# Student Support & Grievances System

A Technical Report on Modernizing Campus Complaint Management

Introduction • Architecture • Implementation • QA

# Introduction & Problem Statement

## The Challenge

Current student grievance methods rely on paper forms, informal emails, and manual tracking. This leads to:

- ✓ Lost records and lack of accountability.
- ✓ Significant delays in resolution.
- ✓ Lack of transparency for students.
- ✓ Student frustration and disengagement.

## The Solution

A centralized, web-based platform designed to:

- ✓ Digitalize the entire reporting workflow.
- ✓ Provide real-time status tracking.
- ✓ Ensure timely responses via notifications.
- ✓ Streamline communication between students and administration.

# Project Objectives

### Submission

Secure login for students to submit detailed grievances with evidence attachments.

### Management

Admin dashboards to review, categorize, and assign complaints to specific staff.

### Tracking

Real-time lifecycle tracking (Submitted -> In Progress -> Resolved) for students.

### Reporting

Generate analytical reports to identify trends and improve campus services.

# System Requirements

## Functional Requirements

Core features the system must perform:

- **User Auth:** Role-based access (Student, Admin, Staff).
- **Grievance Mgmt:** Create, View, Update, Delete (CRUD).
- **Assignment:** Admins route issues to departments.
- **Notifications:** Email/SMS alerts on status change.

## Non-Functional Requirements

Quality attributes of the system:

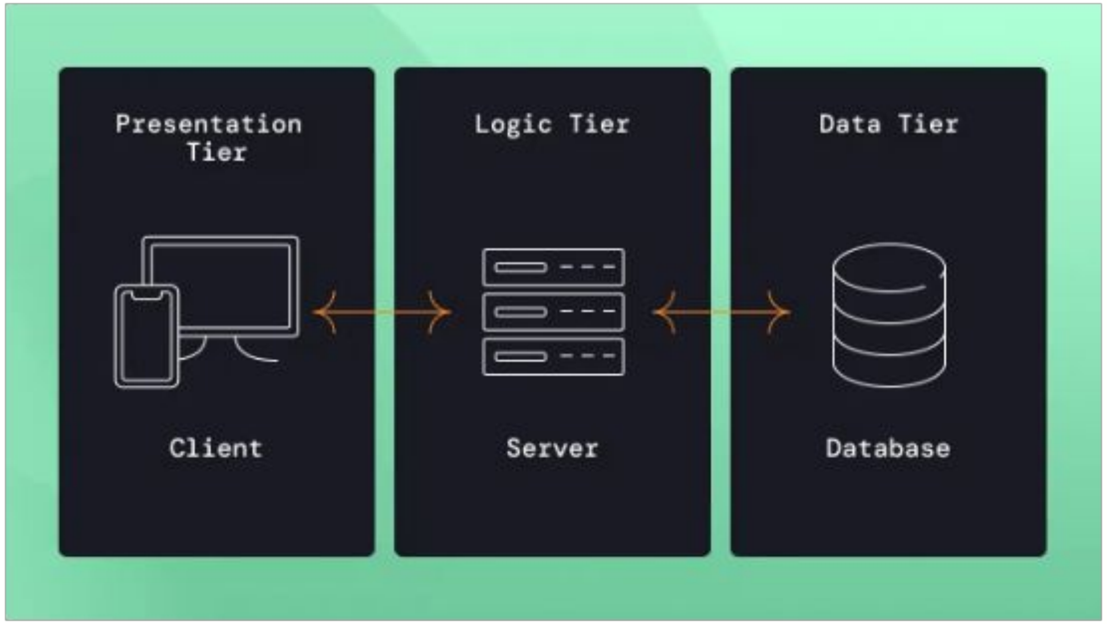
- **Performance:** Page loads < 2 seconds; handles high concurrency.
- **Security:** HTTPS, Password Hashing, Role-Based Access Control (RBAC).
- **Usability:** Intuitive Bootstrap UI/UX.
- **Scalability:** Design allows migration from SQLite to PostgreSQL.

# System Architecture

## 3-Tier Web Architecture

The system follows a classic separation of concerns pattern:

- ✓ **Presentation Layer:** HTML5, CSS3, and Bootstrap running in the client browser.
- ✓ **Application Layer:** Python Django server handling business logic, routing, and authentication.
- ✓ **Data Layer:** SQLite (Prototype) or PostgreSQL (Production) storing user and grievance records.

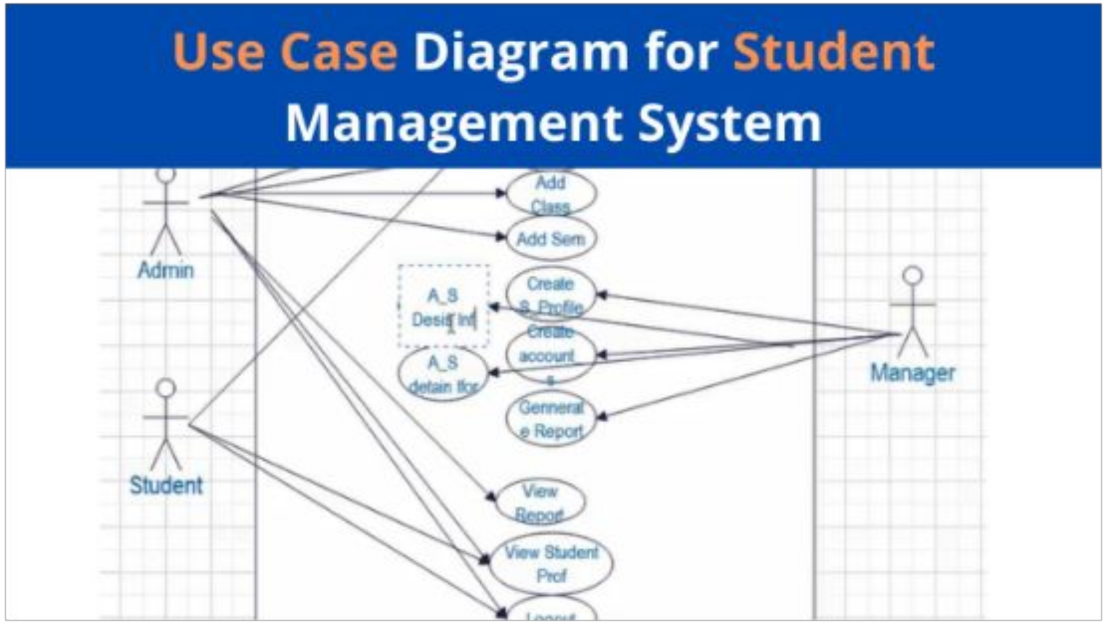


# Use Case Analysis

## Primary Actors

- **Student:** Registers, logs in, submits grievances, views history.
- **Admin:** Manages users, reviews all grievances, assigns tasks, generates reports.
- **Staff:** Views assigned grievances, updates status, adds resolution notes.

*The diagram illustrates the interactions between these actors and the system boundary.*

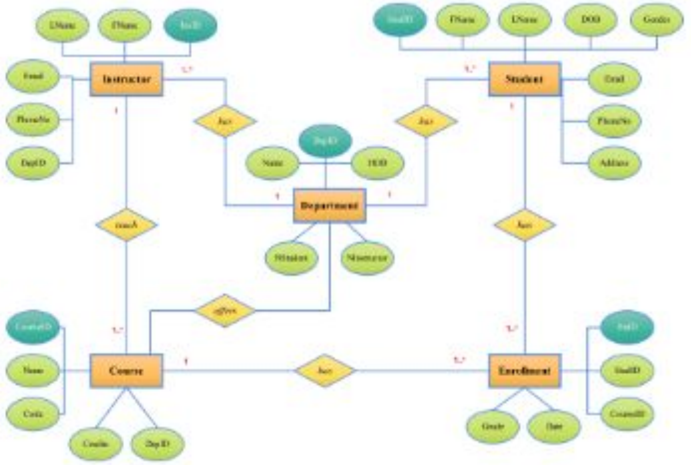


# Data Design (ER Diagram)

## Entity Relationships

Normalized database schema ensures data integrity:

- **Student Entity:** Stores profile info (ID, Name, Dept).
- **Grievance Entity:** The core record containing Description, Category, Status, and Date.
- **User (Staff/Admin):** Linked to grievances for assignment.
- **Relationships:** One-to-Many (Student → Grievances) and Many-to-One (Grievances → Staff).

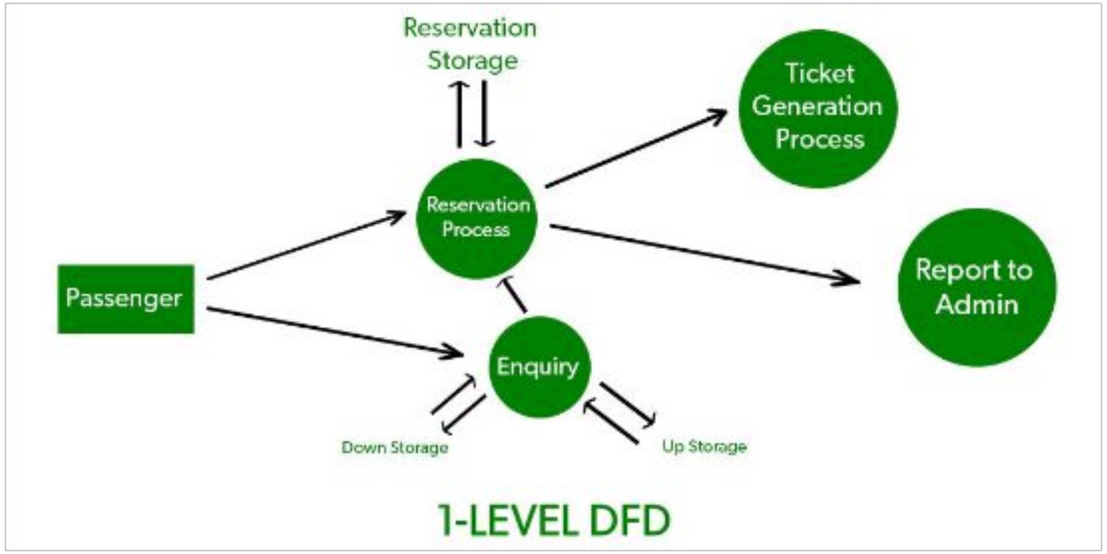


# Data Flow Diagram (Level 1)

## Information Flow

Tracking data from input to storage:

- ✓ **Input:** Student fills form → "Submit Grievance" Process.
- ✓ **Processing:** Validation logic checks input → Stores in Grievance DB.
- ✓ **Output:** System triggers confirmation and updates Admin Dashboard.
- ✓ **Feedback:** Status updates flow back from Staff → "Update Status" Process → Student View.



# Agile Implementation Plan

1

**Sprint 1**

Requirements Gathering & UI Prototyping
(Login/Register)

2

**Sprint 2**

Student Module Dev
(Submission & Dashboard)

3

**Sprint 3**

Admin/Staff Modules
(Assignment & Status Logic)

4

**Sprint 4-5**

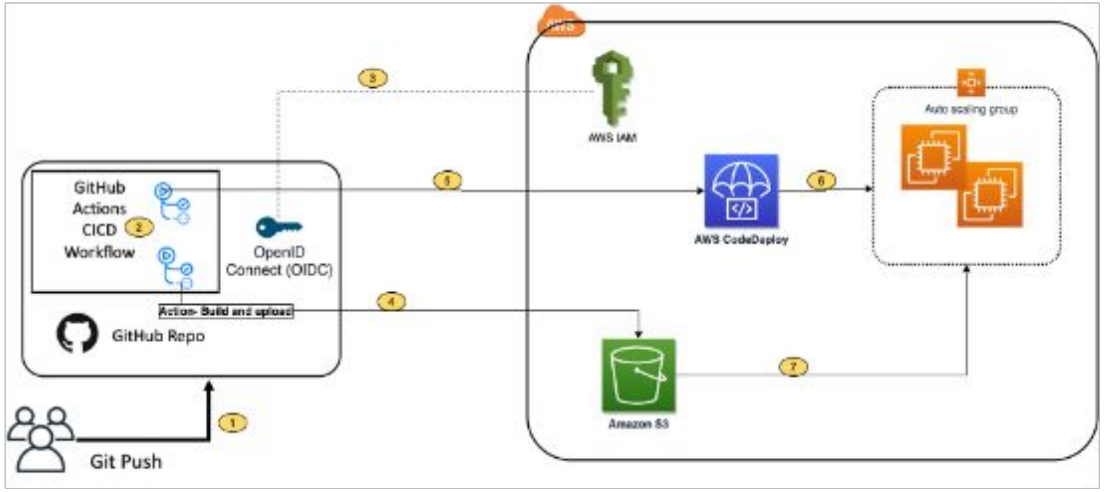
Reporting, Testing,
QA & Final Deployment

# DevOps & Deployment

## CI/CD Workflow

Automated pipeline ensuring code quality and rapid delivery:

- **Source Control:** Git/GitHub for version management.
- **CI (Build):** GitHub Actions triggers unit tests and linting (flake8) on commit.
- **CD (Deploy):** Successful builds deploy to Staging (AWS/Heroku).
- **Infrastructure:** Infrastructure as Code (IaC) ensures reproducible environments.



# Quality Assurance Strategy

## Unit Testing

Testing individual components like Models and Forms using Django TestCase to verify core logic.

## Integration Testing

Verifying interactions between modules (e.g., Submission triggers Database + Notification).

## System Testing

End-to-end validation of the entire workflow in a staging environment mirroring production.

## UAT

User Acceptance Testing with actual students and staff to ensure requirements are met.

# Future Scope & Sustainability

## Sustainability & Growth

**Green IT:** Hosting in renewable-energy powered regions and using efficient caching (Redis) to reduce CPU load.

## Future Enhancements

- ✓ **Mobile App:** Native Android/iOS app for easier access.
- ✓ **AI Integration:** Auto-categorization of complaints using NLP.
- ✓ **Serverless:** Migrating notification services to AWS Lambda.

# Conclusion

## Achievements

The system successfully meets all core objectives: transparent grievance submission, role-based management, and automated tracking. It replaces inefficient manual processes with a robust digital solution.

## Lessons Learned

**Technical:** Rapid prototyping with Django and ORM accelerated development.
**Process:** Clear requirement specification (SRS) and Agile feedback loops were critical for minimizing rework.

# Questions?

Thank you for your time.

# Image Sources

https://vfunction.com/wp-content/uploads/2024/05/blog-3-tier-application.webp

Source: vfunction.com

https://i.ytimg.com/vi/aHezL5MzoEw/maxresdefault.jpg

Source: www.youtube.com

https://images.wondershare.com/edrawmax/templates/er-diagram-for-student-management-system.png

Source: edrawmax.wondershare.com

https://media.geeksforgeeks.org/wp-content/cdn-uploads/20220517162903/level-1.jpg

Source: www.geeksforgeeks.org

https://d2908q01vomqb2.cloudfront.net/7719a1c782a1ba91c031a682a0a2f8658209adbf/2022/03/27/1-ArchitectureDiagram.png

Source: aws.amazon.com

https://techteamsynergy.com/wp-content/uploads/2025/09/Gemini_Generated_Image_hziooohziooohzio-e1756914007491.webp

Source: techteamsynergy.com