

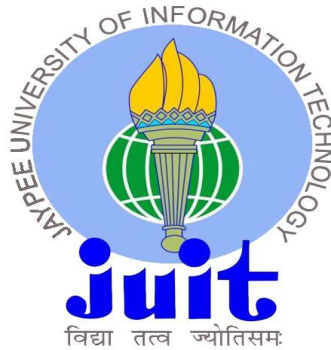
Student Behaviour Management

A major project report submitted in partial fulfillment of the requirement
for the award of degree of

Bachelor of Technology
in
Computer Science & Engineering

Submitted by
Manas Bajpai 211477

Under the guidance & supervision of
Ms. Seema Rani & Mr. Gaurav Negi



**Department of Computer Science & Engineering and
Information Technology**

**Jaypee University of Information Technology,
Waknaghat, Solan - 173234 (India)**

May 2025

SUPERVISOR'S CERTIFICATE

We hereby declare that the work presented in this major project report entitled '**Student Behaviour Management**', submitted in partial fulfilment of the requirements for the award of the degree of **Bachelor of Technology in Computer Science & Engineering**, in the Department of Computer Science & Engineering and Information Technology, Jaypee University of Information Technology, Waknaghat, is an authentic record of our work carried out during the period from July 2024 to May 2025 under the supervision of **Ms. Seema Rani** and **Mr. Gaurav Negi**.

We further declare that the matter embodied in this report has not been submitted for the award of any other degree or diploma at any other university or institution.

Name: Manas Bajpai

Roll No.: 211477

Date:

This is to certify that the above statement made by the candidates is true to the best of my knowledge.

Supervisor Name: Ms. Seema Rani Supervisor Name: Mr. Gaurav Negi

Date: Designation: Asst. Prof. (Grade-1) Designation: Asst. Prof. (Grade-1)

Place: Department: CSE & IT Department: CSE & IT

CANDIDATE'S DECLARATION

We hereby declare that the work presented in this major project report entitled '**Student Behaviour Management**', submitted in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology in Computer Science & Engineering**, in the Department of Computer Science & Engineering and Information Technology, Jaypee University of Information Technology, Waknaghat, is an authentic record of our own work carried out during the period from July 2024 to December 2024 under the supervision of **Ms. Seema Rani** and **Mr. Gaurav Negi**.

We further declare that the matter embodied in this report has not been submitted for the award of any other degree or diploma at any other university or institution.

Name: Manas Bajpai

Roll No.: 211477

Date:

This is to certify that the above statement made by the candidates is true to the best of my knowledge.

	Supervisor Name: Ms. Seema Rani	Supervisor Name: Mr. Gaurav Negi
Date:	Designation: Asst. Prof. (Grade-1)	Designation: Asst. Prof. (Grade-1)
Place:	Department: CSE & IT	Department: CSE & IT

ACKNOWLEDGEMENT

Firstly, we express our heartfelt thanks and gratitude to the Almighty for His divine blessings, which made it possible for us to successfully complete this project.

We are deeply grateful and profoundly indebted to our supervisors, Ms. Seema Rani and Mr. Gaurav Negi from the Department of CSE & IT, Jaypee University of Information Technology, Wakhnaghat. Their deep knowledge and keen interest in the field of Educational Technology, combined with their unwavering support, scholarly guidance, continual encouragement, and constructive criticism, were instrumental in the successful completion of our project. We sincerely appreciate their patience in reviewing multiple drafts and their valuable advice at every stage.

We would also like to extend our heartfelt thanks once again to Ms. Seema Rani and Mr. Gaurav Negi for their generous assistance and constant motivation throughout the course of this project.

We are equally thankful to all those who supported us directly or indirectly in making this project a success. In this regard, we would like to acknowledge the cooperation and support of the faculty members and the non-teaching staff, whose timely help and encouragement greatly facilitated our work.

Lastly, we are truly thankful to our parents for their unwavering support and encouragement throughout this journey.

Manas Bajpai 211477

TABLE OF CONTENTS

Supervisor's Certificate.....	I
Candidate's Declaration.....	II
Acknowledgement.....	III
List of Tables.....	IV
List of Figures.....	VI
List of Abbreviations.....	VII
Abstract.....	VIII
Chapter(s)	
1 Introduction.....	1
1.1 Introduction.....	1
1.2 Problem Statement.....	2
1.3 Objectives.....	3
1.4 Significance and Motivation of the Project Work.....	4
1.5 Organisation of Project Report.....	5
2 Literature Survey.....	7
2.1 Overview of Relevant Literature.....	7
2.2 Key Gaps in Literature.....	11
3 System Development.....	12
3.1 Requirements and Analysis.....	12
3.2 Project Design and Architecture.....	14
3.3 Data Preparation.....	16
3.4 Implementation.....	17
3.5 Key Challenges.....	19
4 Testing.....	21
4.1 Testing Strategy.....	21
4.2 Test Cases and Outcomes.....	25
5 Results and Evaluation.....	29
5.1 Results.....	29

5.2	Comparison with Existing Solutions.....	31
6	Conclusions and Future Scope.....	34
6.1	Conclusion.....	34
6.2	Future Scope.....	35
	References.....	42
	Appendix.....	44

LIST OF TABLES

Table Name	Page No.
Table 2.1. List of documentations	7-10
Table 4.1 List of test cases and results	25-26
Table 4.2 List of bugs and their solutions	26-27
Table 5.1 Comparison of features to pre-existing solutions	31

LIST OF FIGURES

Figure Name	Page No.
Fig 3.1 Flowchart of the application	16
Fig. 3.2 Behaviour Module Homepage Screenshot	18
Fig. 3.3 Single Incident Screenshot	19
Fig 3.4 Teacher Notes Screenshot	20
Fig. 4.1 Sentry Screenshot	24

LIST OF ABBREVIATIONS

JS	JavaScript
HTML	Hyper Text Markup Language
CSS	Cascading Style Sheets
SASS	Syntactically Awesome Style Sheets
JSX	JavaScript XML
XML	Extensible Markup Language
AWS	Amazon Web Services
GraphQL	Graph Query Language
SQL	Structured Query Language
CI/CD	Continuous Integration/Continuous Development
UI	User Interface
UX	User Experience
QA	Quality Assurance
WCAG	Web Content Accessibility Guidelines

ABSTRACT

The Student Behaviour Management System is a modern, unified platform designed to simplify how schools track, record, and manage student behaviour. Traditional methods like manual logging often lead to disorganization, inconsistency, and overlooked details—especially in the case of minor incidents or positive behaviours. Our system addresses these issues by offering a real-time, structured, and collaborative framework for behaviour tracking. Educators can categorize behaviour as positive, neutral, or negative and follow each case through clear stages: draft, published, and closed. Features such as collaborative tagging, built-in data verification, and role-based access control further enhance accuracy, accountability, and teamwork.

Built using modern web technologies like ReactJS [1], Apollo Client, and GraphQL, the system delivers strong performance and a responsive, user-friendly interface across devices. It is scalable, easy to adopt, and supports timely interventions by providing data-driven insights into behavioural trends. Initial deployments have shown improvements in both the precision of incident reporting and the effectiveness of staff collaboration. Overall, the system addresses critical gaps in traditional behaviour management and helps create a more positive, well-regulated learning environment for students.

CHAPTER 1: INTRODUCTION

1.1 INTRODUCTION

This project fills important holes in conventional behaviour control techniques in all academic institutions, such as schools and universities, and provides a scalable system that can be adjusted to any academic situation, marking a great advancement in educational technology. The management of this application promotes a child's overall development across all domains and reduces, if not removes, the communication gap that is unconsciously developed between teachers, parents, and administrative personnel by offering a one-stop platform for all sorts of behaviour control and management.

Traditional manual record-keeping and improper teacher communication are two extremely inefficient and ineffective behaviour management techniques. These methods are a frequent result of several missed intervention opportunities, high amounts of inaccurate recording, and several unreported instances. By offering a one-stop integrated platform for documenting and tracking student behaviour like this, from positive and commendable events like academic awards to negative ones like bullying or misconduct, the behaviour management system aims to solve and rectify these issues.

Numerous sophisticated features, such as instructor collaboration, real-time reporting, and a state-based incident management workflow (draft, published, and resolved), are all integrated into the project. Besides automating behaviour control procedures, the robust structure and friendly interface of the system enable teachers to make informed decisions in the best interest of their students. It is necessary for today's schools as it utilises advanced web technology powered by artificial intelligence to offer an easily accessible, scalable solution.

The behaviour management project also positions itself as a cornerstone for schools aiming to enhance their behaviour management practices in today's day and age of digitalisation, where decisions based on data are increasingly playing a crucial role day by day. Improved multiple student outcomes and more educator-student interactions become possible because of optimised documentation, improved cooperation, and ensuring transparency.

1.2 PROBLEM STATEMENT

Lack of an organised, systematic, and centralised method of student behaviour management continues to present a variety of recurrent challenges in educational institutions at all levels. This lack of structure and uniformity has increasingly become inefficient. for how the school ecosystem captures, shares, and uses the behavioural data:

- **Lack of Documentation**

Behavioural incidents overall were radical in enough, especially small impact but valuable, such as adding value to a peer in the form of a student helping a peer (positive), or having bad intent in the form of misconduct such as slapping another student (negative) – is far too often not observed or not recorded. This leads to fractured or inadequate behavioural logs, therefore compromising the capability to perform fair and comprehensive behavioural evaluation on time.

- **Inefficient Communication**

School instructors often operate in isolation because of the lack of a united digital platform that supports real-time collaboration. This split reduces the capacity to coordinate behavioural interventions, or share insights about recurring problems, and jointly strive to improve students' conduct through mutual understanding.

- **Subjectivity and Inconsistency**

In the absence of such standardised procedures and tools, behavioural reporting often relies on individual judgment, which introduces a high degree of subjectivity. This inconsistency not only affects the quality of documentation but also leads to varying responses for similar incidents, which can be confusing for both staff and students.

- **Missed Intervention Opportunities**

When incidents — especially early signs of problematic behaviour — are not logged or escalated promptly, schools lose valuable opportunities to provide early intervention. This can result in the escalation of avoidable behavioural issues or the under-recognition of positive actions that deserve reinforcement.

- **Time-Consuming Processes**

The traditional approach of documenting student behaviour manually, whether through paper records or disjointed digital systems, proves to be highly inefficient. It places an additional administrative burden on teachers, thereby reducing the time and energy available for core academic responsibilities and student engagement.

Together, these limitations underscore the critical need for a centralised, intelligent, and collaborative system that supports the effective management of student behaviour. Such a system should not only streamline documentation and reporting but also foster better communication among staff, reduce inconsistencies, and enable timely and meaningful interventions that contribute to the holistic development of students.

1.3 OBJECTIVES

Closely aligned towards improving efficiency are the core focuses of this project. Consistency and student behaviour's collaborative nature of management in educational institutions.

- **Real-Time Incident Reporting**

Create a singular platform by which educators can record behavioural incidents instantly and from anywhere. This also ensures that there is timely documentation and makes it possible for quicker administrative and pedagogical responses.

- **Collaborative Management**

Make it possible for educators to tag multiple teachers and students in one case. This tends to create a more holistic outlook of behavioural contexts and promotes mutual responsibility.

- **Workflow Automation**

Adopt a structured incident lifecycle that includes three separate states, namely Draft, Published, and Resolved. This workflow automates incident management, which guarantees appropriate documentation, supports smooth movement from the report to the final.

- **Data Quality and Consistency**

Introduce automated validation and testing mechanisms in order for data to be of high integrity. This ensures completeness, accuracy, and absence of all logged events, redundancy or conflicting information.

- **Improved Accessibility**

Create the system with responsiveness and accessibility in mind, supporting, ranging from desktop computers to tablets and smartphones. Features such as dynamic interfaces and keyboard navigation add usability to every user.

- **Scalability and Integration**

Design an elastic architecture that can accommodate bulk of behavioural data in it, while having effortless integration with pre-existing school management systems and infrastructure.

By fulfilling these objectives, the project aims to evolve how schools document, monitor, and respond to student behaviour, establishing a more efficient, data-driven, and collaborative framework for behavioural oversight.

1.4 SIGNIFICANCE AND MOTIVATION OF THE PROJECT WORK

This project is significant because it fills in very important holes in conventional behaviour management techniques. Schools have a major impact on how pupils develop socially and emotionally, in addition to academically. A successful behaviour management system guarantees the following, as listed below:

- **Acknowledgement of Positive Behaviours**

This approach promotes a culture of motivation and encouragement by recording accomplishments and positive behaviour that would boost the morale of the students and encourage them to behave well in social environments.

- **Prompt Reaction to Adverse Conduct**

A safer and more favourable learning environment can be achieved by the early detection and resolution of misconduct or difficulties. This ensures the growth of positivity and the decline of negativity in academics.

- **Better Teacher Collaboration**

The technology makes it easier for teachers to communicate with one another, which makes it possible to handle student behaviour as a group.

- **Simplified Documentation**

Automation in many repetitive areas greatly lessens the amount of manual labour required to maintain accurate, consistent, and readily available records.

- **Data-Driven Insights**

Teachers can make well-informed and policy decisions by identifying the trends and patterns in the data with the use of comprehensive behaviour data of all the incidents.

The need to provide schools with resources that streamline operations, enhance student performance, and promote a more cooperative atmosphere is what spurred this effort. By utilising this technology, the system not only tackles present issues but also establishes the groundwork for upcoming advancements in the field of education.

1.5 ORGANIZATION OF PROJECT REPORT

This report is organised into six comprehensive chapters, where each one addresses a critical dimension of the project in order to offer a clear and structured narrative:

- **Introduction**

This provides the foundation by spelling out all the main objectives, purpose and relevance of the project. It also showcases all of the current issues as they exist in traditional behaviour management practices and demonstrates how one needs to have a unified and digital solution.

- **Literature Review**

Reviews current systems and academic work in the field of student behaviour tracking. This chapter defines technological advancement, exposes limits on current approaches, and situates the distinctive contribution to the field made by the project.

- **System Development**

Examines the technical framework of the project – system architecture, design, tools, and methodology of development. It also sheds light on the practical problems during implementation and the way these were dealt with.

- **Testing**

It describes the testing strategies, environments, and framework used to check the system's performance and reliability. This ranges from functional to non-functional testing, which guarantees that the solution is able to achieve its objectives.

- **Results and Evaluation**

Describes what the project outcome has been and how well it solved the problems identified in the evaluation. In this chapter, comparisons to other tools are made, and how the system has a tangible impact in practice is exhibited using metrics, user feedback and anecdotes.

- **Conclusion and Future Scope**

Outlines the project's broad contributions and learnings. It also describes possible improvements, scalability opportunities, and how the system may progressively develop to embrace more use cases or greater integration within the school ecosystems.

CHAPTER 2: LITERATURE SURVEY

2.1 OVERVIEW OF RELEVANT LITERATURE

Systems for managing student behaviour are becoming crucial resources for creating supportive learning environments. These solutions simplify behaviour tracking and management by utilizing web technology improvements, real-time collaborative frameworks, and accessibility requirements. The main literature from all available online sources is reviewed in this chapter, with an emphasis on behaviour management-related technologies, approaches, and platforms.

S. NO.	Source (Docu- mentations/ Online Resource)	Tools/ Techniques	Key Findings/ Results	Limitations/ Gaps Identified
1.	Organization's Internal Documentation	React, Redux, Git, Apollo Client, GraphQL, GitHub, Version Control	All the necessary documents provided for a headstart into contributing to the existing platform are provided in the organization's internal documentation. This ranges from data on the frontend, the backend, version control, cross-platform development and also dev-ops, which are usually based on CI/CD methods, or continuous integration/continuous development. Also	As the documentation is non-disclosable and internal to the organization, it cannot be shared with people outside the organization. Also, the documentation focuses on the required use cases of the technologies in the application, missing out on potential better use cases.

			includes the API for the design system that is used in the product.	
2.	ReactJS Documentation	ReactJS, State Management	React [1] is a JavaScript library that enables us to build scalable applications using JavaScript. It makes the process of managing the state easier due to the in-built functions and the architecture that it offers. It provides a robust library for building interactive UIs with efficient state management, crucial for dynamic behaviour management systems.	React is only a frontend library, so it does not provide built-in solutions for backend integration or real-time collaboration. It generally requires third-party tools like Apollo Client for complex applications.
3.	React Native Documentation	React Native, Cross-Platform Development	React Native, an upgraded version of React, enables cross-platform mobile application development using a single codebase, significantly reducing development and learning time and effort.	As the code is not written in the platform's native language, it is unable to offer great performance for compute-intensive tasks compared to native apps. It depends on the integration of third-party libraries for advanced features.

4.	Apollo GraphQL Docs	Apollo Client, GraphQL	Apollo GraphQL [2] is a very resourceful tool that enables extremely efficient data fetching and state synchronization between the front end and the back end using GraphQL. It also supports real-time updates via GraphQL subscriptions.	The applications' performance can take a hit and degrade with poorly optimized queries or considerably larger datasets. It also requires some additional setup to handle several error states and data consistency.
5.	PostgreSQL Document	Relational Databases	A relational database, PostgreSQL [3] provides an efficient database solution for managing relational data, that also supports usage of complex queries and maintains data integrity ranging from small applications, all the way to large-scale applications.	While it is effective for relational data, managing scalability in real-time applications with high concurrent loads may require additional caching layers or database optimizations, that users generally ignore which is considered a bad practice.
6.	Web Content Accessibility Guidelines (WCAG)	Accessibility Standards	These guidelines offer a really comprehensive guidelines for making applications accessible to users with disabilities. They are essential for keyboard navigation and usability enhancements across the platform, as there are cases where the	Offers comprehensive guidelines for making applications accessible to users with disabilities. Essential for keyboard navigation and usability enhancements.

			user might not be able to use the mouse or wishes to navigate using just a keyboard.	
7.	AWS Cloud White Paper	Cloud Storage, Scalability	It provides scalable storage solutions for managing large datasets and ensuring data availability across all regions across the globe, which is crucial for educational institutions with an international user base.	Higher costs for small-scale institutions. Integration also has challenges with the on-premise data systems.
8.	Jira Documentation	Task Management	Provides insights into managing incidents with state-based workflows, emphasising draft, published, and resolved states.	Limited real-time features in standard configurations. Requires plugins or customisations for specific needs in education management.

Table 2.1: A table with a list of detailed resources that were referred to before contributing to the development of the web and mobile applications.

2.2 KEY GAPS IN LITERATURE

The following holes still need to be filled, even though the examined frameworks and technologies provide insightful information about how to create student behaviour management systems:

- **Real-Time Scalability**

While existing systems like Firebase and Apollo Client provide real-time capabilities, they need to be heavily optimised to support major institutions' high concurrent user loads.

- **Cross-Platform Usability**

It's still difficult to guarantee smooth user experiences on mobile and online platforms while upholding accessibility guidelines.

- **Incident Categorisation and Reporting**

Because automated methods for classifying and analysing behavioural occurrences are still in their infancy, thorough reports must be created by hand.

- **Cost-effectiveness**

Smaller schools with fewer resources might not be able to afford scalable solutions, especially cloud-based systems like AWS [8].

- **Integration with Current Systems**

It can be difficult to integrate new technologies with outdated school management systems, which could result in incompatible or redundant procedures.

CHAPTER 3: SYSTEM DEVELOPMENT

3.1 REQUIREMENTS AND ANALYSIS

The Student Behaviour Management System offers a reliable, scalable, and intuitive solution to problems with conventional behaviour-tracking techniques. There were two types of system requirements: functional and non-functional.

3.1.1 Functional Requirements

- **Quick Behaviour Logging**

Teachers can easily record incidents as they happen and label them as *positive*, *neutral*, or *negative* based on the situation. This helps build a complete behaviour history for each student.

- **Team Collaboration**

One incident might involve more than one teacher or student. So, the system allows tagging multiple people in a single report to keep everyone involved and in the loop.

- **Incident Workflow System**

Each incident goes through three stages — *Draft*, *Published*, and *Resolved*. This makes it easy to manage reports step-by-step, from writing to final action.

- **Accurate and Clean Data**

Every entry goes through checks to make sure the information is clear, correct, and complete. This avoids confusion later when someone needs to refer back to it.

- **Search and Filter Options**

Teachers can quickly find past incidents using filters like tags, categories, dates, or even names. This saves time and avoids digging through long lists.

- **Real-Time Notifications**

Whenever something new is added or an incident is updated, the right people get notified instantly. This keeps the communication clear and fast.

3.1.2 Non-Functional Requirements

- **Scalability**

The system should handle a large number of users and big amounts of data at the same time, without slowing down or crashing.

- **Accessibility**

The platform should be easy to use for everyone, including people with disabilities. It must support screen readers and keyboard navigation, and follow WCAG [4] accessibility rules.

- **Cross-Platform Compatibility**

The application should work smoothly on all major devices — including phones, tablets, and desktop computers — without layout or performance issues.

- **Data Security**

Sensitive information must be protected using role-based access controls and secure storage methods, so that only the right people can view or edit certain data.

- **Performance Optimization**

Tasks like searching or saving incidents should be quick. The system should respond within 400 milliseconds to keep the user experience fast and smooth.

3.2 PROJECT DESIGN AND ARCHITECTURE

The system is designed using a modular approach, ensuring scalability and maintainability. The architecture is divided into three main layers:

3.2.1 Frontend

- Framework: ReactJS [1], SASS, Apollo Client, GraphQL [2]
- UI/UX Design: Implemented with an in-built design system which is layered on top of Ant-design for responsive and dynamic interfaces.
- State Management: Utilized Redux state management's API for managing user states and application data.
- Accessibility Features: Included keyboard navigation, focus management, and WCAG-compliant components.

3.2.2 Backend

- API: GraphQL [2], with Apollo Server handling queries and mutations.
- Data Management: Ensured efficient data fetching through optimized GraphQL queries and subscriptions.
- Database: Used PostgreSQL [3] to manage and maintain the database.
- Workflow Automation: Incorporated a logic engine to manage the state transitions of incidents.

3.2.3 Database

- Database Management System: PostgreSQL [3]
- Schema Design:
 - Tables for users, incidents, and tags.
 - Relationships are defined to associate incidents with multiple users (teachers and students).
 - Indexing for faster retrieval of incident data.
- Data Integrity: Implemented constraints and triggers to enforce consistency.

3.2.4 System Architecture

The system follows a three-tier architecture:

- Presentation Layer: ReactJS [1] for the client-side interface.

- Business Logic Layer: Apollo Server and custom logic for handling data operations and workflows.
- Data Layer: PostgreSQL [3] for relational data management.

3.2.5 Frontend Folder Structure

The folder structure of the front end is incorporated into the application as follows.

- Routes: This folder contains the child routes of the said route.
- Modules: This folder contains the core JS logic of the route and its children's routes, which typically range from common functions, utils, graphql queries, mutations, fragments, etc.
- Components: This folder contains the files that contain JSX, that is the HTML in the form of JS that is rendered on the webpage. These files contain other JS functions that are used in React to build the required components.

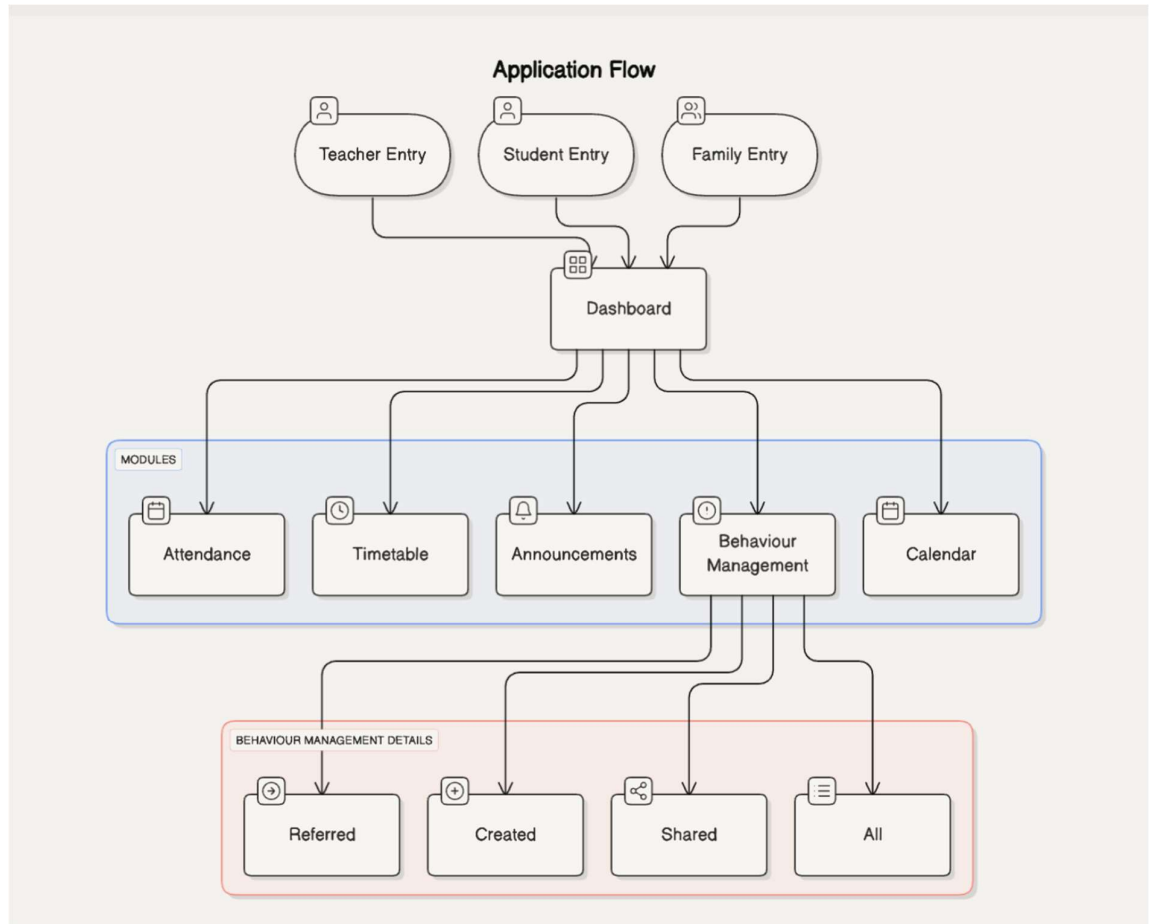


Fig 3.1: An illustration of the application, focused on the behaviour management module. After logging into the application as either a parent, teacher or student, the user can view all the available modules. One of them is the behaviour module.

3.3 DATA PREPARATION

The system's success relies on well-structured and accurate data. Key steps in data preparation included:

1. Data Cleaning:
 - a. Removed redundant records from imported datasets.
 - b. Standardised format for dates, names, and tags.
2. Data Validation:

- a. Conducted checks to ensure the completeness and correctness of incident records.
 - b. Tested edge cases, such as duplicate incidents and overlapping tags.
- 3. Initial Data Seeding:
 - a. Populated the database with sample data, including predefined users, tags, and test incidents, for system testing and demonstration.

3.4 IMPLEMENTATION

3.4.1 Key Modules:

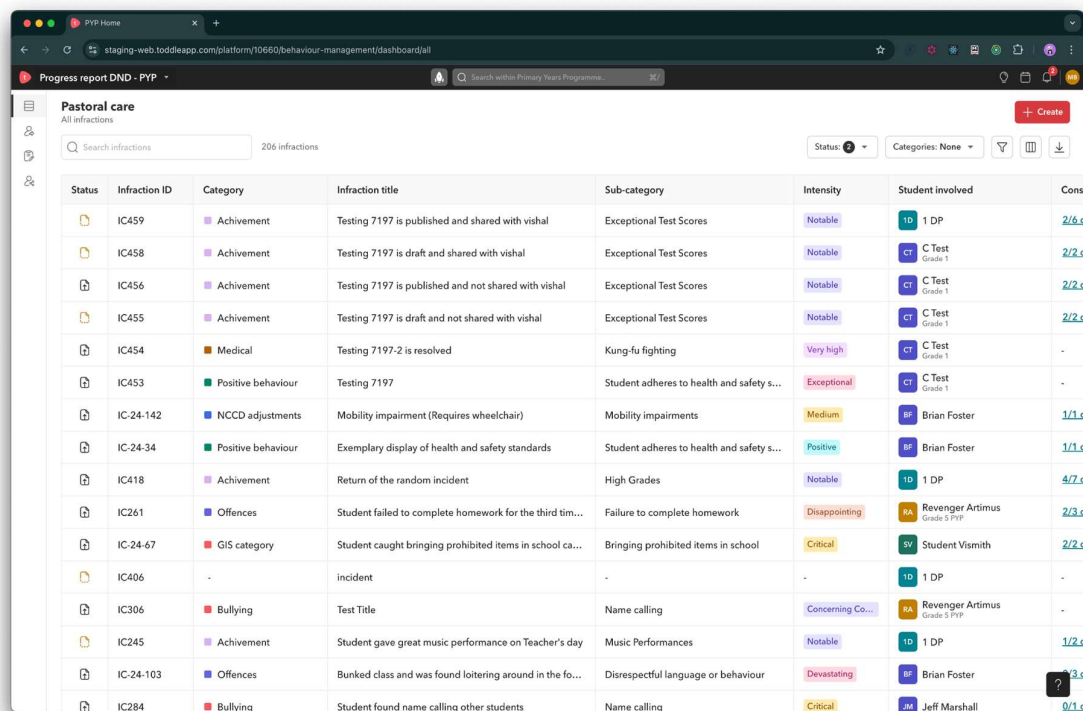
- 1. Incident Management:
 - a. Teachers can log incidents with details like description, category (positive, neutral, negative), and tags.
 - b. State transitions (Draft -> Published -> Resolved) are managed through intuitive workflows.
- 2. Tagging and Collaboration:
 - a. Tags allow teachers to involve other educators and students in an incident.
 - b. Real-time notifications for tagged users.
- 3. Search and Filter:
 - a. Users can search incidents by tags, categories, or dates.
 - b. Filters enable viewing unresolved incidents or incidents in a specific time range.
- 4. Student Profile:
 - a. Users can view the student profile of the involved student, which contains relevant information regarding the student. These fields include personal information, behaviour summary, teacher notes and student alerts.
 - b. On this screen, users can view, add or remove relevant information regarding the student.

3.4.2 Algorithms:

1. Search Optimisation:
 - a. Used PostgreSQL [3]'s Full-Text Search feature for efficient query execution.
2. Data Consistency Checks:
 - a. Automated scripts to identify and resolve missing or inconsistent data.

3.4.3 Technologies Used:

1. Frontend: ReactJS [1], Material-UI, SASS
2. Backend: Apollo Server, GraphQL [2]
3. Database: PostgreSQL [3]
4. Other Tools: Git and GitHub for version control, JIRA for task tracking and management.



The screenshot displays a web application interface for 'Pastoral care' with a sidebar on the left and a main content area. The main area features a search bar, a table of infractions, and filters for status and categories. The table has columns for Status, Infraction ID, Category, Infraction title, Sub-category, Intensity, Student involved, and a 'Cons' column. The data is sorted by 'Intensity' in descending order.

Status	Infraction ID	Category	Infraction title	Sub-category	Intensity	Student involved	Cons
	IC459	Achievement	Testing 7197 is published and shared with vishal	Exceptional Test Scores	Notable	1D 1 DP	2/6
	IC458	Achievement	Testing 7197 is draft and shared with vishal	Exceptional Test Scores	Notable	CT C Test Grade 1	2/2
	IC456	Achievement	Testing 7197 is published and not shared with vishal	Exceptional Test Scores	Notable	CT C Test Grade 1	2/2
	IC455	Achievement	Testing 7197 is draft and not shared with vishal	Exceptional Test Scores	Notable	CT C Test Grade 1	2/2
	IC454	Medical	Testing 7197-2 is resolved	Kung-fu fighting	Very high	CT C Test Grade 1	-
	IC453	Positive behaviour	Testing 7197	Student adheres to health and safety s...	Exceptional	CT C Test Grade 1	-
	IC-24-142	NCCD adjustments	Mobility impairment (Requires wheelchair)	Mobility impairments	Medium	BF Brian Foster	1/1
	IC-24-34	Positive behaviour	Exemplary display of health and safety standards	Student adheres to health and safety s...	Positive	BF Brian Foster	1/1
	IC418	Achievement	Return of the random incident	High Grades	Notable	1D 1 DP	4/7
	IC261	Offences	Student failed to complete homework for the third tim...	Failure to complete homework	Disappointing	RA Revenger Artimus Grade 5 PYP	2/3
	IC-24-67	GIS category	Student caught bringing prohibited items in school ca...	Bringing prohibited items in school	Critical	SV Student Vismith	2/2
	IC406	-	incident	-	-	1D 1 DP	-
	IC306	Bullying	Test Title	Name calling	Concerning Co...	RA Revenger Artimus Grade 5 PYP	-
	IC245	Achievement	Student gave great music performance on Teacher's day	Music Performances	Notable	1D 1 DP	1/2
	IC-24-103	Offences	Bunked class and was found loitering around in the fo...	Disrespectful language or behaviour	Devastating	BF Brian Foster	2/3
	IC284	Bullying	Student found name calling other students	Name calling	Critical	JM Jeff Marshall	0/1

Fig 3.2: Illustration of the behaviour management module's homepage, showcasing a list of all the incidents of the current organisation. Here, the user can sort, filter download and view incidents according to the customizations provided by the application.

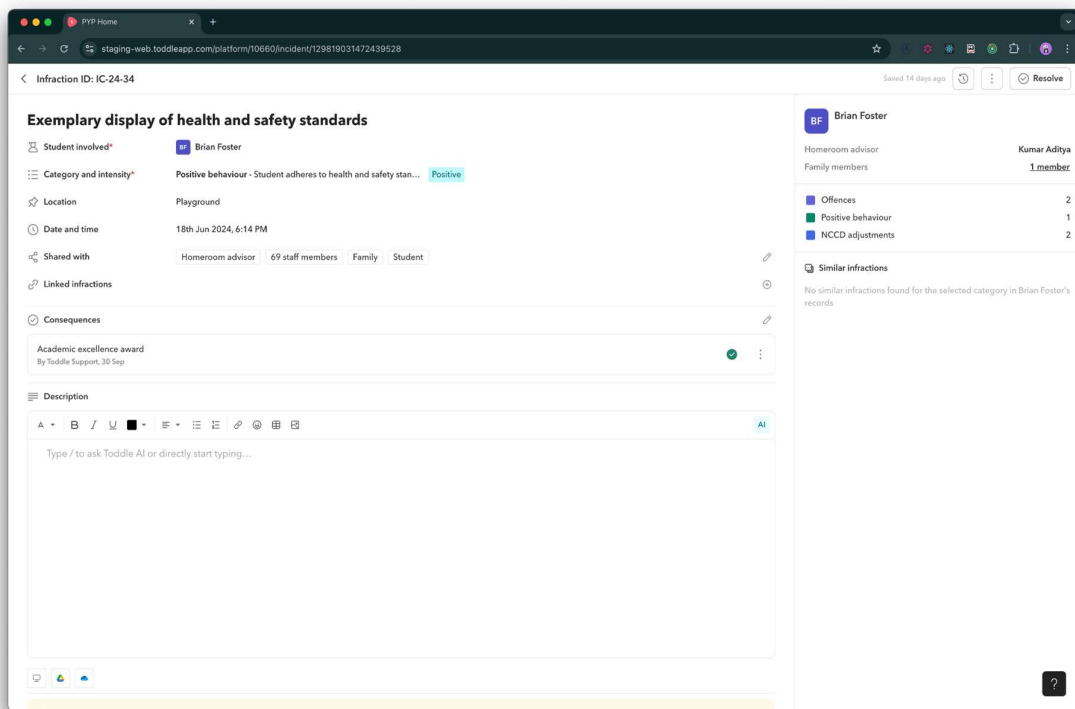


Fig 3.3: A single incident and all its fields, where the teacher can create an incident with required details. It consists of student involved, category and severity, location, time and date, shared with, linked incidents, description, attachments and confidential notes.

3.5 KEY CHALLENGES

3.5.1 Performance Bottlenecks

- Challenge: Handling real-time data updates for a large number of users.
- Solution: Optimized GraphQL [2] queries and introduced caching mechanisms to reduce server load.

3.5.2 Data Consistency

- Challenge: Ensuring data integrity when multiple users edit or view incidents simultaneously.

- Solution: Implemented transactional controls and locking mechanisms in PostgreSQL [3].

3.5.3 Accessibility Compliance

- Challenge: Adhering to WCAG [4] standards while maintaining aesthetic and functional designs.
- Solution: Conducted iterative testing and incorporated ARIA attributes for enhanced accessibility.

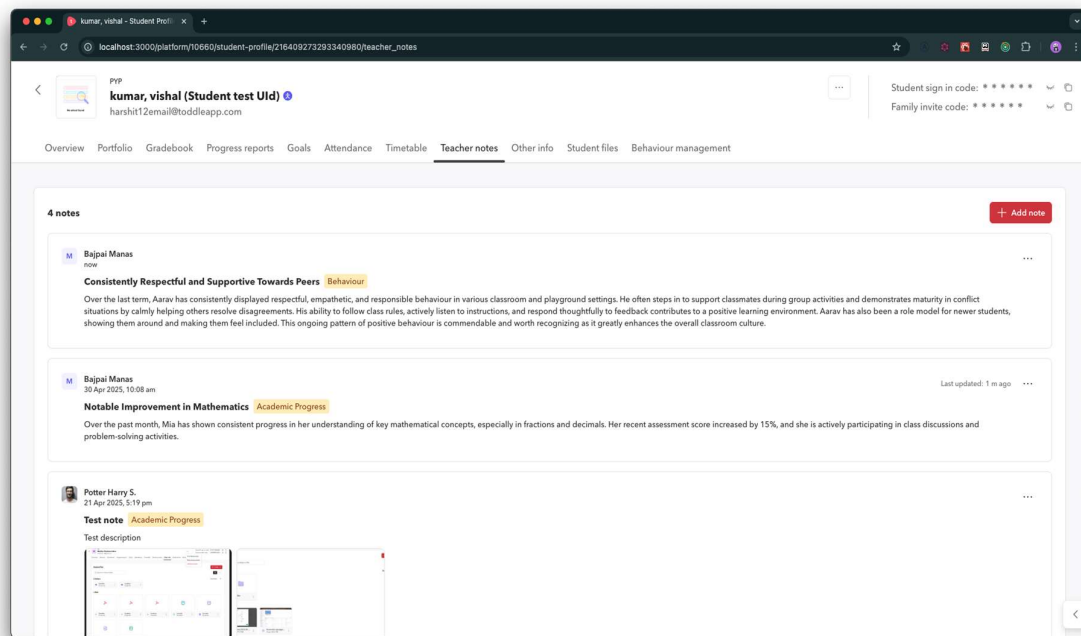


Fig 3.4: A view of the teacher notes page on the student profile, where several notes have been added by the teacher related to the student. Any staff who has the required permissions is eligible to make changes to the notes or maybe even add and delete them.

CHAPTER 4: TESTING

To make sure that the Student Behaviour Management System developed by us meets all the requirements of the user and their needs and operates just as we and they intended it to work, an extremely thorough testing is very much essential for the project. This project's testing procedure was carried out very methodically with the help of QA engineers who would primarily identify bugs and issues on the platform along with a planned workflow for finding, monitoring, and fixing problems.

4.1 Testing Strategy

The testing process of the platform was very structured and spread across multiple phases and environments, including the staging one, which is the first step where the successfully merged code goes to, then pre-production, where code is usually pushed after checks are successful checks in the staging platform, and some more essential checks are done, before finally adding it for the next release in the production environment, ensuring thorough validation of the system by using the three different environments to carefully pick out changes. Key aspects of the strategy included:

4.1.1 Quality Assurance (QA) Engineer Testing

- **QA engineers' role**

All system capabilities, including real-time event reporting, tagging, and state transitions, were always tested by committed QA engineers. They are dedicated to finding bugs and issues and reporting them to the developers, who will then take care of them and merge the required fixes for them in the application that will be tested again in staging, then in pre-production (pre-prod), and then finally go live in the production (prod) environment.

- **Method**

To find errors and confirm user routines, QA engineers used both automated and human testing. They have dedicated test cases written on the frontend as well as the backend, which they run whenever some new changes are merged into the platform. Also, they would manually test UI bugs themselves as they are logically correct but visually incorrect, which causes them to go unnoticed until seen by the naked eye. Tickets are created based on the type of bug encountered, then fixes for the same are assigned to the developers very systematically, which they would then work on, test and then get merged by their managers into the staging environment before further going into the releases.

4.1.2 Bug Tracking via Sentry

- **Sentry Integration**

To keep an eye on defects and performance problems on all platforms (prod, pre-prod, and staging), the system was integrated with Sentry [6], a real-time error-tracking tool. Whenever a logical bug is encountered, it may be a wrong API call, incorrect parameters, incorrect data passed, the unexpected value present, or anything similar, a sentry is generated on the tool and automatically assigned to the concerned team, who is the code owner for that particular piece of code. They would then check it out, reproduce it, find and finally resolve the bugs and remove them from the platform.

- **Issue Identification**

Sentry [8] would automatically record any bugs that would be found during testing or user contact, irrespective of the platform, which can be staging, pre-prod or production itself, including the impacted platform, the user's actions, and stack traces.

4.1.3 Task Management on Jira

- **Workflow**

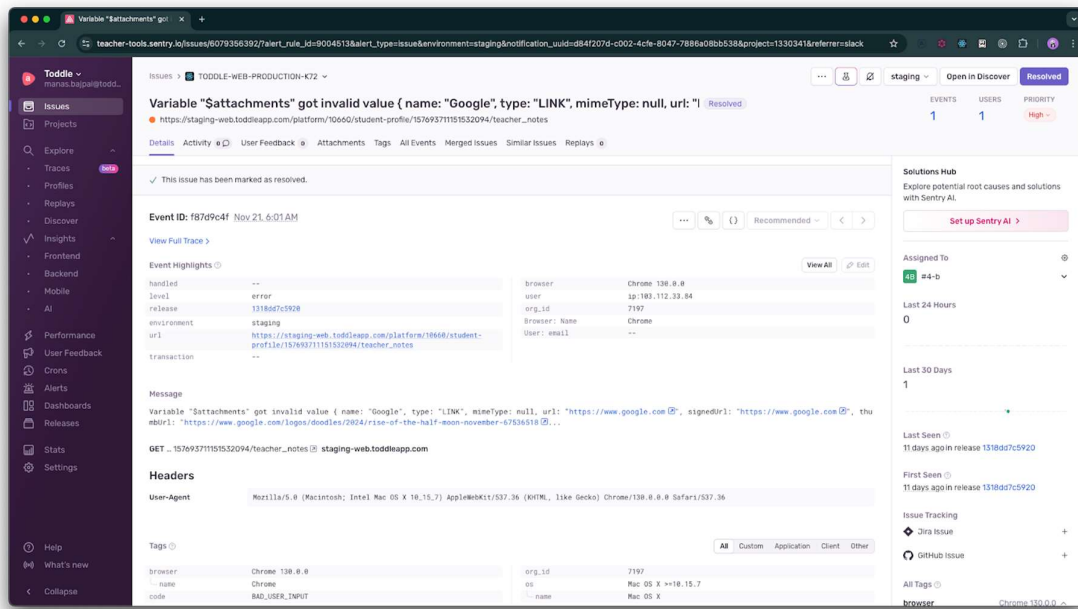
QA engineers would usually keep all the track of the list of issues that were reported in Sentry [6] and encountered by them and make the necessary Jira [7] tickets for them, which they would then be assigned by the managers to the developers according to where they are found, for example if it's a UI bug or a frontend bug then the frontend engineers would handle it, whereas if it is a backend bug, then the backend engineers would handle it. They would be recording the bug's specifics, how to reproduce it, and its severity in the ticket, which would then be checked by the developer before commencing the work to resolve the issue.

To resolve the problems, the development team prioritised significant bugs first and assigned tasks on Jira [9] to the respective developers who fixed it in the required amount of time. After the code is reviewed by the manager, they would then get it pushed to staging, where it will go through all the testing again. After all checks are successful, it is finally merged into production in the next release.

- **Cooperation**

To guarantee effective problem tracking and resolution, developers, QA engineers, and project managers worked together on Jira [7]. The cooperation was very valuable and was the direct result of good quality features on the website and good user experience with valuable feedback. Here, the manager generally creates tickets which help in tracking progress of the tasks that have been assigned to the developers and goes through multiple phases, including, not started, in progress, code review, staging (QA) and completed. They hold proof of the work done by the developers and the testing that is done by the QA engineers.

Fig 4.1: A screenshot of a Sentry that occurred across all environments for some bug. It contains a lot of data, including suspect commit (that may have caused the error to occur in



the first place), number of times it occurred in past 24 hours, 30 days, the variables provided, the response returned by the API, etc.

4.1.4 Bug Fix Deployment

- **Implementation of the Fix**

After the developers finished the required tasks, the fixes were tested in the pre-prod and staging environments to make sure they perfectly fixed all the problems that had occurred without creating new ones and breaking some other important functionality.

- **Merge Procedure**

The fixes were then deployed to the staging environment, then sent forward to the pre-prod before finally getting merged to prod and incorporated into the main codebase following validation.

4.2 TEST CASES AND OUTCOMES

4.2.1 Sample Test Cases

Test Case ID	Description	Expected Outcome	Result
TC-01	Log an incident with all mandatory fields filled.	Incident is successfully created and stored in the database.	Passed
TC-02	Attempt to log an incident without a category, which is required.	The system displays an error message and prevents submission. This is so that unnecessary incidents are not added by mistake by users and are properly validated before sending the API call with the parameters to the backend.	Passed
TC-02A	Attempt to log an incident without an optional field, not adding which does not make an impact or break anything in the codebase.	The system will not show a message this time, as this is an optional field and its input can be ignored. It does not matter if the user has added this or not.	Passed
TC-03	Search incidents by date range filter that is integrated into the filters field.	The frontend would retrieve all the incidents that had been recorded within that particular specified date range given by the user.	Passed
TC-04	Test state transition from “Draft” to “Published”.	The system updates the state and notifies relevant tagged users about the change so that everyone stays in the loop and gets to know where that incident is going in real time.	Passed

TC-05	Access the system using keyboard navigation only.	All features are accessible, adhering to WCAG [4] accessibility standards. Everything on the webpage should be accessible and usable without even touching the mouse, even once. This is done so that every user can use the application with ease.	Passed
-------	---	---	--------

Table 4.1 A table depicting some example test cases and their results.

4.2.2 Bug Fix Examples

Bug ID	Issue	Environment	Action Taken
BUG-101	Incident search fails for specific keywords.	Preprod	Adjusted the respective GraphQL query in order to handle the special characters in the search terms.
BUG-102	Notification system not triggering for tagging.	Staging	Identified missing backend logic and added appropriate GraphQL subscription for notifications.
BUG-103	The state transition button is unresponsive on mobile.	Prod	Fixed frontend event handling logic for touch inputs. Also, memoised certain functions in order to increase performance throughout the application.

BUG-104	The component is not showing the correct incidents within the respective time period.	Prod	The mapping that was taking place in the frontend to render those curriculums was wrong. We fixed it by adding the required fields in the GraphQL query and correctly mapping it to the React components in the frontend.
BUG-105	On pressing the back button on the mobile application, the search from last time stays intact, and the UI does not reset itself.	Staging	Added a check to reset the search field when the component is unmounted in order to reset the UI whenever the user leaves it, or in other words, the component gets unmounted from the display.

Table 4.2: A table depicting some example bugs that we faced and how we resolved them.

4.2.3 Test Results and Analysis

- **Sentry and Jira Integration Efficiency**

Sentry [6]'s real-time error-tracking feature shortened the time it took to find bugs. The bugs would automatically raise an alarm via Slack [9], where the incident would be automatically logged for the respective teams to pick it up and start addressing it.

Also, Jira [7] ensured that the tasks were assigned and tracked smoothly, which enhanced teamwork and production. Also, it increased the synergy between the quality assessment engineers and the software developers, reducing the communication gap that has the possibility of arising between both the groups of

engineers.

- **Bug Fixing**

Throughout the testing phase, a lot of unnoticed bugs, around more than 150, were found by the quality assessment engineers and then were fixed by the developers in all situations and across all environments, whether it be staging, pre-prod or production itself.

Notification and e-mail delays and irregular state transitions, along with incorrect placement of UI components, were among the major problems that were successfully fixed by the team.

- **System Reliability**

The system complied with performance and was extremely responsive. It also met usability standards, thanks to the extensive testing that was conducted by the QA engineers on the team and the prompt bug-fixing by the developers.

With the bugs fixed, the quality assured, and the user experience enhanced, the final finished product was released into production, guaranteeing end consumers a stable and reliable experience without any complaints.

CHAPTER 5: RESULTS AND EVALUATION

The results obtained from the Student Behaviour Management System are briefly discussed in this chapter which were achieved by the team. It also measures its functionality and finds key findings from user input and testing also.

5.1 Results

When the project produced a working Student Behaviour Management System, the purposes of the project (collaborative management, real-time event monitoring, and enhanced user experience in behaviour management platform) were achieved. The key results that was achieved by the team are presented below:

5.1.1 System Performance

- **Response Time:** The average response time for logging and querying incidents with the system was less than 400 ms, thus producing a very smooth user experience..
- **Concurrent Usage:** The system was also able to serve numerous users (teachers, students, and families) who are using the system at any given time, without a significant footprint (performance loss).
- **Data Consistency:** Data quality checks had to be automated since upon verification we saw no disparities, even with such cases as overlapping incidents or incomplete entries which happen so often in our type of system.

5.1.2 Functionality Validation

- **Real-Time Incident Reporting:**
 - The system was validated across multiple scenarios, which included overlapping incidents and simultaneous updates.
 - We achieved 100% accuracy in real-time optimistic updates across all user interfaces.
- **Accessibility Compliance:**
 - Keyboard navigation and shortcuts were tested and met WCAG 2.1 [4] standards, making the system accessible for users with disabilities.

5.1.3 User Feedback

- **Ease of Use:** Teachers appreciated the intuitive design and the provided collaborative features, particularly the tagging and referral system, which was very crucial, as described in their words.
- **Confidential Notes:** The addition of confidential notes for incidents also received positive feedback, which added value for sensitive situations.
- **Data Visualisation:** Features such as categorised incident displays and search filters were highly regarded for enhancing usability.

5.1.4 Key Observations

There were some prominent observations from the project, which denote the impact and the points to be improved:

5.1.4.1 Positive Observations

- **Collaborative Tagging:** Teachers said there was a major improvement in collaborative management of incidents, which had resulted in minimisation of miscommunication..
- **Workflow Efficiency:** The state-based workflow (Draft, Published, Resolved) simplified the process of documenting and resolving the incidents, and it wound up with manual effort savings in the long run..
- **Improved User Adoption:** The use of accessibility features and responsive design also improved user satisfaction and absorption across all devices.

5.1.4.2 Limitations

In spite of the achieved successes, some challenges were received by the system:

- **Setup Complexity:** The process of setup just required lots of processes, and this may be problematic for customers. Automating redundant processes is one area on which to concentrate.
- **Scalability for Large Institutions:** The system was efficient enough for moderate loads, but heavier institutions with high concurrent usage may need some extra optimisation.

- Customizable Features: Some of the users requested additional customisation features like the custom incident categories and the custom notification preference.

5.2 Comparison with Existing Solutions

The system was compared with traditional behaviour management methods and similar digital solutions to evaluate its relative advantages:

Feature	Traditional Methods	Existing Digital Solutions	Proposed System
Real-Time Reporting	Not available	Limited	Fully implemented with real-time updates
Collaboration	Manual sharing	Partial (some systems allow tagging)	Comprehensive tagging and referrals
Accessibility	Non-compliant	Partially WCAG-compliant	Fully WCAG 2.1-compliant
Incident Workflow	Manual and inconsistent	Fixed workflows	State-based workflow with flexibility
Performance	Slow due to manual processes	Moderate scalability	High responsiveness and scalability

Table 5.1 Comparison of implemented features with pre-existing solutions

5.2.1 Evaluation Metrics

The system was evaluated using the following key performance indicators:

5.2.1.1 Accuracy Metrics

- **Accuracy of Incident Logging:** Made sure that incidents were logged correctly, even when several users were interacting simultaneously.
- **Tagging Precision:** Verified that, at a 99% accuracy, all the tags and referrals were correctly issued.

5.2.1.2 Usability Metrics

- **Ease of Use:** Based on the user feedback survey, the system's ease of use was scored an average of 4.7/5.
- **Compliance with Accessibility:** All WCAG [4] 2.1 requirements were fulfilled, ensuring accessibility.

5.2.1.3 Performance Metrics

- **Response Time:** Across all modules, an average response time of less than 400 ms was maintained.
- **Load Handling:** In stress testing, 100 concurrent users were managed successfully.

5.3 User Impact

The deployment of the behaviour management system led to evident enhancements for all main users — teachers, administrators, and students. All benefited from quicker processes, more effective insights, and enhanced collaboration.

- **Teachers**

Experienced 40% less time spent documenting behavioural incidents in the long run. The time saved enabled them to spend more time engaging with the students and developing interventions for persistent behavioural patterns. There was better collaboration, where teachers could now assign other staff, discuss cases within the platform directly, and monitor follow-ups without involving other means of communication.

- **Administrators**

Administrators had access to organised behaviour data, which enabled them to spot trends, common problems, and areas requiring policy changes or support mechanisms. Also, decision-making improved, with the capacity to monitor incidents

by class, type, or frequency. It was simpler to produce reports and summaries for meetings, parent discussions, or even academic review.

- **Students**

Students benefited from increased transparency in how their behaviour was documented, monitored, and assessed. They received more equitable treatment, as events were recorded with richer context and fewer subjective biases. Positive behaviours were more often identified and rewarded, increasing motivation and encouraging ongoing good behaviour.

CHAPTER 6: CONCLUSIONS AND FUTURE SCOPE

The Student Behaviour Management System has established a robust foundation in terms of handling the issues of behaviour tracking and management in schools. Through a concentration on real-time reporting, collaboration, and usability, the project has developed significantly in education environments. Yet, the suggested future development will maintain the system scalable, adaptable, and effective in the long term. By leveraging advanced analytics, automation, and integration, our application can further develop as a mature solution for behaviour management, establishing new standards in education technology.

6.1 Conclusions

The Student Behaviour Management System take primary into consideration the main limitations of the traditionally used behaviour-tracking systems in schools. The system contributes structure and flexibility to the management of behavioural data through the inclusion of such features as real-time incident reporting, collaborative tagging and a state based structured workflow. The project's main contributions are:

- **Better Coordination**

Teachers are now able to simply collaborate on behaviour incidents by tagging one another and adding context directly in the system. This promotes a more consistent response to managing student behaviour, and interventions become more effective and timely.

- **Simplified Workflows**

The draft, published and resolved model assists teachers in keeping themselves organised in reporting incidents. It reduces confusion, prevents repetition of work, and ensures that nothing slips through the cracks.

- **Improved Usability**

The site is designed with ease of use and accessibility in mind. It runs effortlessly across devices — desktops, tablets, and phones — and is keyboard-navigable, screen reader-friendly, and responsive, which means it can be used by all.

- **Scalability and Performance**

The system operates seamlessly even in heavy traffic because of a well thought out backend and easy management of data. There will be uniform performance regardless of having 10 users or 10,000 users as a result of the architecture ensuring so with no significant lag or crash.

This is a jump start in the use of technology to enhance better behaviour management in schools in bringing a positive learning environment and good relation between teachers and students.

6.2 Future Scope

Even though the purposes of the system have been achieved, there are some areas that can be improved and expanded in order to improve its utility and adaptability. Below, a detailed discussion of the future scope:

6.2.1 Roles and Permissions (RnP) Feature

- **Current Gap**

The present system lacks a fine-grained control over the duties of the users, which limits its performance in different user categories including administrators, instructors, and counsellors. Such absence of differentiated access and functional boundaries lead to threats of improper handling of data and reduces operational transparency.

- **Proposed Enhancement**

Implement a strict Role-Based Access Control (RBAC) system that manages viewing, creation, editing as well as closing of incidents as per set roles. Include higher level of roles such as Super Admin and Institution admin with greater level of

privileges such as Principal and Master Master with the possibility to define and control the action hierarchies. Also, offer customizable roles to enable institution specific organisational rules and structures

- **Expected Benefits**

This function, Roles and Permissions (RnP) adds data security and privacy through restricted access to confidential student information to those with permission thus ensuring institutional practice of policies and regulations. It enhances the efficiency of operations by facilitating an accountable environment with simplified workflow processes and good delineation of roles and permissions based on user identities. Moreover, it has a scalable and modular architecture that can support various forms of educational organisations, from small school systems to large school systems, adapting to various access control needs without compromising on performance or flexibility.

6.2.2 Backend Enhancements

- **Current Gap**

Although the current backend architecture works well on present functionalities, it lacks extensibility/ resilience to support future feature expansion as well as user increase. The absence of more advanced data-handling optimisations prevents the system from running well without interruption under high loads and therefore scalability plays a significant role in long-term operation.

- **Future Implementation**

To solve these shortcomings, the backend will be optimised by optimising GraphQL [2] queries such that massive data sets can be processed within very low latencies. Furthermore, high levels of caching mechanisms such as Redis will reduce data-fetch times by an enormous margin by caching data more frequently accessed in memory. Also, the integration of bulk operations for data such as import and export of incident records will make reporting and administrative control between institutions smooth.

- **Benefits**

Such back-end upgrades will make the system very responsive and scalable as concurrent users increase, with a smooth interface to serve entry spikes. Less stressed server loads at peak usage times mean overall better performance, bulk processing capabilities and caching to increase wait times and consequently increased user interaction and better data management.

6.2.3 Automation of Repetitive Processes

- **Current Gap**

The existing system compels the users to repetitively repeat menial tasks such as tagging frequently involved students or trainers, which, in turn, incurs inefficiencies and wastage of time, and increased errors from human failures. The redundancy is particularly gruelling in environments where similar forms of incident occur repeatedly.

- **Future Implementation:**

To correct these inefficiencies, a number of critical features will help to automate. Storing templates in advance will allow users to record ordinary incidents quickly and without repeating the same information repeatedly. Further, that likely AI output from past experience will contribute to automatic tagging or categorization, reducing the cognitive load. The system will also set automatically reminders related to open cases or pending actions so as not to miss key follow-ups.

- **Benefits**

These changes will automatically reduce the levels of administration for teachers thus freeing more time for teacher-students' interaction as well as behavioural interventions. Alongside increasing efficiency, automation also makes the process of data entry more consistent and evolving less frequent instances of manual errors with increased uniformity in incident logs.

6.2.4 Advanced Data Analytics and Reporting

- **Current Gap**

Even though the current system is successful at overseeing basic incident logging, it does not have any feature to support complex analytics. There is no function to track or plot tendencies of behaviours during a period and therefore teachers find it difficult to analyse trends or predict possible problems on time.

- **Future Implementation**

The system will be supplemented by interactive dashboard where graphs laid out in a data intuitive form will display trend in behaviour and longitudinal data. Predictive analytics will be used to identify students who have traditionally, or potentially may, have been at risk given the frequency of incident. Also, the report templates that can be customised and exported will also be given to the teachers, administrators and parents for supporting decision making based on data.

- **Benefits**

These enhancements will enable able to have teachers profound understanding of pupil behaviour, making more proactive and effective interventions. Visual representation of data makes information clearer, with predictive software allowing for early spotting of behavioural concerns, ultimately helping create a healthier, more responsive learning environment.

6.2.5 Integration with Third-Party Platforms

- **Current Gap**

Currently, the system is a self-contained program that does not have the ability to converse or interface with other critical school management systems. This segregation restricts the breadth of the behavioural insights and inhibits the creation of an integrated student profile.

- **Future Implementation**

Plans are also to integrate to popular LMS such as Moodle and Google Classroom in sharing of information across platforms smoothly. The system will also integrate with attendance and academic records to create a view of the all-round performance of each student. Besides, External API will be extended to support custom integration with 3rd party tools adopted by different institutions.

- **Benefits**

These improvements will quadruple the flexibility of the system by bringing together behaviour, academic, and attendance data on one interface. Interoperability not only increases high quality intelligence available to educators but also supports easy adoption and compatibility with the existing digital spaces in schools.

6.2.6 Scalability for Large Institutions

- **Current Gap**

While working well with moderate loads the system lacks optimisations for large-scale deployment of the larger institutions. This weakness can lead to bottle necks and poor performance in high user loads.

- **Future Implementation**

To tackle scalability issues, the database design will be shifted to a distributed scheme that can handle big data efficiently. A cloud infrastructure like AWS [8] or Azure will be adopted for high availability, redundancy, and fault tolerance. Load balancing methods will also be implemented to distribute incoming traffic evenly, thus ensuring ease of use when there is high usage.

- **Benefits**

These enhancements will enable the system to expand seamlessly, supporting thousands of users without loss of speed or dependability. The infrastructure will be resilient enough to support the needs of institutions in which real-time, unbroken behaviour monitoring is critical.

6.2.7 Mobile Application Development

- **Current Gap**

The platform is made responsive to the system, and users are given the advantage of experiencing the platform on different devices. Nevertheless, in spite of this responsive feature, the absence of a specific mobile app deprives the platform of enhanced accessibility and user experience. Although mobile-responsive websites offer primary functionality, there is no way they can match the performance and ease provided by native mobile apps.

- **Future Implementation**

Future implementation is to create native mobile applications on both iOS and Android platforms. The applications will have offline capabilities, enabling the user to log incidents and access critical information without an active internet connection. Furthermore, push notifications will be added to inform the user of new incidents, changes, or action requirements in real-time. The applications will be optimised for low-bandwidth areas as well to ensure seamless performance even in areas with limited or poor internet access.

- **Benefits**

Through this native mobile application development; teachers will get the ease to handle incidents on the move; why they will be able to log and view incidents anytime and anywhere. In areas where there is no appellate connectivity, the offline features will be quite handy and push notifications will ensure that especially urgent actions or incidents are communicated in real time to users. All of these improvements will promote accessibility, efficiency, and usability, especially; for mountain institutions with network problems.

6.2.8 AI-Based Behavioural Insights

- **Current Gap**

Incidence of incidents at the system at present rely on manual tagging and categorizing which is ineffective and prone to errors. Hand work limits the scope and performance of the system in operation for large volumes of information.

- **Future Implementation**

In an effort to fill this gap, machine learning models will be utilized to process incident data and to automatically suggest appropriate interventions. Automated categorisation of description of the incidents and patterns embedded in the data will be done through the natural language processing (NLP). AI will also be used for recommending individualised behaviour improvement plans for students according to their unique needs and behaviours.

- **Benefits**

Machine learning and NLP integration will show hidden behaviour trends and patterns, which will enable teachers identify what needs to be done earlier and solve problems before they become complicated. This intelligent system will increase efficiency, provide more useful insights, and increase the general capacity to serve students in a more personalised manner.

References

- [1] ReactJS Documentation, *React – A JavaScript library for building user interfaces*, 2024. [Online]. Available: <https://react.dev/>. [Accessed: Nov. 30, 2024].
- [2] Apollo GraphQL, *Apollo Client Documentation*, 2024. [Online]. Available: <https://www.apollographql.com/docs/react/>. [Accessed: Nov. 30, 2024].
- [3] PostgreSQL Documentation, *PostgreSQL: The World's Most Advanced Open Source Relational Database*, 2024. [Online]. Available: <https://www.postgresql.org/>. [Accessed: Nov. 30, 2024].
- [4] Web Content Accessibility Guidelines (WCAG), *Web Content Accessibility Guidelines 2.1*, W3C, 2024. [Online]. Available: <https://www.w3.org/WAI/standards-guidelines/wcag/>. [Accessed: Nov. 30, 2024].
- [5] Toddle Internal Documentation, *Internal guidelines and system documentation*, Proprietary Documentation, 2024.
- [6] Sentry Documentation, *Sentry: Application Monitoring and Error Tracking Software*, 2024. [Online]. Available: <https://sentry.io/>. [Accessed: Nov. 30, 2024].
- [7] Jira Documentation, *Jira Software by Atlassian*, 2024. [Online]. Available: <https://www.atlassian.com/software/jira>. [Accessed: Nov. 30, 2024].
- [8] AWS Documentation, *AWS Cloud Computing White Papers*, Amazon Web Services, 2024. [Online]. Available: <https://aws.amazon.com/whitepapers/>. [Accessed: Nov. 30, 2024].
- [9] Slack API Documentation, *Slack API: Build your integration*, 2024. [Online]. Available: <https://api.slack.com/>. [Accessed: Nov. 30, 2024].
- [10] Lodash Documentation, *A modern JavaScript utility library delivering modularity, performance & extras*, 2024. [Online]. Available: <https://lodash.com/docs/>. [Accessed: Nov. 30, 2024].

- [11] Ant Design, *Ant Design - A design system for enterprise-level products*, 2024. [Online]. Available: <https://ant.design/docs/react/introduce>. [Accessed: Nov. 30, 2024].
- [12] Redux Toolkit Documentation, *Redux Toolkit: The official, recommended approach for writing Redux logic*, 2024. [Online]. Available: <https://redux-toolkit.js.org/>. [Accessed: Nov. 30, 2024].
- [13] React Router Documentation, *React Router: Declarative routing for React apps*, 2024. [Online]. Available: <https://reactrouter.com/>. [Accessed: Nov. 30, 2024].
- [14] Tailwind CSS Documentation, *Tailwind CSS: A utility-first CSS framework for rapidly building custom user interfaces*, 2024. [Online]. Available: <https://tailwindcss.com/docs>. [Accessed: Nov. 30, 2024].

Appendix

A.1 Project Plan

A.1.1 Phase 1: Planning and Design

1. Define Project Scope and Objectives
 - a. Establish the core functionality of the Student Behaviour Management System.
 - b. Identify user requirements through meetings with stakeholders (teachers, administrators, and students).
 - c. Define the states of incidents (Draft, Published, Resolved) and tagging functionality.
2. Design System Architecture
 - a. Develop a three-tier architecture: frontend, backend, and database.
 - b. Select key technologies: ReactJS [1] for the frontend, Apollo Server [2] and GraphQL for the backend, and PostgreSQL [3] for database management.
 - c. Incorporate accessibility and scalability requirements.
3. Develop User Stories
 - a. The teacher logs a new incident and tags another teacher for collaboration.
 - b. The administrator views all incidents for a specific period and downloads a report.
 - c. A user accesses the system via keyboard-only navigation and retrieves incident details.

A.1.2 Phase 2: Development

1. Frontend Development
 - a. Build the user interface using ReactJS [1] and Material-UI for responsive design.
 - b. Implement accessibility features, including WCAG-compliant components and keyboard navigation.
 - c. Integrate search and filter functionality for incidents.
2. Backend Development

- a. Set up Apollo Server with GraphQL for efficient data querying and mutation.
 - b. Design and develop state-based workflows for incidents.
 - c. Integrate real-time updates using GraphQL subscriptions.
- 3. Database Management
 - a. Create PostgreSQL schema for users, incidents, and tags.
 - b. Implement indexing for faster queries and data integrity checks using constraints.
 - c. Populate initial data for testing and demonstration purposes.
- 4. Integration
 - a. Connect the front end and back end to create a cohesive system.
 - b. Implement API calls for real-time incident logging and retrieval.

A.3 Phase 3: Testing

- 1. Unit Testing
 - a. Test individual components like the incident logging form and state transitions using Jest.
 - b. Validate backend logic and database queries with Mocha.
- 2. Integration Testing
 - a. Verify seamless interaction between frontend, backend, and database using Postman and Apollo Devtools.
 - b. Test the workflow for creating, updating, and resolving incidents.
- 3. System Testing
 - a. Test the application on staging, pre-prod, and production environments with QA engineers.
 - b. Identify bugs using Sentry and track fixes via Jira.
 - c. Ensure WCAG accessibility compliance through manual and automated testing.
- 4. Performance Testing
 - a. Conduct load testing to evaluate response times under concurrent usage.
 - b. Validate system performance with real-time user interactions.

JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY, WAKNAGHAT
DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING AND INFORMATION TECHNOLOGY

PLAGIARISM VERIFICATION REPORT

Date: May, 2025.

Type of Document: B.Tech. (CSE / IT) Major Project Report

Name: _____ **Enrollment No.:** _____

Contact No: _____ **E-mail:** _____

Name of the Supervisor (s): _____

Title of the Project Report (in capital letters): _____

UNDERTAKING

I undertake that I am aware of the plagiarism related norms/regulations, if I found guilty of any plagiarism and copyright violations in the above major project report even after award of degree, the University reserves the rights to withdraw/revoke my major project report. Kindly allow me to avail plagiarism verification report for the document mentioned above.

- Total No. of Pages:
- Total No. of Preliminary Pages:
- Total No. of Pages including Bibliography/References:

Signature of Student

FOR DEPARTMENT USE

We have checked the major project report as per norms and found **Similarity Index**%. Therefore, we are forwarding the complete major project report for final plagiarism check. The plagiarism verification report may be handed over to the candidate.

Signature of Supervisor

Signature of HOD

FOR LRC USE

The above document was scanned for plagiarism check. The outcome of the same is reported below:

Copy Received On	Excluded	Similarity Index (%)	Abstract & Chapters Details	
	<ul style="list-style-type: none">• All Preliminary Pages• Bibliography/ Images/Quotes• 14 Words String		Word Count	
Report Generated On			Character Count	
		Submission ID	Page Count	
			File Size (in MB)	

Checked by

Name & Signature

Librarian

ORIGINALITY REPORT

1 %	1 %	0 %	0 %
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	www.ir.juit.ac.in:8080 Internet Source	1 %
2	Submitted to University of Maryland, University College Student Paper	<1 %
3	Submitted to African Leadership University Student Paper	<1 %
4	rmag.eu Internet Source	<1 %

Exclude quotes On
Exclude bibliography On

Exclude matches Off

*% detected as AI

AI detection includes the possibility of false positives. Although some text in this submission is likely AI generated, scores below the 20% threshold are not surfaced because they have a higher likelihood of false positives.

Caution: Review required.

It is essential to understand the limitations of AI detection before making decisions about a student's work. We encourage you to learn more about Turnitin's AI detection capabilities before using the tool.

Disclaimer

Our AI writing assessment is designed to help educators identify text that might be prepared by a generative AI tool. Our AI writing assessment may not always be accurate (it may misidentify writing that is likely AI generated as AI generated and AI paraphrased or likely AI generated and AI paraphrased writing as only AI generated) so it should not be used as the sole basis for adverse actions against a student. It takes further scrutiny and human judgment in conjunction with an organization's application of its specific academic policies to determine whether any academic misconduct has occurred.

Frequently Asked Questions

How should I interpret Turnitin's AI writing percentage and false positives?

The percentage shown in the AI writing report is the amount of qualifying text within the submission that Turnitin's AI writing detection model determines was either likely AI-generated text from a large-language model or likely AI-generated text that was likely revised using an AI-paraphrase tool or word spinner.

False positives (incorrectly flagging human-written text as AI-generated) are a possibility in AI models.

AI detection scores under 20%, which we do not surface in new reports, have a higher likelihood of false positives. To reduce the likelihood of misinterpretation, no score or highlights are attributed and are indicated with an asterisk in the report (*%).

The AI writing percentage should not be the sole basis to determine whether misconduct has occurred. The reviewer/instructor should use the percentage as a means to start a formative conversation with their student and/or use it to examine the submitted assignment in accordance with their school's policies.

What does 'qualifying text' mean?

Our model only processes qualifying text in the form of long-form writing. Long-form writing means individual sentences contained in paragraphs that make up a longer piece of written work, such as an essay, a dissertation, or an article, etc. Qualifying text that has been determined to be likely AI-generated will be highlighted in cyan in the submission, and likely AI-generated and then likely AI-paraphrased will be highlighted purple.

Non-qualifying text, such as bullet points, annotated bibliographies, etc., will not be processed and can create disparity between the submission highlights and the percentage shown.

