SCH '2026

SUDARSANAM R	TEAM ID
PS09(EXAM CELL)	TEAM MEMBERS
AUTOMATED UNIVERSITY THEORY EXAMINATION SEATING ARRANGEMENT SYSTEM	KANAGARAJ M PRIME R S SANJITH N C SRIRAM T

AUTOMATED UNIVERSITY THEORY EXAMINATION SEATING ARRANGEMENT SYSTEM

Manual Inefficiency: Traditional manual seating arrangement is time-consuming, tedious, and prone to human error, especially for large universities.

Malpractice Prevention: Manually ensuring that no two students appearing for the same subject sit adjacent to each other (to prevent copying) is extremely complex.

Data Handling: Managing thousands of student records, subjects, and hall tickets manually is chaotic.

Lack of audit trails: paper-based records are hard to verify post-exam.

PROBLEMS

- Manual Inefficiency
- High Risk of Human Error
- Suboptimal Resource Management
- Information Dissemination Delays
- Malpractice Vulnerability
- Rigid Infrastructure Management
- Inconsistent Spacing Protocols
- Data Silos & Fragmentation

ABSTRACT

Frontend (React/TypeScript): A modern Single Page Application (SPA) built with **Vite** and **Tailwind CSS**. It provides a high-performance administration dashboard for hall management and real-time seating visualization.

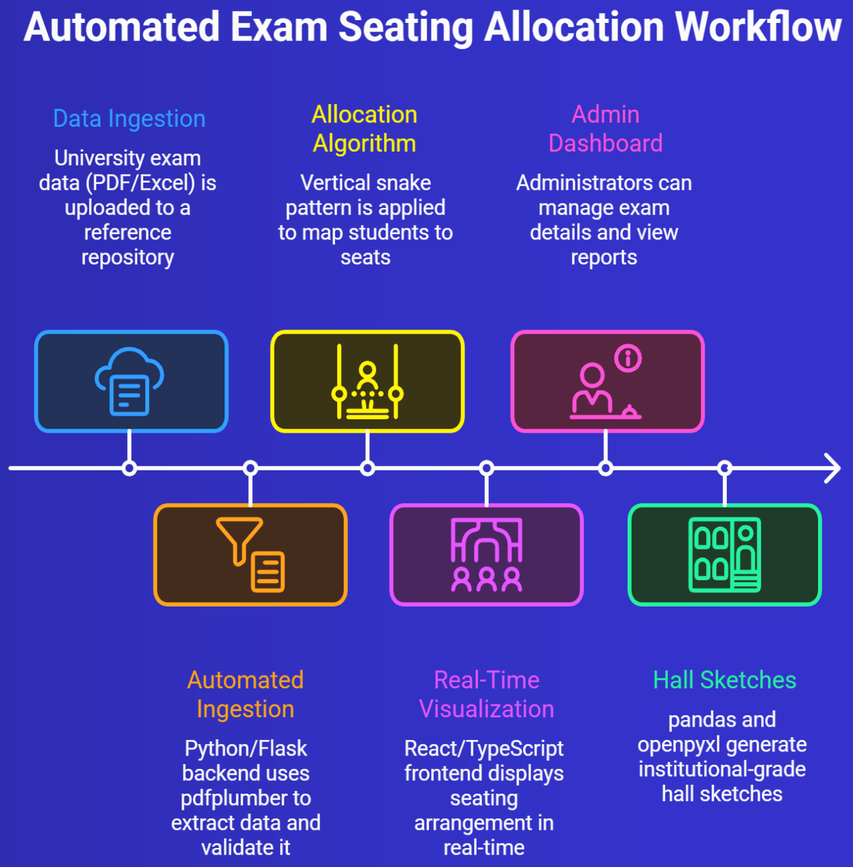
Backend (Python/Flask): A modular RESTful API serving as the computation engine. It handles complex PDF parsing, executes the allocation algorithms, and generates formatted Excel reports.

Data Layer (Reference): Integrates real-world university exam data (PDF/Excel) for validation, utilizing a dedicated /Reference repository for test-driven development and accuracy benchmarking.

Automated Ingestion: Backend uses PDF plumber

Excel Engine: Uses pandas and open pyxl to generate institutional-grade **"Hall Sketches“**

Vertical Snake Pattern: Fills the hall grid in a zigzag column-wise pattern, providing superior physical separation compared to linear filling.



PROPOSED SOLUTION

1) Intelligent Data Ingestion (One-Click ETL)

Process: Drag-and-drop interface for University PDFs.

Mechanism: Automated parsing of Student Names, Reg Nos, and Schedules using a custom **Regex Engine**, eliminating manual data entry.

2) Algorithmic Allocation Engine (The Core)

Vertical Snake Traversal: Disrupts physical lines of sight by alternating filling directions (Top-Down / Bottom-Up).

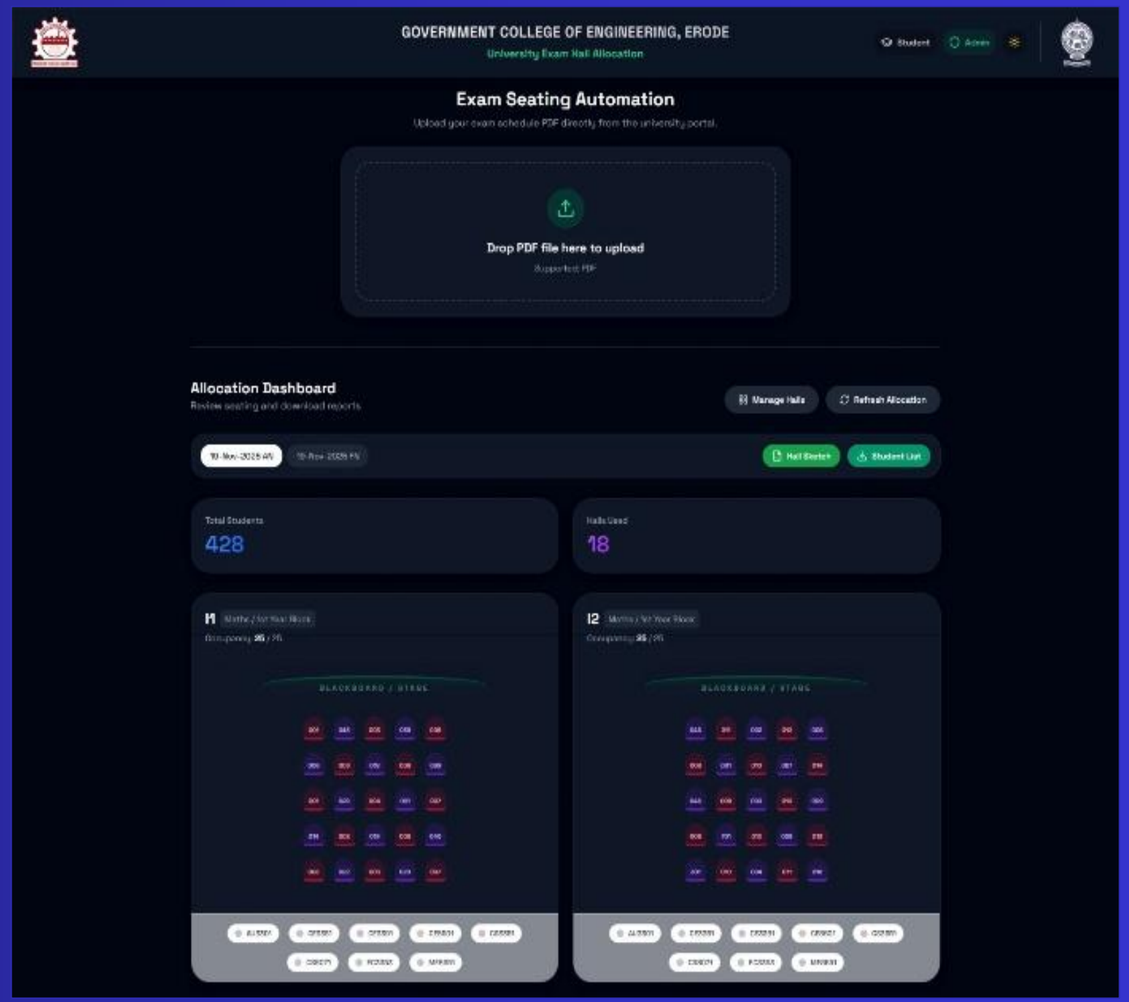
Subject Interleaving: Mathematically separates students by Subject and Department (e.g., Mixes CSE and IT) to prevent peer clusters.

Smart Spacers: Automatically injects empty "buffer seats" if subject conflicts are detected in the queue.

3) Digital "Hall Sketch" & Reporting

Outputs: Generates formatted **Physical Hall Layouts** (Excel/PDF) ready for immediate printing.

Control Tower: Centralized Dashboard for real-time tracking of hall utilization and capacity management



METHODOLOGY

Methodology: Technical Workflow

1. Data Ingestion (ETL)

- **Process :**Automated PDF parsing using PDF Plumber + pypdf
- **Logic :**Extracts student metadata and subject codes; validates data integrity.

2. Infrastructure Mapping

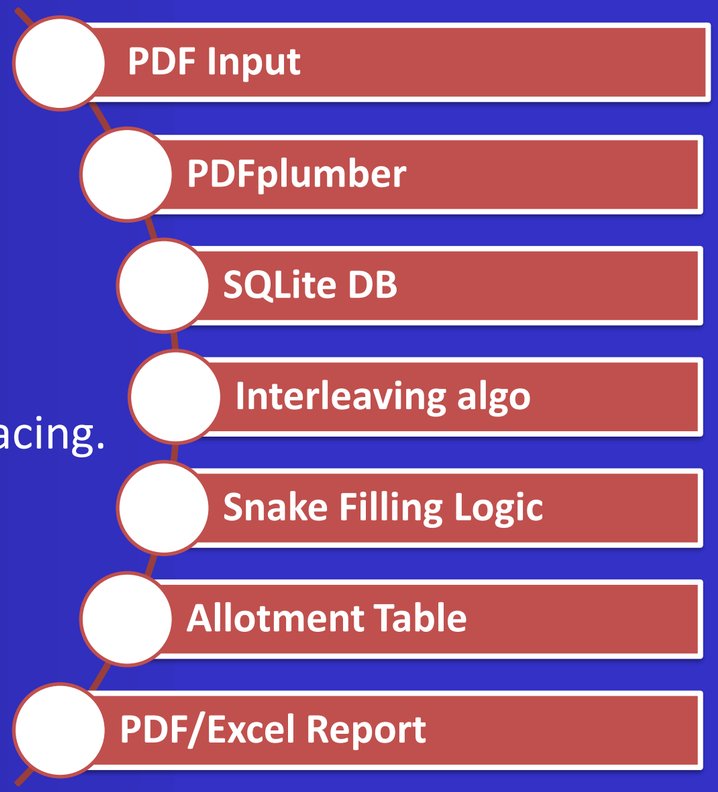
- **Configuration :** Digital modeling of physical halls (Rows X columns).

3. Vertical Snake Engine (Core Logic)

- **Interleaving: Round-Robin** selection from department "buckets" to maximize spacing.
- **Traversal:** Serpentine filling (Odd Cols: Top-Down | Even Cols: Bottom-Up).
- **Security:** Physically disrupts lines of sight to neutralize malpractice.

4. Visualization & Reporting

- **Analytics:** Real-time occupancy metrics via **Next.js 15** dashboard.
- **Deliverables:** Automated **PDF Seating Grids** and **Excel Attendance Sheets**.



THANK YOU