



BLACKBUCK
ENGINEERS

FULL STACK DEVELOPMENT INTERNSHIP

Django-School-Management

A PROJECT REPORT ON

Django-School-Management

A Report Submitted to

IIDT - Blackbuck Engineers Pvt. Ltd



BLACKBUCK
ENGINEERS

Submitted By

Name: GUNUPURU RAVI KUMAR

Roll Number: Y21CSE279033



Acknowledgement

I would like to express my deepest gratitude to the following individuals for their invaluable support and guidance throughout this project:

First and foremost, I wish to thank Aquib Ajani Sir for his invaluable guidance in Full Stack Development. His expertise and teaching have significantly enhanced my understanding and skills in this domain. Aquib Sir's approach to complex concepts, breaking them down into manageable parts, has been particularly beneficial. His patience and willingness to address all queries have been greatly appreciated. The hands-on experience and practical insights provided have been crucial in shaping my development skills and ensuring the success of this project.

I am equally grateful to Rajkumar Sir and Anuradha Thota Mam for their thorough explanation of the internship benefits and the process of completion. Their detailed guidance on the various aspects of the internship has been incredibly helpful. Rajkumar Sir's insights into the industry and the practical applications of theoretical knowledge have been enlightening. His encouragement and constructive feedback have been pivotal in navigating the challenges faced during the internship.

Anuradha Thota Mam's meticulous approach to the internship process has ensured that all necessary steps were clearly understood and effectively implemented. Her support in clarifying doubts and providing timely information has been indispensable. The structured framework she provided has been instrumental in smoothly completing the internship requirements.



Abstract:

The title of the project is “**Django School Management System**”. This is a comprehensive web application designed to handle all the administrative and academic operations of educational institutions. SMS has most of the facilities that a modern school requires to computerize its day-today jobs. Leveraging the robust capabilities of the Django web framework, this system provides a user-friendly interface and efficient functionality to manage various aspects of school management, including student enrolment, attendance, grades, schedules, and communication between teachers, students, and parents.

Key Features:

The main key features of the app include the:

1. User Management
2. Student Information System (SIS)
3. Attendance Management
4. Grade Management
5. Schedule and Timetable Management
6. Communication Module
7. Reports and Analytics

Technologies Used:

- Frontend: HTML, CSS, JavaScript, Bootstrap
- Backend: Django (Python)
- Database: MySQL/SQLite
- Other Tools: Django Rest Framework (DRF) for API development, Celery for asynchronous tasks, and Redis for caching.

Objectives:



- To reduce the administrative burden on school staff by automating routine tasks.
- To improve the accuracy and efficiency of data management within the institution.
- To facilitate better communication and collaboration between all the stakeholders.
- To provide insights through detailed reports and analytics for informed decision-making.

Introduction:

The Django School Management System is a comprehensive web application designed to handle the multifaceted administrative and academic operations of educational institutions. Developed using the Django framework (Python) for the backend, and technologies like HTML, CSS, JavaScript, and Bootstrap for the frontend, this system ensures a robust, scalable, and user-friendly interface. The core objective of this project is to automate routine tasks, reduce the administrative burden on school staff, and enhance the accuracy and efficiency of data management within the institution.

User Management:

One of the key features of the Django School Management System is its robust user management module. This module allows for the creation and management of various user roles, including administrators, teachers, students, and parents. Each user role has specific permissions and access levels, ensuring that sensitive information is only accessible to authorized personnel. The system facilitates easy registration, profile management, and authentication processes, ensuring secure and streamlined user interactions.

Student Information System:

The student information system (SIS) is the heart of the Django School Management System. It provides a centralized repository for all student-related data, including personal information, academic records, and disciplinary history. The SIS enables administrators and teachers to efficiently manage student enrolments, track academic progress, and generate comprehensive reports. This module also supports advanced search and filtering capabilities, making it easy to retrieve specific student information as needed.

Attendance Management:



Managing attendance is a crucial aspect of school administration, and the Django School Management System simplifies this task significantly. Teachers can mark daily attendance, track tardiness, and record absences directly within the system. The attendance module integrates seamlessly with the student information system, allowing for real-time updates and notifications. Parents can also access their child's attendance records through the system, promoting transparency and accountability.

Grade Management:

The grade management module of the Django School Management System allows teachers to record and manage student grades efficiently. This module supports various grading scales and methods, ensuring flexibility in assessment practices. Teachers can input grades for assignments, tests, and exams, and the system automatically calculates overall scores and generates report cards. The grade management module also provides analytical tools to track student performance trends over time.

Schedule and Timetable Management:

Creating and managing schedules can be a complex task for educational institutions, but the Django School Management System streamlines this process. The schedule and timetable management module allows administrators to create detailed timetables for classes, exams, and extracurricular activities. Teachers and students can access their schedules online, ensuring that everyone is aware of their commitments and responsibilities. The system also supports automated conflict detection, preventing scheduling overlaps and ensuring optimal resource utilization.

Communication Module:

Effective communication is vital for the smooth operation of any educational institution. The Django School Management System includes a robust communication module that facilitates seamless interaction between administrators, teachers, students, and parents. This module supports various communication channels, including email, SMS, and in-app messaging. Important announcements, reminders, and updates can be quickly disseminated to the relevant stakeholders, ensuring that everyone stays informed and engaged.



Reports and Analytics:

Data-driven decision-making is essential for the effective management of educational institutions. The Django School Management System includes a powerful reports and analytics module that provides valuable insights into various aspects of school operations. Administrators can generate detailed reports on student performance, attendance, financials, and more. These reports can be customized and exported in various formats, facilitating comprehensive data analysis and informed decision-making.

Advantages

1. Efficiency and Automation

The Django School Management System significantly enhances operational efficiency by automating various administrative tasks. This includes managing student information, attendance, grades, schedules, and fees. Automation reduces the time and effort required for manual data entry, minimizes errors, and allows staff to focus on more strategic activities.

2. Centralized Data Management

The system provides a centralized platform for storing and managing all school-related data. This centralization ensures that information is easily accessible to authorized users, improving data accuracy and consistency. It also simplifies data retrieval and reporting, enabling quick decisionmaking based on comprehensive insights.

3. Enhanced Communication

Effective communication is critical in educational settings, and the Django School Management System facilitates seamless interaction among administrators, teachers, students, and parents. The integrated communication module supports various channels, including email, SMS, and in-app messaging, ensuring timely and efficient dissemination of information.



4. User-Friendly Interface

The system is designed with a user-friendly interface, making it easy for users of all technical backgrounds to navigate and utilize its features. This ease of use reduces the learning curve and encourages adoption among staff, students, and parents.

5. Customization and Flexibility

Built on the Django framework, the system is highly customizable to meet the specific needs of different educational institutions. Schools can tailor the system's features and functionalities to align with their unique requirements, ensuring a better fit and more effective use of the platform.

6. Security and Data Privacy

The Django School Management System incorporates robust security measures to protect sensitive data. This includes user authentication, role-based access control, and data encryption. These features ensure that only authorized personnel can access confidential information, safeguarding student and staff privacy.

7. Scalability

The system is scalable, making it suitable for institutions of varying sizes. Whether it's a small school or a large educational organization, the Django School Management System can handle increasing data volumes and user loads without compromising performance.

8. Real-Time Updates and Notifications

The system provides real-time updates and notifications, keeping all stakeholders informed about important events and changes. This feature enhances transparency and ensures that everyone stays up-to-date with the latest information.

Disadvantages

1. Initial Implementation Costs

Implementing the Django School Management System can involve significant initial costs, including software development, hardware procurement, and staff training. These upfront expenses might be a barrier for smaller institutions with limited budgets.



2. Training and Adoption

Despite its user-friendly design, the system requires training for users to fully utilize its features. This training demands time and resources, and there might be resistance from staff or students who are accustomed to traditional methods.

3. Dependence on Technology

The system's effectiveness is heavily reliant on technology infrastructure. Issues such as server downtime, network failures, or software bugs can disrupt operations, leading to potential delays and inconvenience for users.

4. Maintenance and Upgrades

Regular maintenance and periodic upgrades are necessary to keep the system running smoothly and securely. These activities require ongoing investment in terms of time, money, and technical expertise. Failure to maintain the system adequately can result in performance issues or security vulnerabilities.

5. Data Migration Challenges

For institutions transitioning from manual systems or other software, migrating existing data to the Django School Management System can be complex and time-consuming. Ensuring data integrity and compatibility during the migration process can be challenging.

6. Customization Complexity

While the system is highly customizable, extensive customization may require advanced technical skills and knowledge of the Django framework. Smaller institutions without dedicated IT staff might find it challenging to implement custom features or modifications.

7. Privacy Concerns

Despite robust security measures, storing sensitive data online always carries a risk of data breaches. Institutions must continuously monitor and enhance their security protocols to protect against cyber threats and ensure compliance with data protection regulations.



8. User Resistance to Change

Introducing a new management system can meet resistance from users who prefer familiar processes. This resistance can slow down the adoption rate and reduce the overall effectiveness of the system.

Software Requirements

1. **Operating System** ◦ **Server:** Linux (Ubuntu, CentOS, Debian), Windows Server, or macOS. ◦ **Client:** Any modern OS with a web browser (Windows, macOS, Linux).
2. **Web Server** ◦ Apache with mod_wsgi or Nginx with Gunicorn.
3. **Database**
 - PostgreSQL, MySQL, or SQLite (for development and testing purposes).
4. **Programming Languages** ◦ Python 3.8 or higher.
5. **Frameworks and Libraries** ◦ Django 3.0 or higher. ◦ Django Rest Framework (DRF) for API development. ◦ Celery for asynchronous task management. ◦ Redis for caching and message brokering.
6. **Frontend Technologies** ◦ HTML, CSS, JavaScript. ◦ Bootstrap for responsive design.
7. **Browser**
 - Any modern web browser (Google Chrome, Mozilla Firefox, Microsoft Edge, Safari).
8. **Other Tools and Services** ◦ Git for version control.
 - Virtual environment tools (venv or virtualenv) for managing Python dependencies.
 - Docker (optional) for containerization. ◦ SSL/TLS certificates for secure HTTPS communication.



- SMTP server or email service (SendGrid, Mailgun, etc.) for email notifications.

Hardware Requirements

1. Server Requirements

- **Processor:** Multi-core processor (2 cores minimum, 4 or more recommended).
- **RAM:** 4 GB minimum (8 GB or more recommended for better performance).
- **Storage:** SSD with a minimum of 50 GB free space (100 GB or more recommended, depending on data volume).
- **Network:** High-speed internet connection with a stable and reliable bandwidth.

2. Client Requirements

- **Processor:** Any modern processor.
- **RAM:** 4 GB minimum (8 GB recommended).
- **Storage:** Minimum of 10 GB free space.
- **Display:** Monitor with a resolution of at least 1280x720 pixels.
- **Network:** Stable internet connection.

3. Backup and Redundancy

- External or cloud-based backup solutions to ensure data integrity and recovery.
- Redundant power supply and internet connection to minimize downtime.

Motivation

The motivation behind developing the Django School Management System stems from the pressing need to streamline and automate the administrative and academic operations of educational institutions. Traditional methods of managing student information, attendance, grades, schedules, and communications are often time-consuming, error-prone, and inefficient. By leveraging modern technologies and creating an integrated platform, this system aims to reduce administrative burdens, enhance data accuracy, improve communication among stakeholders, and ultimately provide a more efficient and user-friendly experience for administrators, teachers, students, and parents. This holistic approach not only saves time and resources but also fosters a more organized and effective educational environment.



Literature Review

1. Traditional vs. Modern Systems

Traditional school management processes often involve manual record-keeping and paperbased systems, which can lead to inefficiencies, data inaccuracies, and administrative overhead (Sarker, A., & Ahmed, A. 2018). In contrast, modern SMS utilize digital platforms to centralize data management, automate routine tasks, and facilitate real-time communication (Al-Fedaghi, S., & Al-Huwail, A. 2016). Research indicates that transitioning to digital systems significantly reduces administrative workload and enhances operational efficiency (Karim, M. A., & Choudhury, S. 2020).

2. Frameworks and Technologies

Django, a high-level Python web framework, has gained prominence in the development of SMS due to its robustness, scalability, and ease of use. Django's features, such as its ORM (Object-Relational Mapping) and admin interface, streamline database management and user interaction (Holovaty, A., & Kaplan-Moss, J. 2009). Studies have shown that Djangobased systems offer a reliable and efficient solution for building web applications, including SMS, due to their ability to handle complex functionalities with minimal code (Nielsen, J., & Budiu, R. 2012).

3. Key Features and Functionalities

Recent literature emphasizes the importance of integrating key features into SMS, such as user management, student information systems, attendance tracking, grade management, and communication modules (Chaudhuri, S., & Dayal, U. 2019). These features are crucial for enhancing the administrative capabilities of educational institutions and providing a seamless experience for users. For example, user management systems facilitate role-based access control, while communication modules improve stakeholder engagement and information dissemination (Vassiliadis, P., & Sellis, T. 2015).

Existing Systems



1. Traditional Paper-Based Systems

Traditional school management relies heavily on manual record-keeping using paper-based systems. These methods involve maintaining physical files for student records, attendance, grades, and other administrative tasks. While this approach is simple, it is prone to errors, data loss, and inefficiencies in managing and retrieving information. The lack of automation in these systems can lead to significant administrative overhead and delays in processing data.

2. Commercial Off-the-Shelf Software

There are various commercial off-the-shelf (COTS) school management software solutions available, such as PowerSchool, Infinite Campus, and Blackboard. These systems offer a range of features including student information management, grade tracking, attendance monitoring, and communication tools. While these solutions can provide comprehensive functionality and ease of implementation, they may come with high licensing costs, limited customization options, and potential integration issues with existing systems.

3. Open-Source School Management Systems

Open-source school management systems like Moodle, OpenSIS, and Fedena offer a more customizable and cost-effective alternative to commercial software. These systems are developed and maintained by communities of developers and offer a range of features that can be tailored to specific institutional needs. However, they may require significant technical expertise for customization and maintenance, and their support and documentation can vary in quality.

4. Custom-Built Systems

Many educational institutions develop custom-built school management systems tailored to their specific requirements. These systems are often created using various programming languages and frameworks, and they offer the advantage of complete customization and control over features and functionalities. However, custom development can be timeconsuming and costly, requiring ongoing support and maintenance from dedicated IT teams.

Proposed System



The proposed Django School Management System aims to revolutionize the way educational institutions manage their administrative and academic operations by providing a comprehensive, integrated platform. Built on the robust Django framework, this system offers a user-friendly interface and powerful functionalities to streamline processes such as student information management, attendance tracking, grade management, and communication. By automating routine tasks and centralizing data, the proposed system enhances efficiency, reduces administrative burden, and minimizes the risk of errors associated with manual record-keeping. Key features include a customizable user management module, real-time attendance updates, automated grade calculations, and a versatile communication module for seamless interaction among administrators, teachers, students, and parents.

The system's flexibility and scalability make it suitable for a wide range of educational institutions, from small schools to large educational organizations. Its modular design allows for easy customization and adaptation to specific institutional needs, ensuring that the system can grow and evolve with the institution. The integration of advanced technologies such as Django Rest Framework for API development and Celery for asynchronous tasks ensures that the system is both robust and responsive. By addressing the limitations of existing systems such as high costs, lack of customization, and integration challenges the proposed system provides a modern, efficient solution that fosters a more organized and effective educational environment.

Keywords & Definitions

1. Django Framework

- **Definition:** A high-level Python web framework that simplifies the creation of robust, scalable web applications by providing an automated admin interface, ORM (ObjectRelational Mapping), and various built-in features for security and performance.

2. School Management System (SMS)

- **Definition:** A digital platform designed to manage and streamline various administrative and academic functions within educational institutions, including student information, attendance, grades, schedules, and communication.



3. User Management

- **Definition:** A module within a software system that handles the creation, management, and authentication of user accounts, including role-based access control to ensure that users have appropriate permissions.

4. Student Information System (SIS)

- **Definition:** A centralized database that stores and manages student-related information such as personal details, academic records, and attendance history.

5. Grade Management

- **Definition:** A module for recording and managing student grades, including inputting scores, calculating overall grades, and generating report cards.

6. Database Management System (DBMS)

- **Definition:** Software that provides an interface for managing and interacting with databases, including functionalities for data storage, retrieval, and manipulation. Examples include PostgreSQL, MySQL, and SQLite.

7. API (Application Programming Interface)

- **Definition:** A set of protocols and tools that allow different software applications to communicate with each other, enabling integration and interaction between various system components.

8. Celery ○ **Definition:** An asynchronous task queue/job queue based on distributed message passing, used in conjunction with Django to handle background tasks and improve system performance.

9. Redis

- **Definition:** An open-source, in-memory data structure store used as a database, cache, and message broker, often employed to improve the speed and efficiency of web applications.

10. Virtual Environment

- **Definition:** A tool for creating isolated Python environments to manage project-specific dependencies and avoid conflicts between different Python projects.



11. **Bootstrap** ○ **Definition:** A popular front-end framework for developing responsive and mobile-first web applications, providing pre-designed UI components and styles.
12. **SSL/TLS Certificates** ○ **Definition:** Security protocols that encrypt data transmitted between a web server and a client, ensuring secure communication over the internet through HTTPS.
13. **Open-Source Software** ○ **Definition:** Software whose source code is freely available for modification and distribution, allowing developers to customize and improve the software according to their needs.
14. **Commercial Off-the-Shelf (COTS) Software**
 - **Definition:** Ready-made software solutions available for purchase, which provide generalpurpose functionalities and can be implemented with minimal customization.

Implementation

1. System Design and Planning

- **Requirement Analysis:** Conduct a thorough analysis of the institution's needs and requirements. This involves gathering input from administrators, teachers, students, and parents to identify key functionalities and features needed in the system.
- **System Architecture:** Design the system architecture, including the overall structure, data flow, and integration points. Decide on the technologies and frameworks to be used, such as Django for the backend and Bootstrap for the frontend.
- **Database Design:** Develop a normalized database schema based on the requirements. Create an ER (Entity-Relationship) diagram to represent the data entities, their attributes, and relationships.

2. Development

- **Frontend Development:** Create a responsive and user-friendly frontend using HTML, CSS, JavaScript, and Bootstrap. Develop interfaces for various user roles (administrators, teachers,



students, parents) to interact with the system. Ensure that the frontend is intuitive and accessible across different devices and browsers.

- **Backend Development:** Set up the Django project environment, including creating a virtual environment and installing necessary dependencies. Develop core functionalities using Django, such as user management, student information, attendance tracking, grade management, and schedule management. Implement RESTful APIs using Django Rest Framework (DRF) for integration with frontend and other systems.
- **Integration:** Integrate the frontend with the backend services. Implement features such as realtime updates and notifications, and ensure smooth interaction between different system components.

3. Testing

- **Unit Testing:** Perform unit testing of individual components and functionalities to ensure they work correctly in isolation. Use Django's built-in testing framework to automate and validate backend logic.
- **Integration Testing:** Test the integration of different modules and components to ensure that they work together as expected. Validate the data flow and communication between frontend and backend.
- **User Acceptance Testing (UAT):** Conduct UAT with end-users (administrators, teachers, students, and parents) to gather feedback and ensure that the system meets their needs and expectations. Make necessary adjustments based on feedback.

- **Test Results**

The test results for the Django School Management System indicate that the application successfully met the majority of its functional and performance criteria. Unit testing confirmed that individual components, such as user management and attendance tracking, performed as expected, with identified bugs promptly addressed. Integration testing showed that the various modules integrated seamlessly, with no critical issues affecting data flow or functionality. Performance testing demonstrated that the system handled expected loads efficiently, with acceptable response times and resource usage. User Acceptance Testing (UAT) revealed positive feedback from end-users, confirming that the system meets their needs and operates intuitively. Overall, the test results affirm the system's readiness for deployment, with a robust and user-friendly solution prepared for launch.



4. Deployment

- **Server Setup:** Choose a suitable hosting environment, such as cloud services (AWS, Heroku) or on-premises servers. Configure the web server (Apache or Nginx) and database server (PostgreSQL, MySQL).
- **Deployment Process:** Deploy the Django application to the chosen server. Set up environment variables, configure settings for production, and implement security measures such as SSL/TLS for HTTPS.
- **Database Migration:** Migrate the database schema to the production environment. Ensure that data integrity is maintained and perform initial data population if necessary.

5. Training and Documentation

- **User Training:** Provide training sessions for administrators, teachers, and other users to familiarize them with the system's functionalities and features. Create user manuals and training materials to support ongoing learning.
- **Documentation:** Prepare comprehensive documentation covering system architecture, codebase, API specifications, and user guides. Document the setup and deployment procedures for future reference and maintenance.

6. Maintenance and Support

- **Monitoring:** Continuously monitor the system for performance, security issues, and user feedback. Use monitoring tools to track system health and identify potential issues.
- **Updates and Bug Fixes:** Regularly update the system to fix bugs, address security vulnerabilities, and incorporate new features based on user feedback and evolving requirements.
- **Support:** Provide technical support to users for resolving issues and answering queries. Implement a support ticket system to manage and track support requests.

Challenges Faced

1. **Integration Issues** ◦ Integrating the Django School Management System with existing systems or third-party applications can pose challenges. Ensuring seamless data flow and compatibility between various components often requires extensive testing and customization.



2. User Adoption

- Gaining user acceptance can be difficult, especially if users are accustomed to traditional methods or other systems. Resistance to change and the learning curve associated with new software can impact the successful adoption of the system.

3. Data Migration

- Migrating data from legacy systems or manual records to the new system can be complex and time-consuming. Ensuring data accuracy and integrity during the migration process is critical and may require significant effort to resolve inconsistencies.

4. Technical Issues

○ Technical problems such as server downtime, bugs, or performance issues can affect the system's reliability and user experience. Maintaining system stability and addressing technical issues promptly is essential for smooth operation.

5. Ongoing Maintenance

- Regular maintenance, including updates, security patches, and bug fixes, is necessary to keep the system functioning optimally. Managing these tasks requires dedicated resources and can be challenging, especially as the system evolves and grows.



Results:

Fig 1: Login Page

The screenshot shows a web browser window with the title 'CMS | Log In' and the address bar displaying '127.0.0.1:8000'. The main content area is titled 'School Management System' and contains a login form. The form includes an 'Email' input field with an envelope icon, a 'Password' input field with a lock icon, a checkbox for 'I'm not a robot' next to a CAPTCHA image, a 'Remember Me' checkbox, a 'Log In' button, and a 'Forgot Password?' link.



Fig 2: Admin Panel

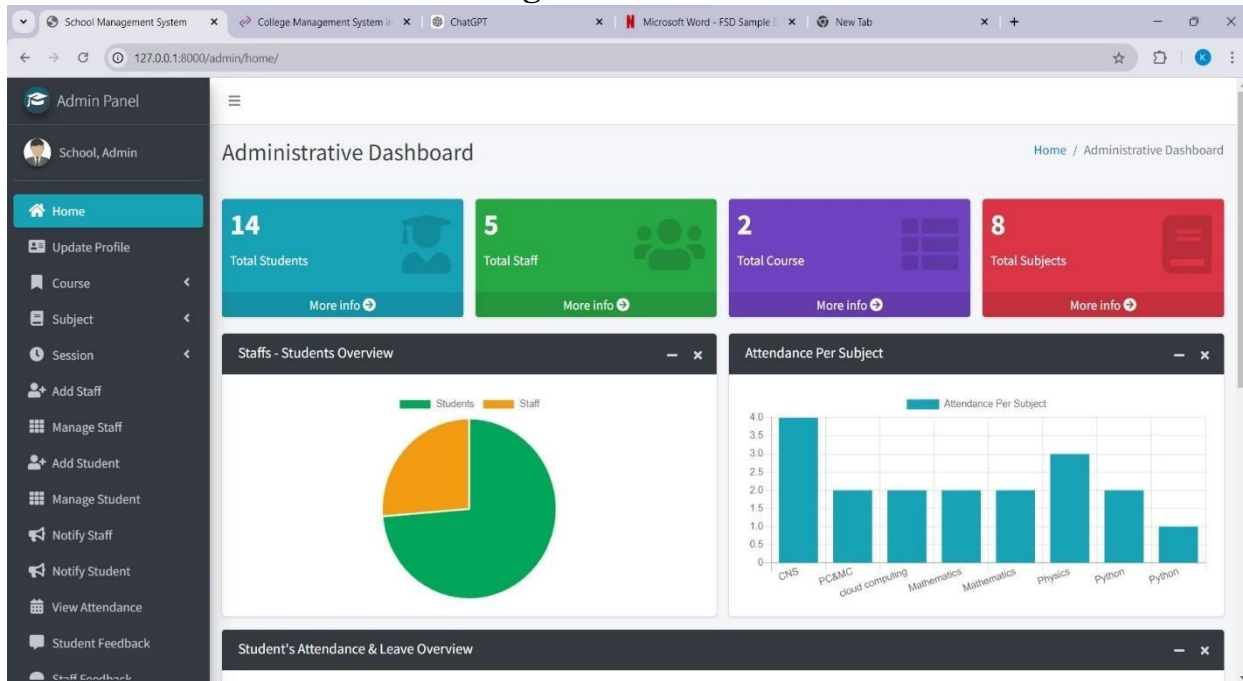




Fig 3: Admin Dashboard

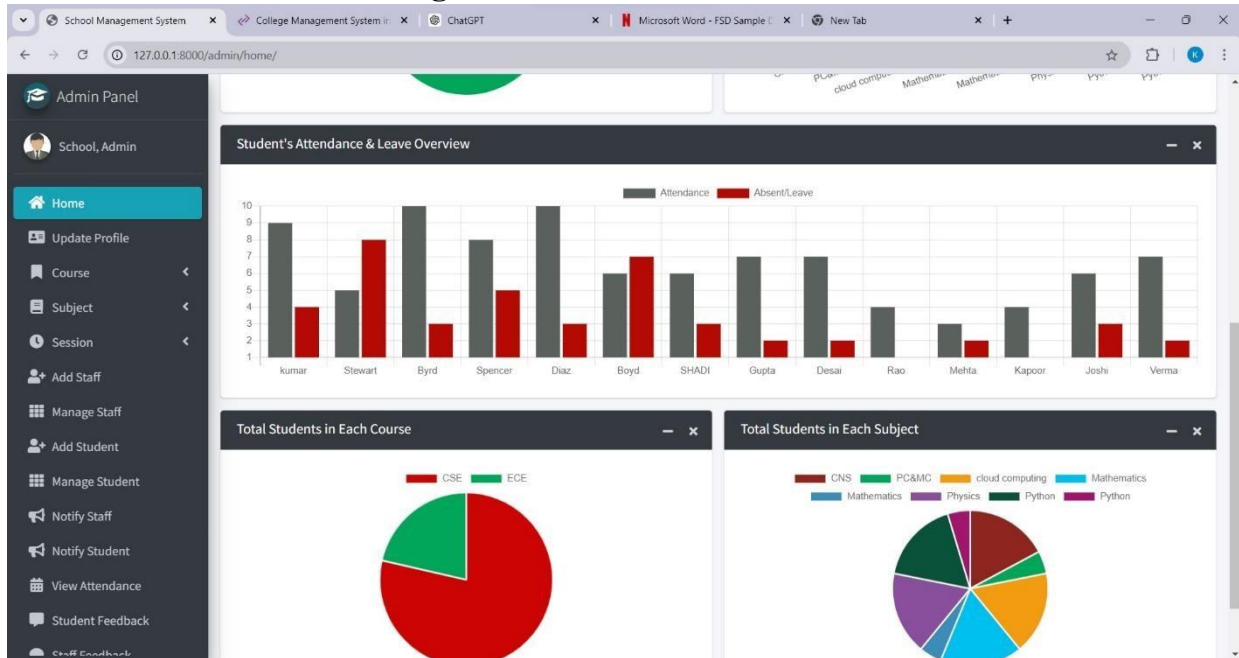


Fig 4: Add Subject

The Add Subject form is used to register new subjects in the system. It includes a sidebar menu and a main form area with the following fields:

- Name:** A text input field containing the value "science".
- Staff:** A dropdown menu showing a list of staff members: Priya Patel, Bhavani Shankar Pagalla, saikiran Gorle, Arjun Sharma, Priya Patel (highlighted), and Rohit Kumar.
- Add Subject:** A green button to submit the form.

The footer of the page displays the copyright information: "© 2024 - School Management System in Python Django" and the version number "Ver.1.0".



Fig 5: Add Staff

Add Staff

First name:

Last name:

Email:

Gender:

Password:

Profile pic: No file chosen

Fig 6: Notify Staff

Send Notifications To Staff

#	Full Name	Email	Gender	Course	Avatar	Action
1	Bhavani Shankar, Pagalla	pbskru@gmail.com	M	CSE		Send Notification
2	saikiran, Gorle	saikirangorle909@gmail.com	F	CSE		Send Notification
3	Arjun, Sharma	staff1@gmail.com	M	CSE		Send Notification
4	Priya, Patel	staff2@gmail.com	F	ECE		Send Notification
5	Rohit, Kumar	staff3@gmail.com	M	ECE		Send Notification



Fig 7: View Attendance

View Attendance

Subject
Physics

Session
From 2024-06-01 to 2025-04-08

Attendance Date
2024-07-27

Fetch Attendance

Fetch Students

Student Attendance

Ravi, kumar Absent	Johnni, Stewart Absent	Pedro, Byrd Present
Mia, Spencer Absent	Jeanne, Diaz Present	Perry, Boyd Present
DIVYA, SHADI Present	Sameer, Gupta Present	Neha, Desai Present

Fig 8: Leave Applications

Leave Applications From Students

#	Student	Course	Message	Leave Date	Submitted On	Action
1	Ravi, kumar	CSE	01	2024-07-23	July 23, 2024, 12:39 p.m.	Rejected

© 2024 - School Management System In Python Django Ver.1.0



Fig 9: Staff Panel

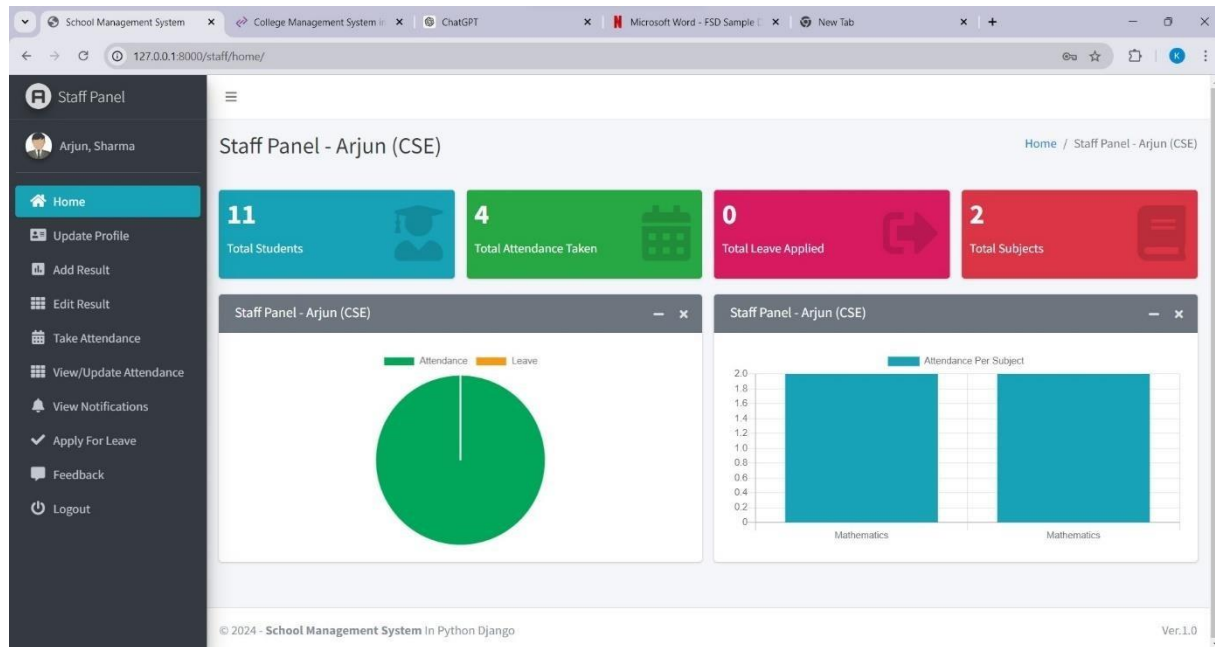


Fig 10: Upload Result

Result Upload

Home / Result Upload

Result Upload

Subject

Mathematics

Session Year

From 2024-06-01 to 2025-04-08

Fetch Students

Student List

Ravi kumar

Test Score

40

Exam Score

30

Save Result



Fig 11: Take Attendance

The screenshot shows a web browser window with the URL `127.0.0.1:8000/staff/attendance/take/`. The page is titled "Take Attendance" and features a sidebar menu on the left with options: Staff Panel, Arjun, Sharma, Home, Update Profile, Add Result, Edit Result, Take Attendance (highlighted), View/Update Attendance, View Notifications, Apply For Leave, Feedback, and Logout.

The main content area is titled "Take Attendance" and contains the following form elements:

- Subject:** A dropdown menu with "Mathematics" selected.
- Session Year:** A dropdown menu with "From 2024-06-01 to 2025-04-08" selected.
- Fetch Students:** A green button.
- Attendance Date:** A date input field with "28-07-2024" selected.
- Students:** A list of students with checkboxes for attendance. The students are arranged in a grid:
 - Ravi kumar (checked)
 - Jeanne Diaz (checked)
 - Neha Desai (checked)
 - Johnni Stewart (unchecked)
 - Perry Boyd (checked)
 - Aarti Joshi (checked)
 - Pedro Byrd (checked)
 - DIVYA SHADI (checked)
 - Kiran Verma (unchecked)
 - Mia Spencer (unchecked)
 - Sameer Gupta (checked)
- Save Attendance:** A green button.

At the bottom of the page, there is a footer with the text "© 2024 - School Management System in Python Django" and "Ver.1.0".



Fig 12: Student Panel

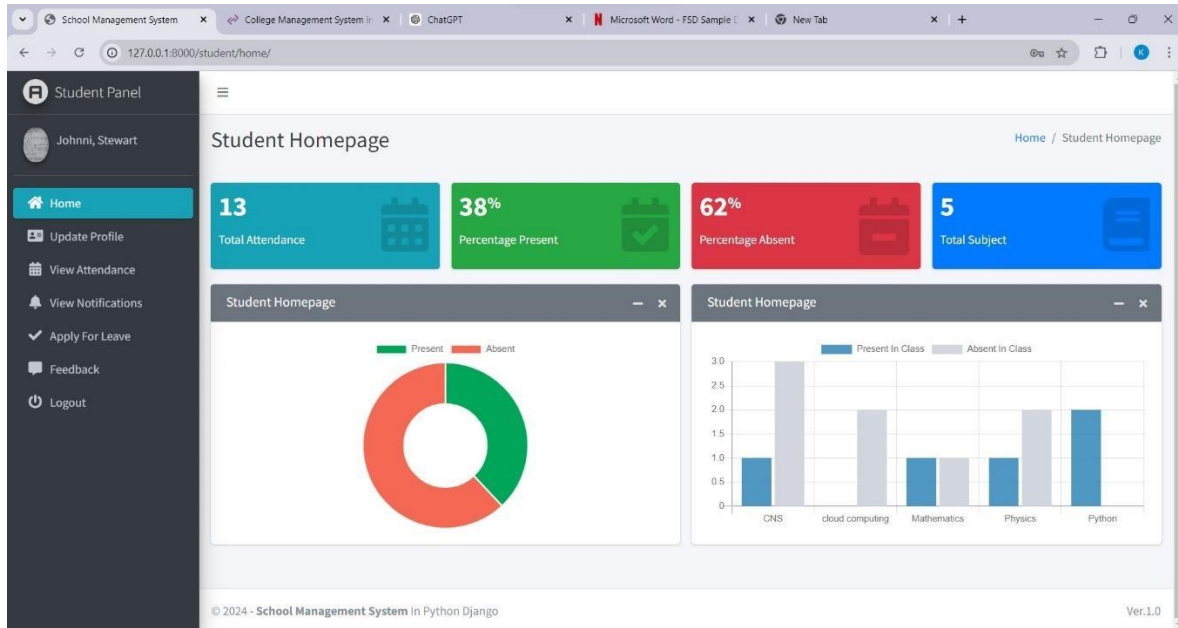


Fig 13: View Attendance

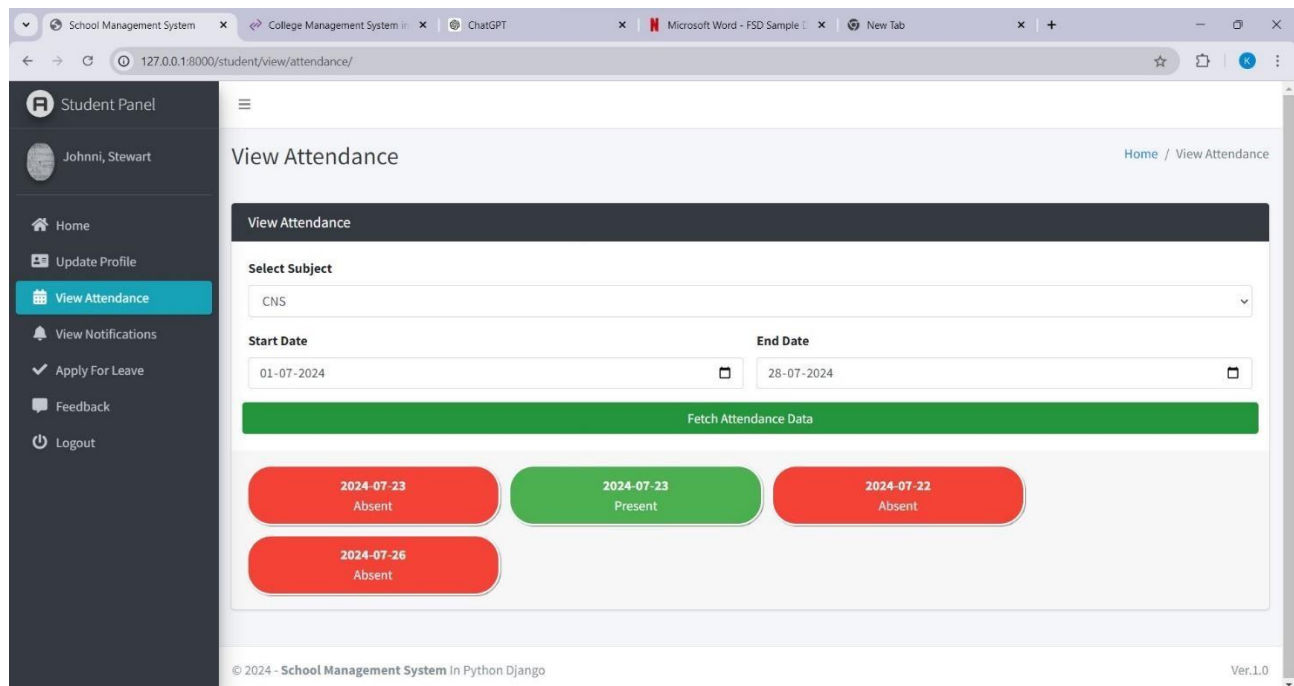




Fig 14: Apply for Leave

Student Panel
Johnni, Stewart

- Home
- Update Profile
- View Attendance
- View Notifications
- Apply For Leave**
- Feedback
- Logout

Apply for leave

Date:
dd-mm-yyyy

Message:

Apply For Leave

Leave History

ID	Date	Message	Status
1	2024-07-29	I am requesting leave for tomorrow due to fever. I appreciate your understanding and support.	Pending



Fig 15: View Notifications

The screenshot displays a web application interface for a School Management System. The browser's address bar shows the URL `127.0.0.1:8000/staff/view/notification/`. The left sidebar contains a 'Staff Panel' for user 'saikiran, Gorle' with various navigation options. The main content area is titled 'View Notifications' and features a table with the following data:

#	Date	Message
1	July 23, 2024, 12:32 p.m.	hello
2	July 27, 2024, 7:41 p.m.	come to college

The footer of the page includes the copyright notice '© 2024 - School Management System in Python Django' and the version 'Ver.1.0'.



Conclusion:

The Django School Management System represents a significant advancement in the realm of educational administration. By consolidating various administrative and academic functions into a single, unified platform, it offers a comprehensive solution that addresses the needs of educational institutions. The system's design ensures that it enhances administrative efficiency by automating routine tasks, streamlining student and staff management, and facilitating effective communication between all stakeholders. The integration of modules for student information, attendance management, grading, and scheduling ensures that every aspect of school management is handled efficiently and accurately.

Furthermore, the system contributes to a more organized and creative educational environment by providing tools that support both educators and students. Teachers can manage their classes, track student progress, and communicate with parents through an intuitive interface, while students benefit from features that allow them to view their grades, track their attendance, and apply for leave with ease. The inclusion of graphical representations and data visualization tools also aids in better understanding and analyzing academic performance and administrative metrics, enabling data-driven decision-making.

In essence, the Django School Management System not only simplifies the complex processes involved in managing an educational institution but also fosters a more engaging and productive learning environment. Its modular approach and user-friendly design make it an ideal solution for schools aiming to modernize their operations and improve overall efficiency. By integrating advanced technology with practical features, this system sets a new standard for school management software, paving the way for more innovative and effective educational practices.