and Communication Technology (NMEICT), Ministry of Education (MoE), Government of India.

1.2.1 Projects and Activities

The FOSSEE Project supports the use of various FLOSS tools to enhance education and research. Key activities include:

- **Textbook Companion**: Porting solved examples from textbooks using FLOSS.
- **Lab Migration**: Facilitating the migration of proprietary labs to FLOSS alternatives.
- **Niche Software Activities**: Specialized activities to promote niche software tools.
- **Forums**: Providing a collaborative space for users.
- **Workshops and Conferences**: Organizing events to train and inform users.

1.2.2 Fellowships

FOSSEE offers various internship and fellowship opportunities for students:

- Winter Internship
- Summer Fellowship
- Semester-Long Internship

Students from any degree and academic stage can apply for these internships. Selection is based on the completion of screening tasks involving programming, scientific computing, or data collection that benefit the FLOSS community. These tasks are designed to be completed within a week.

For more details, visit the official FOSSEE website.

Figure 1.1: FOSSEE Projects and Activities

1.3 Osdag Software

Osdag (Open steel design and graphics) is a cross-platform, free/libre and open-source software designed for the detailing and design of steel structures based on the Indian Standard IS 800:2007. It allows users to design steel connections, members, and systems through an interactive graphical user interface (GUI) and provides 3D visualizations of designed components. The software enables easy export of CAD models to drafting tools for construction/fabrication drawings, with optimized designs following industry best practices [1, 2, 3]. Built on Python and several Python-based FLOSS tools (e.g., PyQt and PythonOCC), Osdag is licensed under the GNU Lesser General Public License (LGPL) Version 3.

1.3.1 Osdag GUI

The Osdag GUI is designed to be user-friendly and interactive. It consists of

- **Input Dock**: Collects and validates user inputs.
- **Output Dock**: Displays design results after validation.
- **CAD Window**: Displays the 3D CAD model, where users can pan, zoom, and rotate the design.
- **Message Log**: Shows errors, warnings, and suggestions based on design checks.

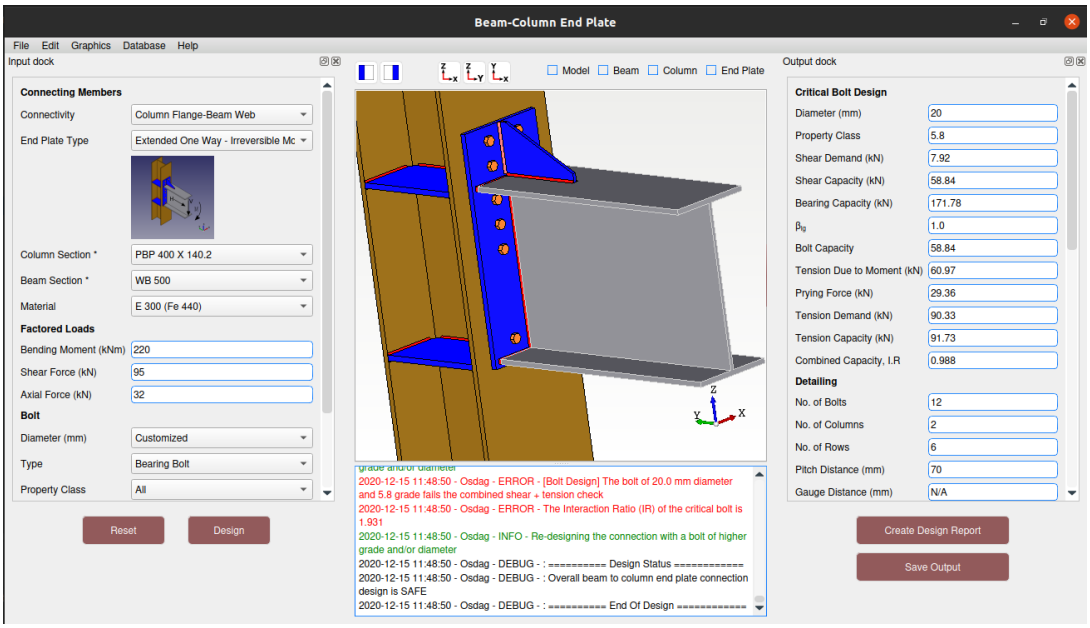


Figure 1.2: Osdag GUI

1.3.2 Features

- **CAD Model**: The 3D CAD model is color-coded and can be saved in multiple formats such as IGS, STL, and STEP.
- **Design Preferences**: Customizes the design process, with advanced users able to set preferences for bolts, welds, and detailing.
- **Design Report**: Creates a detailed report in PDF format, summarizing all checks, calculations, and design details, including any discrepancies.

For more details, visit the official Osdag website.

Chapter 2

Screening Task

2.1 Problem Statement

Create the UI of the Osdag web app using React and develop endpoints using Django/REST. (Any stream)

2.2 Tasks Done

For my Osdag web app project, I chose to work on the **Beam to Beam End Plate** module. My work involved both backend and frontend development, ensuring a complete and functional workflow for this connection type.

Backend

- Set up the backend using **Django** and **Django REST Framework**.
- Created a REST API endpoint (`/api/beam-to-beam-connection/`) accepting design parameters via `POST` requests.
- Implemented logic for bolt design (shear, tension, prying), plate sizing, and weld checks.
- API returns a structured JSON with all computed results.
- Followed Django best practices for a modular and reusable codebase.

Frontend

- Built with **React**, focusing on a clean and responsive UI.
- Structured UI into three sections: **Input Dock**, **View and Log**, and **Output Dock**.
- Integrated backend using `axios`; managed state with `zustand`.

Integration

- Ensured smooth data flow between frontend and backend.
- Maintained clean, maintainable code following best practices.

In summary, I have successfully implemented the Beam to Beam End Plate module as a full-stack feature, covering both the engineering logic on the backend and a modern, interactive UI on the frontend, with smooth API integration between the two.

Chapter 3

Task 1: Struts In Trusses Module Integration in Osdag Web Application

3.1 Task 1: Problem Statement

The objective was to integrate a new module named **Struts In Trusses** into the Osdag web application. This module allows users to design and analyze strut members in truss structures, providing a seamless experience from data input to output, including 3D visualization and PDF report generation. The implementation needed to align with existing architectural standards in Osdag, ensuring maintainability, scalability, and user-friendliness.

3.2 Task 1: Tasks Done

1. Frontend Integration (osdagclient)

a. Component Creation:

- **File Added:** `osdagclient/src/components/compressionMember/StrutsInTrusses.jsx`
- **Function:** Main UI for user inputs, output dock, CAD visualization, logs, and report generation.

b. Input Handling:

- Managed inputs: Section profile, material, length, end conditions, load, etc.

- Used Ant Design components (`Select, Input, Modal, Button`) for user inputs.
- State management with `useState` and `useEffect`.

c. API Integration via Context:

- Integrated with `ModuleContext` for API communication:
  - `createSession("Struts In Trusses")`
  - `createDesign(param, "Struts-In-Trusses")`
  - `createDesignReport()`
  - `getDesingPrefData()`

d. Output Handling:

- `StrutsInTrussesOutputDock.jsx` displays results.
- Logs shown via `Logs.jsx`.
- CAD model rendered

e. Supporting Components:

- `DesignPrefSections.jsx`, `CustomSectionModal.jsx`, `DropdownMenu.jsx`, `ScreenShotCapture`

2. Backend Integration

a. API Endpoints:

- Session: `createSession`, `deleteSession`
- Design: Receives parameters and returns analysis.
- CAD: Generates 3D model files.
- Report: Generates PDF reports.

b. Design Logic:

- File: `design_type/compression_member/compression.py`
- Calculates slenderness, critical load, checks safety, etc.

c. CAD Generation:

- Folder: `cad/items/`
- Generates 3D models and returns paths to frontend.

d. Report Generation:

- Folder: `design_report/`
- Uses metadata from frontend and renders report via LaTeX.

e. State Management:

- Ensures isolated session data and cleanup.

3.3 Task 1: Python and JavaScript Code

3.3.1 Description of the Code

- **Frontend:** Collects input, communicates with backend, renders CAD, manages state and report generation.
- **Backend:** Validates input, performs structural calculations, generates 3D models and reports.

3.3.2 JavaScript Code

Listing 3.1: API Call in React Frontend

```
const handleSubmit = async () => {
  let param = {
    "Connector.Material": inputs.connector_material,
    "Conn_Location": inputs.connection_section,
    "End_1": inputs.end1,
    "End_2": inputs.end2,
    "Member.Designation": inputs.section_designation,
    // ...other params...
  };
```

```
  createDesign(param, "Struts-In-Trusses");
  setDisplayOutput(true);
  setLoading(true);
};
```

3.3.3 Python Code

Listing 3.2: Backend Calculation for Strut Design

```
def design_strut_in_truss(params):
    section = params.get("Member.Designation")
    length = float(params.get("Member.Length"))
    material = params.get("Connector.Material")
    axial_load = float(params.get("Load.Axial"))

    # Slenderness ratio calculation
    slenderness = length / get_radius_of_gyration(section)

    # Critical load
    critical_load = calculate_critical_load(section, slenderness,
        material)

    # Safety check
    is_safe = critical_load >= axial_load

    return {
        "Slenderness Ratio": slenderness,
        "Critical Load": critical_load,
        "Is Safe": is_safe,
        # ...other results...
    }
```

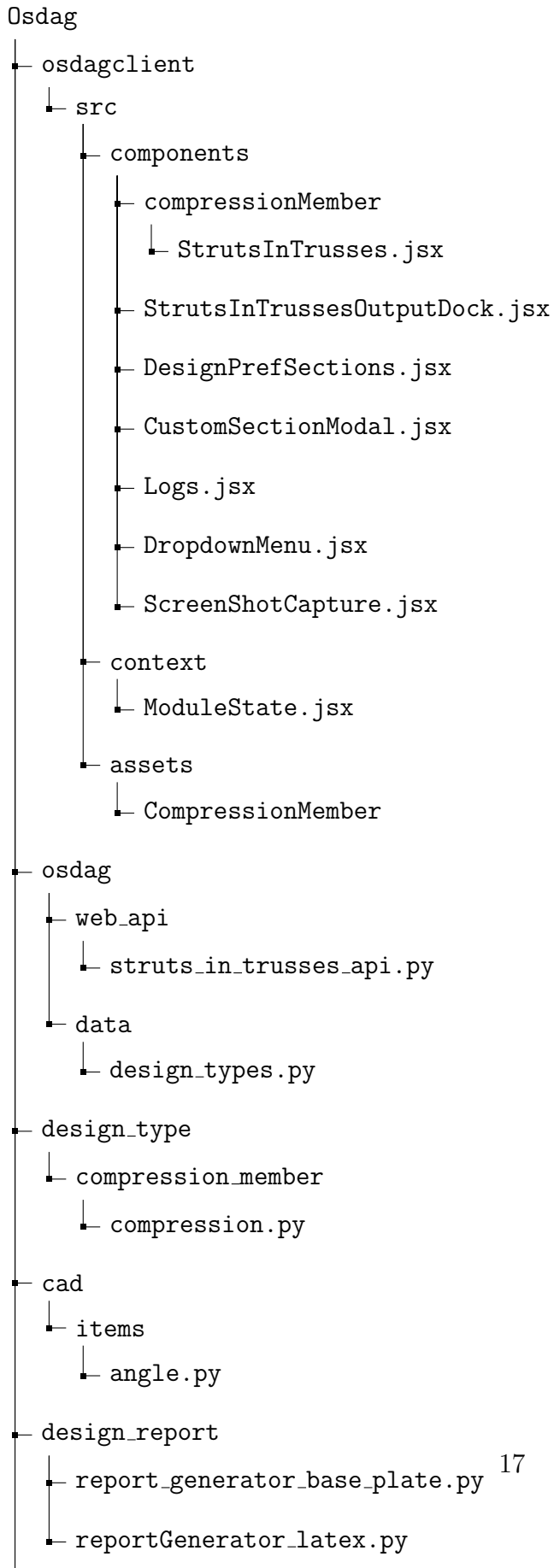
3.3.4 Explanation of the Code

- **Lines 1–4:** Extracts design parameters from frontend.

- **Lines 6–7:** Performs structural calculations using IS code.
- **Lines 8–10:** Performs adequacy check and returns results.

3.4 Task 1: Documentation

3.4.1 Directory Structure



Program Start Calls

- **Frontend:**

```
cd osdagclient
npm start
```

- **Backend:**

```
python manage.py runserver
```

3.5 Summary of My Contribution

- Analyzed the architectural flow of Osdag and ensured new module consistency.
- Developed complete frontend component for **Struts In Trusses**.
- Integrated backend APIs for calculation, CAD generation, and reporting.
- Implemented calculation logic for strut members.
- Added support for input validation, session state, and design preferences.
- Rendered 3D CAD models using FreeCAD integration.
- Tested the full workflow, handled edge cases, and documented the implementation.

This comprehensive implementation ensures the new module is robust, maintainable, and user-friendly, aligning with the standards and architecture of the Osdag web application.

Chapter 4

Task 2: Restructuring the Struts In Trusses Module (Frontend)

4.1 Problem Statement

The objective of Task 2 was to **refactor and restructure the Struts In Trusses module's frontend** in the Osdag web application. The initial implementation was monolithic, tightly coupled, and hard to maintain. The goal was to modularize the code, improve readability, and align the structure with other modules in the updated codebase. This restructuring aimed to facilitate ease of development, scalability, and improved onboarding for new contributors.

4.2 Tasks Done

1. Frontend Restructuring (`osdagclient/src/modules/compressionMember/str`

a. Directory and File Organization

- Created a dedicated folder: `osdagclient/src/modules/compressionMember/strutsInTrusses/`
- Followed the updated architecture pattern for modular separation.
- Organized files into:
  - Main container/component

  - `configs/` for static configuration
  - `components/` for inputs, outputs, modals
  - `assets/` for diagrams/images

b. Component Refactoring

- Moved orchestration logic to `StrutsInTrusses.jsx`
- Broke down into smaller, manageable sub-components:
  - `InputSection.jsx`
  - `OutputDock.jsx`
  - `ReportModal.jsx`
  - Dropdowns, logs, viewer components
- Ensured clear data flow via props and context

c. Configuration and Constants

- Moved all static dropdown options, defaults, and validations to `configs/strutsInTrussesConfig`
- Decoupled UI logic from data for better maintainability

d. API and Context Integration

- Centralized API calls (`createSession`, `createDesign`, `createDesignReport`) using context
- Handled loading and error states effectively
- Shared design data, logs, and model paths via `ModuleContext`

e. UI/UX Improvements

- Adopted consistent UI styling using Ant Design
- Ensured dynamic updates to diagrams based on user input
- Improved accessibility and responsiveness across devices

f. Supporting Files and Utilities

- Moved images to `assets/CompressionMember/`
- Scoped styles using CSS modules if needed

2. Integration and Flow

Refactored Module Workflow:

1. User navigates to the Struts In Trusses module via main menu.
2. `StrutsInTrusses.jsx` initializes a backend session on mount using `createSession("Struts In Trusses")`.
3. Inputs are collected and validated through `InputSection`.
4. On clicking "Design", data is passed to `createDesign()` and handled asynchronously.
5. `OutputDock` displays results and logs. CAD viewer is updated with 3D model.
6. `ReportModal` collects metadata and calls `createDesignReport` for PDF generation.
7. Preferences and custom section modals are triggered from corresponding UI elements.
8. On module exit, session cleanup is performed.

4.3 Code and Structure

4.3.1 Directory Structure (After Refactor)

Listing 4.1: New Folder Structure

```
osdagclient/
          src/
              modules/
                  compressionMember/
                      strutsInTrusses/
                          StrutsInTrusses.jsx
                          configs/
                                strutsInTrussesConfig.js
                          components/
                                InputSection.jsx
                                OutputDock.jsx
```

```
                                    ReportModal.jsx
                                    ... (other sub-components)
                              assets/
                                    ... (images, diagrams)
```

4.3.2 Code Snippets

Main Component

Listing 4.2: StrutsInTrusses.jsx (simplified)

```
import InputSection from './components/InputSection';
import OutputDock from './components/OutputDock';
import ReportModal from './components/ReportModal';

const StrutsInTrusses = () => {
  const { createSession, createDesign, createDesignReport } =
     useContext(ModuleContext);
  const [inputs, setInputs] = useState({});
  const [output, setOutput] = useState(null);

  useEffect(() => {
    createSession("Struts In Trusses");
  }, []);

  const handleDesign = () => {
    createDesign(inputs, "Struts-In-Trusses");
  };

  return (
    <>
      <InputSection inputs={inputs} setInputs={setInputs} />
      <button onClick={handleDesign}>Design</button>
      {output && <OutputDock output={output} />}
      <ReportModal />
    </>
```

```
  );
};
```

Config File

Listing 4.3: Config Options for Dropdowns

```
export const sectionOptions = [
  { label: "ISA 100x75x8", value: "ISA 100x75x8" },
  { label: "ISA 130x65x10", value: "ISA 130x65x10" },
];

export const materialOptions = [
  { label: "Fe 410", value: "Fe 410" },
  { label: "Fe 250", value: "Fe 250" },
];
```

4.4 Summary of My Contribution

- Identified architectural and structural issues in the original implementation.
- Created a new folder structure specific to the module under `modules/compressionMember`.
- Refactored the main component and split logic into smaller sub-components.
- Migrated static configuration to dedicated config files.
- Integrated API calls using centralized context and ensured clean state management.
- Improved UI/UX design consistency and responsiveness.
- Validated the full end-to-end workflow post-refactoring.

This restructuring has significantly enhanced the readability, modularity, and scalability of the **Struts In Trusses** module, making it aligned with modern frontend architecture and Osdag's evolving codebase.

Chapter 5

Conclusions

This summer fellowship at FOSSEE, IIT Bombay, provided an immersive experience in developing and enhancing open-source software for structural engineering. The work centered on the Osdag platform, focusing on the implementation and refinement of the **Struts in Trusses** module. The fellowship was divided into two primary tasks: the initial end-to-end development of the module and its subsequent architectural restructuring. The successful completion of these tasks has resulted in a robust, user-friendly, and maintainable addition to the Osdag ecosystem.

5.1 Tasks Accomplished

Throughout the fellowship, a comprehensive set of tasks were successfully completed, spanning the full software development lifecycle:

- **End-to-End Module Development:** The primary achievement was the creation of the *Struts in Trusses* compression member module. This involved:
  - **Frontend Implementation:** A complete user interface was developed using React, allowing users to input design parameters, select preferences, and visualize results.
  - **Backend Integration:** The frontend was successfully linked with the Python-based backend. This included integrating APIs for structural calculations, 3D CAD model generation using FreeCAD, and the creation of detailed LaTeX-based design reports.

- **Calculation Logic:** The core design logic for strut members was implemented on the backend, ensuring accurate engineering calculations based on user inputs.
- **State Management:** Session state management and input validation were implemented to ensure a smooth and error-free user experience.

- **Architectural Refactoring of the Frontend:** The second major task involved a significant restructuring of the *Struts in Trusses* module to align with the latest architectural standards of the Osdag project.
  - **Modularization:** The monolithic frontend was broken down into smaller, reusable components (e.g., `InputSection`, `OutputDock`, `ReportModal`), improving code readability and maintainability.
  - **Directory Restructuring:** A dedicated and organized directory structure was created for the module, separating components, configurations, and assets.
  - **Centralized Configuration:** Static data such as dropdown options and validation rules were moved to dedicated configuration files, decoupling them from the UI logic.
  - **Improved State and API Management:** Integration with backend services was streamlined by centralizing API calls and state management through React's Context API.

- **Documentation and Testing:** The entire workflow was rigorously tested to identify and handle edge cases. The implementation and architectural changes were documented to facilitate future development and easier onboarding for new contributors.

5.2 Skills Developed

This fellowship has been instrumental in honing a diverse range of technical and professional skills:

Technical Skills

- **Full-Stack Web Development:** Gained hands-on experience in developing a full-stack web application using **React.js** for the frontend and **Python** (with frameworks like Django) for the backend.
- **Frontend Technologies:** Developed proficiency in modern frontend development, including component-based architecture, state management (using React Hooks and Context API), and UI/UX design using **Ant Design**.
- **Backend API Development:** Acquired skills in creating and integrating RESTful APIs for data processing and client-server communication.
- **Software Architecture:** Gained a deep understanding of modular design, separation of concerns, and code refactoring for scalability and maintainability.
- **Version Control:** Enhanced proficiency in using **Git** for collaborative development.
- **3D CAD Integration:** Learned to integrate **FreeCAD** within a web application to render complex 3D models.
- **Report Generation:** Programmatically generated structured and professional reports using LaTeX.

Professional Skills

- **Problem-Solving:** Developed the ability to break down complex technical requirements into manageable tasks and implement efficient solutions.
- **Project Management:** Gained experience managing development timelines from analysis to deployment.
- **Adaptability:** Learned to navigate and contribute effectively to a large and existing codebase.
- **Attention to Detail:** Practiced writing clean, maintainable code with a focus on handling edge cases and improving test coverage.

- **Collaboration and Communication:** Improved the ability to work in a team, understand project goals, and contribute to open-source development.

Chapter A

Appendix

A.1 Work Reports

Internship Work Report

Name:	Mohit Rana
Project:	Osdag
Internship:	FOSSEE Summer Fellowship 2025

DATE	DAY	TASK	Hours Worked
16-May-2025	Friday	Fellowship orientation. Installed Ubuntu and set up the basic development toolchain (VS Code, Git).	6
17-May-2025	Saturday	Cloned the Osdag web repository and all related submodules. Began installing project dependencies.	5
18-May-2025	Sunday	Self-study: Reviewed project documentation and explored the existing Osdag application features.	4
19-May-2025	Monday	Resolved dependency issues. Successfully ran the frontend (npm start) and backend (runserver) servers.	7
20-May-2025	Tuesday	Conducted a thorough analysis of the Osdag project's architectural flow and overall directory structure.	7
21-May-2025	Wednesday	Studied the existing modules to understand the design patterns and coding standards.	6
22-May-2025	Thursday	Met with mentor to discuss the plan for the "Struts in Trusses" module and clarify initial questions.	5
23-May-2025	Friday	Began backend development by outlining the calculation logic for struts in data/design_type/compression_member/compression.py.	7
24-May-2025	Saturday	Implemented the core design calculation logic for strut members based on IS 800:2007 guidelines.	6
25-May-2025	Sunday	Weekend/Self-study: Researched Python integration with FreeCAD for CAD model generation.	4
26-May-2025	Monday	Developed the backend API endpoint in web_api/struts_in_trusses_api.py to handle requests from the frontend.	8
27-May-2025	Tuesday	Integrated backend logic for session state management and input validation to ensure data integrity.	7
28-May-2025	Wednesday	Developed the API logic to interface with FreeCAD for generating the 3D CAD model of the designed strut.	8
29-May-2025	Thursday	Implemented the backend logic in reportGenerator_latex.py to compile design data for the report.	7
30-May-2025	Friday	Started frontend development by creating the main component file StrutsInTrusses.jsx and ModuleState context.	7
31-May-2025	Saturday	Built the user interface for the input section, including dropdowns and fields for design parameters.	6
01-Jun-2025	Sunday	Weekend/Self-study: Reviewed React component composition.	4
02-Jun-2025	Monday	Implemented the OutputDock and Logs components to display results from the backend.	7
03-Jun-2025	Tuesday	Created supporting frontend components like DropdownMenu, DesignPrefSections, and CustomSectionModal.	8
04-Jun-2025	Wednesday	Connected the frontend to the backend API to trigger the design calculation and retrieve results.	8
05-Jun-2025	Thursday	Integrated the CAD viewer to render the 3D model received from the FreeCAD API.	7
06-Jun-2025	Friday	Integrated the backend report generation API	7
07-Jun-2025	Saturday	Conducted full end-to-end testing of the module's workflow and performed initial bug fixing.	6

Internship Work Report

Name:		Mohit Rana	
Project:		Osdag	
Internship:		FOSSEE Summer Fellowship 2025	
08-Jun-2025	Sunday	Weekend/Self-study.	0
09-Jun-2025	Monday	Documented the initial module implementation and held a review session with the mentor.	6
10-Jun-2025	Tuesday	Analyzed the monolithic structure of the module and planned the refactoring to improve modularity.	7
11-Jun-2025	Wednesday	Created the new, organized directory structure under osdagclient/src/modules/compressionMember/strutsInTrusses/ .	7
12-Jun-2025	Thursday	Began refactoring by creating a separate InputSection.jsx and migrating input logic.	8
13-Jun-2025	Friday	Extracted the results display logic into a dedicated OutputDock.jsx component.	7
14-Jun-2025	Saturday	Migrated the report generation modal into its own ReportModal.jsx component.	6
15-Jun-2025	Sunday	Weekend/Self-study.	0
16-Jun-2025	Monday	Moved all static configurations (dropdown options, defaults) to a dedicated configs/strutsInTrussesConfig.js file.	7
17-Jun-2025	Tuesday	Refactored API calls (createSession, createDesign) to use the centralized ModuleContext.	8
18-Jun-2025	Wednesday	Improved UI/UX by applying consistent styling with Ant Design across all refactored components.	7
19-Jun-2025	Thursday	Moved all relevant images and diagrams to the new assets/ folder and ensured dynamic UI updates.	6
20-Jun-2025	Friday	Performed a complete workflow validation of the refactored module to ensure all functionality was preserved.	7
21-Jun-2025	Saturday	Debugged minor issues from refactoring and tested responsiveness of the UI.	6
22-Jun-2025	Sunday	Weekend/Self-study.	0
23-Jun-2025	Monday	Updated all project documentation to reflect the new modular architecture and code structure.	7
24-Jun-2025	Tuesday	Held a final code review session with the mentor for the refactored module.	5
25-Jun-2025	Wednesday	Began drafting the final internship report, summarizing Task 1 and Task 2.	7
26-Jun-2025	Thursday	Completed the "Conclusions" and "Skills Developed" sections of the report.	6
27-Jun-2025	Friday	Submitted the first draft of the final report to the mentor and began preparing the final presentation.	7
28-Jun-2025	Saturday	Incorporated mentor feedback into the final report and practiced the presentation.	6
29-Jun-2025	Sunday	Final code cleanup and ensured all branches were merged and pushed to the repository.	5
30-Jun-2025	Monday	Presented the final work and submitted all deliverables, and completed exit formalities.	8

Bibliography

[1] Siddhartha Ghosh, Danish Ansari, Ajmal Babu Mahasrankintakam, Dharma Teja Nuli, Reshma Konjari, M. Swathi, and Subhrajit Dutta. Osdag: A Software for Structural Steel Design Using IS 800:2007. In Sondipon Adhikari, Anjan Dutta, and Satyabrata Choudhury, editors, *Advances in Structural Technologies*, volume 81 of *Lecture Notes in Civil Engineering*, pages 219–231, Singapore, 2021. Springer Singapore.

[2] FOSSEE Project. FOSSEE News - January 2018, vol 1 issue 3. Accessed: 2024-12-05.

[3] FOSSEE Project. Osdag website. Accessed: 2024-12-05.