PROJECT REPORT ON

SEC NEXUS

(An Integrated Digital Platform for Event Operations within St. Edmund's College)

SUBMITTED BY

Sambuddha Das
University Roll No.: 23111408
Registration No.:23006411

IN PARTIAL FULFILLMENT OF THE FYUP UNDERGRADUATE PROGRAMME UNDER NORTH EASTERN HILL UNIVERSITY (NEHU)

Department of Computer Science
St. Edmund's College
Shillong

Department of Computer Science
St. Edmund's College
Shillong - 793003

CERTIFICATE OF AUTHENTICITY

This is to certify that the project report entitled **"SEC-NEXUS: A Web-Based Event Management System for St. Edmund's College, Shillong"** is a bona fide work submitted by **Sambuddha Das**, bearing **Roll No.: 23111408** and **Registration No.: 23006411** in partial fulfilment of the requirements for the 5th Semester Major Project during the academic year 2025.

To the best of my knowledge, the work presented herein is original and has not been submitted elsewhere for the award of any other degree or diploma.

Mr Jeremy Kharchandy	Mrs. Preeti Thapa	External Examiner
Project Supervisor	Head of Department	
	Department of Computer Science	

ACKNOWLEDGEMENT

I would like to express my sincere gratitude to my project supervisor, Sir Mr Jeremy Kharchandy, for their invaluable guidance, support, and encouragement throughout this project. Their insights and timely feedback were instrumental in shaping the project from its initial concept to its final implementation.

I also extend my heartfelt thanks to the faculty of the Computer Science Department at St. Edmund's College for providing me with the necessary academic foundation and resources.

Finally, I am grateful to my friends for their constant support and understanding, which motivated me to complete this project successfully.

1. INTRODUCTION

1.1 Background

Educational institutions like St. Edmund's College are vibrant hubs of academic, cultural, and co-curricular activities. These events, ranging from departmental workshops and seminars to college-wide fests and club activities, are fundamental to the holistic development of students.

1.2 Motivation

The traditional methods of managing these events, such as physical notice boards, fragmented social media posts, and word-of-mouth communication, are often inefficient. This fragmentation leads to several challenges:

- **Information Silos**: Students and faculty find it difficult to get a consolidated view of all upcoming events.
- **Poor Discoverability**: Events relevant to specific departments or clubs are easily missed by interested students.
- **Administrative Overhead**: The process for event organisers to submit proposals, get approvals, and publicise their events is often manual, paper-based, and slow.
- **Scheduling Conflicts**: A lack of a central calendar increases the likelihood of overlapping events.

This project was motivated by the need to solve these issues by creating a centralized, digital-first platform.

1.3 Importance of the Research Area

The development of web-based management systems is a critical area of software engineering. This project applies modern web technologies, specifically a full-stack framework (Next.js 15 with React 19), a NoSQL database (MongoDB), and third-party services for authentication and file storage, to create a practical, scalable, and secure solution for a real-world organisational problem.

2. PROBLEM STATEMENT

The core problem addressed by this project is the lack of a centralized, streamlined, and accessible system for managing the entire lifecycle of events at St. Edmund's College.

The existing fragmented process results in:

1. **Inefficient Promotion**: Organisers struggle to reach their target audience effectively.

2. **Low Student Engagement**: Students are unaware of opportunities that match their interests.

3. **Lack of Administrative Oversight**: The college administration has no simple mechanism to moderate, approve, or reject event postings, leading to potential quality control and scheduling issues

4. **Data Fragmentation**: Event information is scattered across multiple platforms, making it impossible to analyse or archive.

5. **No Automated Approval Window**: Event participation requests are manually reviewed without centralised tracking.

SEC-NEXUS is designed to be the single source of truth for all events, streamlining the process from creation and approval to discovery and participation.

3. OBJECTIVES

The primary objective of this project is to design, develop, and deploy a comprehensive event management platform for St. Edmund's College.

The specific objectives are:

1. To Develop a Secure Authentication System: Implement a robust user sign-up and sign-in flow to manage user identities and protect routes.
2. To Implement Full Event CRUD Functionality: Enable authorized users to create, read, update, and delete events.
3. To Integrate File Uploads: Provisionalize banner uploads for event creation.
4. To Build an Admin Moderation Dashboard: Create a secure admin-only page where a designated administrator can review all submitted events and either "approve" or "reject" them.
5. To Implement Dynamic Filtering: Allow users to filter and search for events on the homepage based on title, category, department, and/or club.
6. To Ensure Complex Form Validation: Enforce complex business logic (e.g., an event can be for a department *or* a club, but not both; roles are required)
7. To Create a Personalized User Profile: Develop a profile page that lists all events created by the currently logged-in user.
8. To Deliver a Responsive UI: Build a modern, mobile-first user interface.

4. METHODOLOGY

4.1 Development Model

An **Agile-inspired iterative model** was used. Features were developed and refined through multiple feedback loops.

4.2 System Architecture

- Modular Next.js frontend and API routes.
- Secure storage for uploaded media.
- Role-based authentication for admin controls.

4.3 Technologies Used

Category	Technology
Frontend	Next.js 15 + Tailwind CSS
Authentication	Clerk
Database	MongoDB Atlas
File Storage	Uploadthing
Hosting	Vercel

5. DATASET & DATA ORGANIZATION

The system uses a dynamic, user-generated dataset stored in MongoDB.

MongoDB Collections:

- **users** – Stores synced user identity data.
- **events** – Stores event details, timestamps, banners, and metadata.
- **categories** – List of event types.
- **clubs** – Club directory.
- **departments** – Department list.

Data is generated on user sign-up and event creation.

6. EVALUATION STRATEGIES

6.1 Functional Testing

User Flow

- Verified correct signup/login behaviour.
- Event creation saves new entries as *pending*.

Validation Logic

- Department + Club both selected → Error.
- Department selected, but CR field empty → Error.
- Club selected but Club Role empty → Error.

Admin Flow

- Non-admin accessing /admin/dashboard → Redirect.
- Admin approval updates the event status to *approved*.

Public Flow

Only approved events appear on the homepage.

Filtering & Search

All filters dynamically update event listings.

6.2 Performance & Accessibility

- High Lighthouse scores on Vercel.
- Application configured as an installable **PWA** (via next-pwa).

7. RESULTS AND OUTCOMES

Major Deliverables:

- Fully deployed event platform.
- Centralised discovery hub for all events.
- Streamlined event submission process.
- Reliable admin moderation workflow.
- Secure role-based access system.
- Scalable database models.
- Dynamic UI powered by real-time data.

8. TECHNICAL SNIPPETS

This chapter includes database selection rationale, implementation details, and DFD diagrams.

8.1 Database Selection

SQL vs MongoDB Comparison

Aspect	SQL	MongoDB
Data Structure	Tables	Collections (documents)
Schema	Strict	Flexible
Relationships	Joins	References/Embeds
Scalability	Vertical	Horizontal
Diagram	ERD	Conceptual Diagram

Why MongoDB Was Chosen

SQL limitations:

- Rigid schemas slow down development.
- Frequent changes to event structure require migrations.
- Multiple joins needed between events, users, departments, and clubs.

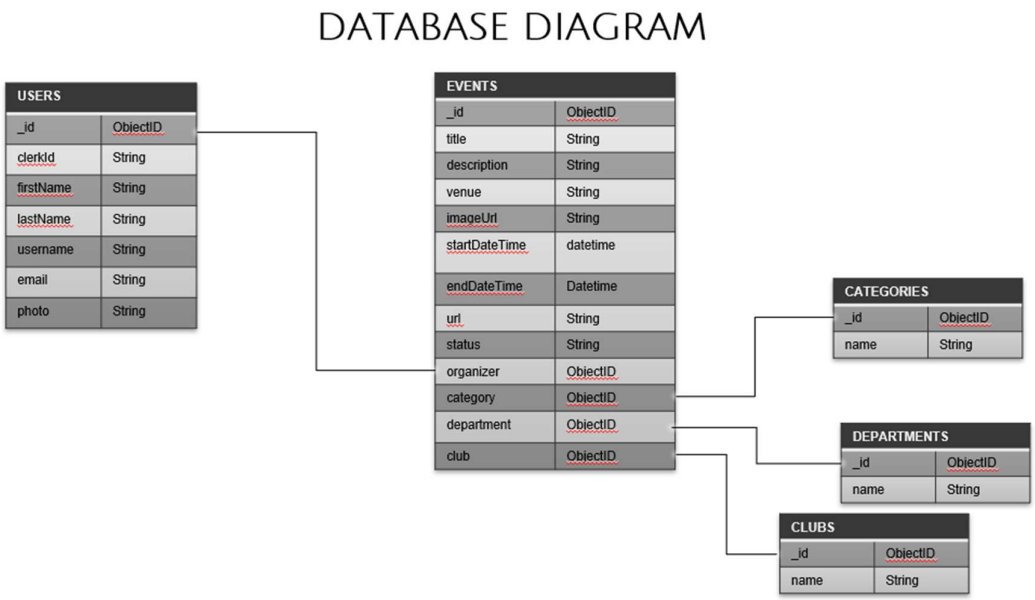
MongoDB advantages:

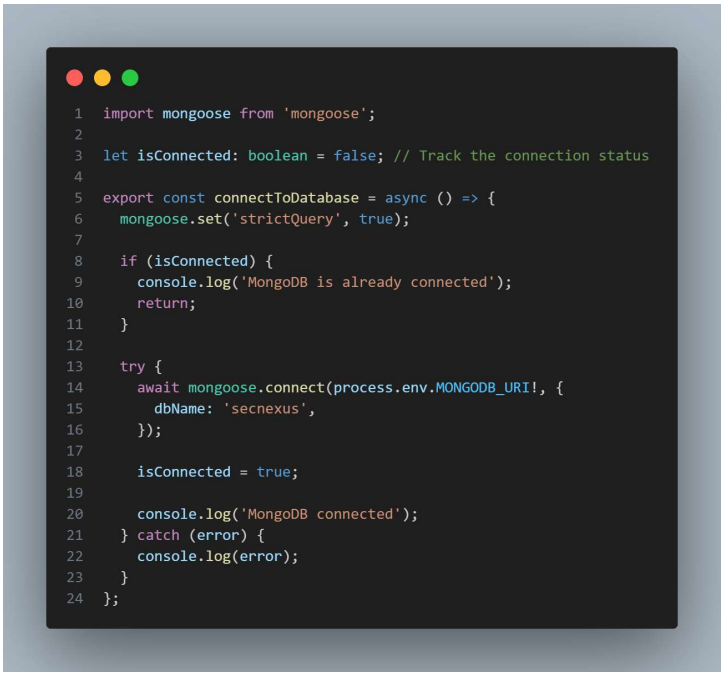
- JSON-like structure fits event models naturally.
- Schemas evolve smoothly as the platform grows.
- Faster prototyping and iteration.
- Cloud-native horizontal scaling.

8.2 SQL ERD vs MongoDB Conceptual Diagram

Element	SQL ERD	MongoDB Diagram
Unit	Tables	Collections
Attributes	Columns	Flexible fields
Relationships	Enforced with foreign keys	Soft references
Enforcement	Strict	Optional
Purpose	Exact schema blueprint	High-level structure guide

8.3 Database Connection





```
import mongoose from 'mongoose';

let isConnected: boolean = false; // Track the connection status

export const connectToDatabase = async () => {
  mongoose.set('strictQuery', true);

  if (isConnected) {
    console.log('MongoDB is already connected');
    return;
  }

  try {
    await mongoose.connect(process.env.MONGODB_URI!, {
      dbName: 'secnexus',
    });

    isConnected = true;

    console.log('MongoDB connected');
  } catch (error) {
    console.log(error);
  }
};
```

- Uses an isConnected flag to prevent multiple connections in a serverless context.
- .env stores the secure connection string.
- Shared across server actions.

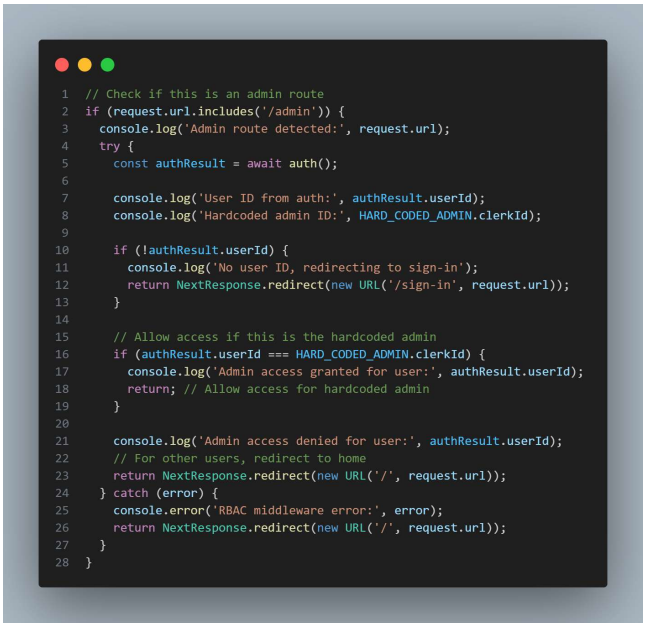
8.4 Server Actions (Next.js 15)

```
"use server"

import { connectToDatabase } from "../database";
import Category from "../database/models/category.model";
import { CreateCategoryParams } from "../types"
import { handleError } from "../utils"

export const createCategory = async ({categoryName} :CreateCategoryParams)=> {
    try {
        await connectToDatabase();

        const newCategory = await Category.create({ name: categoryName });

        return JSON.parse(JSON.stringify    (newCategory));
    } catch (error) {
        handleError(error)

    }
}

```

- Marked with **"use server"**.
- Execute only on the backend.

- Handle event creation, deletion, and updates.

8.5 Middleware (RBAC)

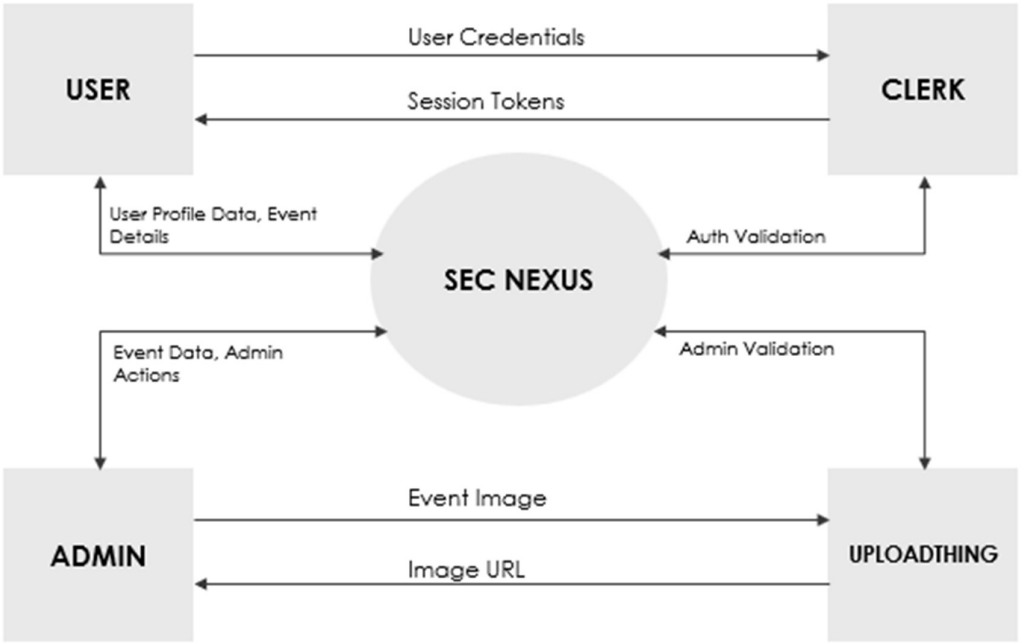


```
// Check if this is an admin route
if (request.url.includes('/admin')) {
  console.log('Admin route detected:', request.url);
  try {
    const authResult = await auth();

    console.log('User ID from auth:', authResult.userId);
    console.log('Hardcoded admin ID:', HARD_CODED_ADMIN.clerkId);

    if (!authResult.userId) {
      console.log('No user ID, redirecting to sign-in');
      return NextResponse.redirect(new URL('/sign-in', request.url));
    }

    // Allow access if this is the hardcoded admin
    if (authResult.userId === HARD_CODED_ADMIN.clerkId) {
      console.log('Admin access granted for user:', authResult.userId);
      return; // Allow access for hardcoded admin
    }

    console.log('Admin access denied for user:', authResult.userId);
    // For other users, redirect to home
    return NextResponse.redirect(new URL('/', request.url));
  } catch (error) {
    console.error('RBAC middleware error:', error);
    return NextResponse.redirect(new URL('/', request.url));
  }
}
```

- Checks every incoming request.
- Redirects non-admin users from admin routes.
- Ensures strict access control.

8.6 DATA FLOW DIAGRAMS (DFD)

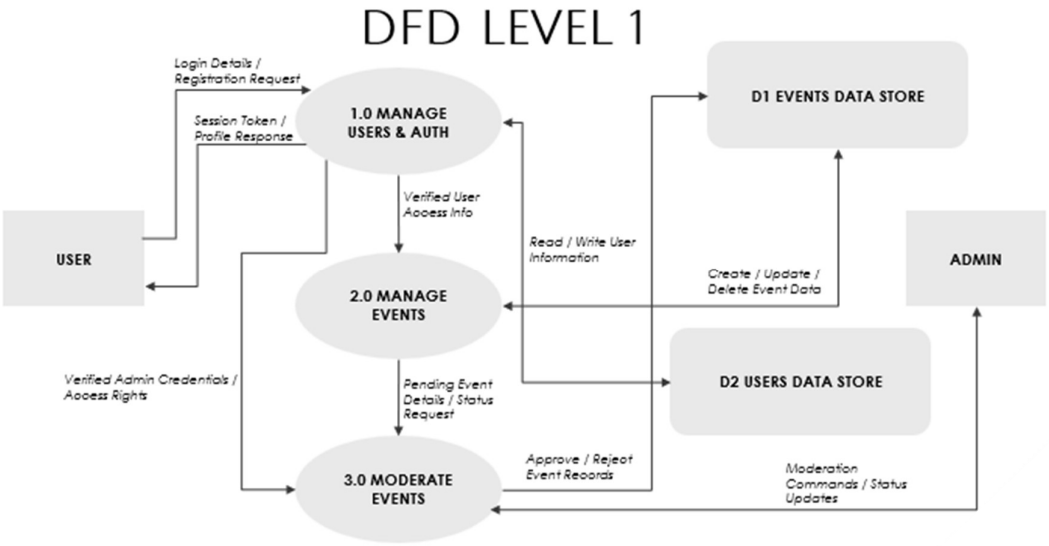
8.7 DFD Level 0 – Context Diagram



Represents SEC-NEXUS as a single high-level process interacting with:

- User
- Admin
- Database

8.8 DFD Level 1 – Expanded Diagram



Subprocesses:

1. User Authentication
2. Event Creation
3. Event Moderation
4. Event Viewing and Filtering
5. File Upload Management

10. CONCLUSION & FUTURE SCOPE

10.1 Conclusion

SEC-NEXUS successfully addresses the challenges of decentralised event workflows at St. Edmund's College. By unifying event creation, moderation, and viewing under one modern platform, it improves communication and increases student engagement.

10.2 Future Scope

1. Multi-role RBAC (faculty-level moderators).
2. Internal event registration system.
3. Auto-filled forms based on user metadata.
4. Email/SMS-based automated notification system.
5. Calendar visualisation for monthly event planning.
6. Offline access and push notifications using PWA enhancements.

11. REFERENCES

- Next.js Documentation – nextjs.org/docs
- Clerk Authentication – clerk.com/docs
- MongoDB & Mongoose – mongoosejs.com/docs
- Uploadthing – docs.uploadthing.com
- Tailwind CSS – tailwindcss.com/docs
- Shadcn UI – ui.shadcn.com
- Zod – zod.dev
- React Hook Form – react-hook-form.com