FORGE 1.0 Buildathon

Campus Operations Challenge

Official Problem Statement

Event Date	Build Window	Submission Deadline
February 15, 2026	10:00 AM – 2:00 PM *(4 hours)*	2:15 PM *sharp*
Team Size: 2–3 members		

Case Background

NIT Jalandhar functions like a small city.

Every day, thousands of students, faculty members, clubs, hostel staff, and administrators depend on numerous operational processes such as classroom allocation, lab access, event coordination, maintenance requests, resource sharing, approvals, and transportation.

Despite this scale, most of these processes are still managed manually through WhatsApp messages, spreadsheets, calls, and informal follow-ups.

This leads to recurring issues:

- missed information
- delayed responses
- double bookings and conflicts
- lack of visibility into status
- repeated manual coordination
- unnecessary time loss for students and staff

Individually, these problems appear minor. Collectively, they create significant inefficiency across campus.

Many of these inefficiencies do not require large platforms to fix.

They require small, reliable automation systems that eliminate repetitive manual work.

Scenario

In response to these challenges, a student-initiated endeavor designated ***CampusFlow*** has been established. The CampusFlow methodology is predicated upon simplicity and efficacy:

- Identification of operational bottlenecks
- Development of targeted solutions
- Automation of repetitive processes
- Delivery of immediate, measurable operational impact

You are joining CampusFlow as the inaugural technical team. Your mandate encompasses the conceptualization and development of the first operational tools designed to enhance campus system functionality.

The Challenge

Your Objective

Identify a specific operational inefficiency within the campus environment and architect a fully functional automated system that provides comprehensive, end-to-end resolution.

Your system must demonstrably incorporate the following components:

- Data or request ingestion mechanism
- Backend processing infrastructure or decision logic
- Automated actions or trigger mechanisms
- Quantifiable and verifiable outcome

Input → Processing → Automation → Result

Only fully operational systems will be considered complete. Conceptual frameworks, visual mockups, or presentation-only proposals are unacceptable.

Problem Domains (Illustrative Examples)

You may select any authentic operational challenge. The following categories serve as illustrative examples:

Academic Operations
Classroom availability management, laboratory or equipment reservation conflicts, deadline oversight, dispersed announcements, assignment coordination inefficiencies.

Campus Resources
Library seating transparency, facility reservation systems, printer or consumable inventory shortages, Wi-Fi incident reporting, shared resource tracking mechanisms.

Hostel and Mess Operations
Maintenance requisition workflows, food quality feedback systems, guest meal coordination, occupancy visibility, communication latency.

Clubs and Events
Manual registration processes, venue scheduling conflicts, attendance monitoring, calendar clashes, feedback aggregation.

Transport and Navigation
Unpredictable bus schedules, vehicle availability uncertainty, parking space visibility, campus wayfinding challenges.

Administration
Manual approval workflows, document verification delays, opaque request status tracking, repetitive follow-up requirements.

You are not constrained by these examples. Any operational inefficiency demonstrating tangible impact is acceptable.

Minimum System Requirements

Your solution must incorporate the following foundational elements:

- Backend processing logic (not merely frontend interfaces)
- Database or persistent storage infrastructure
- Genuine automation that substantively eliminates manual intervention
- Comprehensive, end-to-end workflow implementation

The system must execute reliably during demonstration. Any workflow component that remains manual, simulated, or hardcoded renders the solution incomplete.

Exclusions

Avoid the following approaches:

- Comprehensive "all-in-one" management platforms
- Feature-saturated systems
- Authentication boilerplate lacking substantive core logic
- UI-centric mockups utilizing static data
- Generic applications without demonstrated operational impact
- Solutions of excessive scope, precluding completion within the four-hour timeframe

Prioritize solving one workflow exceptionally well over addressing multiple workflows inadequately.

Deliverables

Submit before 2:15 PM sharp:

1. Presentation (PDF, 3–5 slides)

- Problem statement articulation
- Solution architecture overview
- Automation mechanism explanation
- Demonstration link
- GitHub repository URL
- Live deployment link (optional)

2. Demonstration Video (3–5 minutes)
Demonstrate the following operational components:

- Input mechanism
- Backend processing workflow
- Automation execution
- Final outcome delivery

3. GitHub Repository

- Functional source code
- Comprehensive README with setup instructions

Late submissions will not be accepted under any circumstances.

Final Instruction

Identify a genuine problem.

Architect a pragmatic system.

Demonstrate its operational viability.

FORGE 1.0 — Build Real. Ship Fast.