Feature -1

Based on your feature idea for **Passive Student Performance Monitoring**, here's a concise breakdown of what you should include after adding assignments and quizzes to your Campus Assistant application:

Core Components to Implement

1. Assignment & Quiz Management System

- Assignment submission tracking (due dates, submission status, timestamps)
- Quiz results storage (scores, completion dates)
- Grade recording and history
- Late submission detection and logging

2. Real-Time Risk Meter Dashboard Widget

- **Visual indicator** (color-coded: green/yellow/red) showing current academic health
- **Risk factors display**:
  - Number of late submissions in last X assignments
  - Current grade trend (improving/declining)
  - Upcoming deadline pressure score
  - Quiz performance consistency
- **Auto-refresh** based on new submissions or grades

3. Performance Analytics & Forecasting

- **Data collection**: Historical grades, submission patterns, quiz scores
- **Lightweight ML model**: Linear or polynomial regression to predict next assignment/quiz grade
- **Trend visualization**: Line charts showing grade trajectory over time
- **Confidence indicators**: Show prediction accuracy/reliability

4. Prescriptive Recommendation Engine

- **Smart suggestions** triggered by risk thresholds:
  - Specific handbook chapters related to struggling subjects
  - Relevant library books for weak topics
  - Past high-performing assignments for reference
  - Study group recommendations
- **Proactive alerts**: Push notifications when risk level increases
- **Action tracking**: Monitor if student followed suggestions and measure impact

5. Integration Points with Existing Features

- **Timetable**: Highlight classes where performance is declining
- **Reminders**: Auto-create study reminders based on risk alerts
- **Chat/Knowledge Base**: Quick access to recommended resources

- **Dashboard**: Central widget showing all metrics at a glance

6. Data Privacy & Permissions

- Student consent for performance tracking
- Privacy controls for what data is monitored
- Optional sharing with advisors/mentors

Quick Implementation Priority:

1. ✅ Add Assignments & Quizzes modules first
2. ✅ Build Risk Meter widget (most visible value)
3. ✅ Implement basic forecasting
4. ✅ Add prescriptive recommendations
5. ✅ Refine ML model over time

Feature -2

AI-Powered Self-Study Planning Feature - Short Description

Here's what you should include to add this feature to your Campus Assistant application:

1. Timetable Analysis Module

- Automatically detect free time by analyzing the student's class schedule
- Identify after-college hours (evenings, weekends)
- Calculate total available study time per day and week

2. AI Study Plan Generator

- Accept inputs: available time slots, upcoming deadlines, weak subjects, student preferences
- Use AI (OpenAI, Gemini, or similar) to create personalized study schedules
- Generate specific study tasks with time allocations and recommended resources
- Prioritize based on assignment urgency and performance gaps from the Risk Meter

3. Smart Scheduling Engine

- Allocate study blocks based on topic difficulty and deadline proximity
- Incorporate break intervals (Pomodoro technique)
- Avoid overlap with college hours and existing commitments
- Optimize task order (harder topics when energy is high)

4. Study Session Components

Each planned session should include:

- Subject and specific topic to cover

- Duration (with built-in breaks)
- Linked resources (handbook chapters, library books, videos)
- Clear learning goal for the session
- Completion tracking

5. Interactive Planning Features

- Daily AI-generated suggestions based on upcoming exams/quizzes
- Adaptive rescheduling if sessions are missed
- Manual editing capability (drag-and-drop time blocks)
- Voice or chat interface for quick plan requests
- Progress visualization (completed vs. planned sessions)

6. Integration with Existing Features

- **Timetable**: Auto-sync to avoid scheduling conflicts
- **Reminders**: Convert study sessions into automated reminders
- **Knowledge Base**: Direct links to recommended study materials
- **Performance Monitoring**: Prioritize weak subjects identified by Risk Meter
- **Chat**: Ask AI for study plan adjustments via chat interface

7. Notification & Motivation System

- Pre-session reminders (15 minutes before start time)
- Completion prompts and progress tracking
- Motivational messages and streak tracking
- Weekly summary of study adherence

8. Data Storage Requirements

- Study sessions database (planned, completed, skipped)
- Student preferences (study duration, break times, preferred hours)
- AI-generated plan history for learning and improvement
- Session completion statistics for analytics

MVP (Minimum Viable Product) Flow:

1. Student clicks "Generate Study Plan"
2. System analyzes timetable and identifies free hours
3. AI generates a personalized weekly schedule with specific tasks
4. Schedule displayed in calendar view with all details
5. Student receives notifications before each session
6. Student marks sessions as complete

7. System adapts future plans based on completion patterns

Feature-3

1. Core Social Feed

- **Campus Updates**: Instagram-style posts for events, announcements, and campus news
- **Anonymous Confessions & Polls**: Students share thoughts and vote on campus topics
- **Achievement Sharing**: Post project completions, internships, certifications
- **Event Photo Gallery**: Upload and view campus event memories

2. Learning & Trends Hub (Key Addition)

- **Weekly Tech Digest**: AI-curated list of 5-7 trending topics relevant to student's major
  - Example: "Top Trends This Week: AI Agents, Rust Language, Edge Computing"
- **Recommended Learning**: 2-3 high-quality free resources per trending topic (YouTube videos, articles, GitHub repos)
- **Skill Tracker**: Students can mark skills they're learning and see who else is learning the same

3. Opportunity Board

- **Internship Alerts**: Simple listing of latest opportunities (scraped from APIs or manually added)
- **Hackathon Calendar**: Upcoming hackathons with registration links
- **Campus Events**: Jobs fairs, workshops, seminar announcements

4. Student Showcase

- **Project Gallery**: Upload project screenshots/demos with description and tech stack
- **Simple Likes/Comments**: Basic engagement on posts
- **Filter by Department**: View projects from your field of study

5. Discussion Forums

- **Subject-wise Threads**: Ask questions, share tips per course
- **Career Discussions**: Interview experiences, resume advice, job search tips
- **Simple Search**: Find past discussions on specific topics

6. Smart Features (AI-Light)

- **Personalized Feed**: Show content matching student's course and interests (basic filtering)
- **Trending Tags**: Display most-discussed topics this week (#MachineLearning #Placements #ReactJS)
- **Weekly Highlights Notification**: One push notification with "Top 3 things happening this week"

Feature -4

1. Input Collection Module

Admin provides:

- **Faculty List**: Names, subjects they can teach, availability preferences (days/time slots they can't teach)
- **Class List**: Class names/sections, total periods per week, subject requirements
- **Constraints**:
    - Number of visits per faculty to each class (e.g., 5 times/week)
    - Weekly time slots (e.g., Monday-Friday, 8 AM - 4 PM, with period durations)
    - Room availability
    - Maximum consecutive periods per faculty per day
    - Lunch breaks, buffer times

Optional Inputs:

- Faculty subject expertise matching with class needs
- Priority rules (senior faculty preferences, special accommodations)

2. Constraint Validation System

Before generating schedules, the system checks if assignment is possible:

- **Calculate total slots needed**: 10 faculty × 8 classes × 5 visits = 400 faculty-class assignments per week
- **Check against available time**: Total periods per week × number of parallel rooms
- **Validate**: Ensure no faculty is over-scheduled or under-utilized
- **Alert admin** if constraints are mathematically impossible (e.g., not enough time slots)

3. Scheduling Algorithm (Optimization Engine)

Step 1: Initial Assignment

- Distribute each faculty across all 8 classes evenly
- Assign 5 periods per faculty-class combination across the week
- Use round-robin or balanced distribution to avoid clustering

Step 2: Conflict Detection

- Check for overlaps (faculty teaching two classes at same time)
- Verify no class has multiple teachers simultaneously
- Ensure room availability

Step 3: Optimization

- Minimize back-to-back classes in different buildings
- Balance daily workload (avoid 8 classes in one day, 0 in another)

- Respect faculty break times and preferences
- Group similar subjects for continuity

Step 4: Iterative Refinement

- Use constraint satisfaction algorithms (backtracking, genetic algorithms, or simulated annealing)
- Swap slots to resolve conflicts
- Re-run until all constraints are satisfied or report unsolvable conflicts

4. Smart Scheduling Features

Load Balancing:

- Distribute faculty workload evenly across weekdays
- Avoid assigning faculty more than X consecutive periods
- Ensure each day has similar number of faculty visits per class

Pattern Recognition:

- Learn from previous successful schedules
- Identify preferred teaching patterns (morning vs afternoon)
- Avoid historically problematic slot combinations

Conflict Resolution:

- Prioritize hard constraints (no double-booking) over soft ones (preferences)
- Suggest manual adjustments if full automation fails
- Highlight conflicting faculty/rooms for admin review

5. Output Generation

Multi-View Timetables:

- **Faculty-wise**: Each teacher sees their weekly schedule (which class, when, where)
- **Class-wise**: Each class sees which faculty visits them and when
- **Room-wise**: Room occupancy schedule
- **Master Grid**: Admin view showing entire campus schedule

Export Options:

- Downloadable PDFs, Excel sheets
- Calendar integration (sync to Google Calendar, Outlook)
- Mobile app display for students and faculty

6. Admin Dashboard Features

Pre-Generation Controls:

- Set priority rules, blackout times, special requests
- Lock certain slots (e.g., "Dr. Smith always teaches Class A on Monday morning")
- Import previous semester schedules as templates

Post-Generation Review:

- Visual conflict highlights (red flags for issues)
- Statistics (average gaps per faculty, utilization rates)
- Manual edit mode (drag-and-drop to swap slots)
- Re-generate specific days or classes without affecting entire schedule

Approval Workflow:

- Preview before publishing
- Send draft to faculty for feedback
- Make adjustments and finalize
- Publish to all users simultaneously

7. Ongoing Management

Dynamic Adjustments:

- Handle faculty leave (auto-reassign or mark as free period)
- Substitute teacher assignments
- Mid-semester changes without rebuilding entire schedule
- Emergency rescheduling for events/holidays

Version Control:

- Keep history of all schedule changes
- Rollback to previous versions if needed
- Track who made what changes and when

8. Integration with Campus Assistant

For Students:

- Automatically populate student timetables based on their enrolled class
- See which faculty is teaching next period
- Get notifications if faculty/room changes

For Faculty:

- View personal teaching schedule
- Mark availability/unavailability
- Request schedule swaps with peers

For Admins:

- Central control panel for all scheduling
- Analytics on resource utilization
- Semester planning tools

Example Workflow

1. **Admin Input**: Enter 10 faculty names, 8 class sections, 5 visits per combo
2. **System Calculates**: 400 total assignments needed across 5 days, 8 periods/day = 40 slots/day available
3. **Algorithm Runs**: Assigns faculty to time slots ensuring no conflicts
4. **Preview**: Admin reviews generated schedule in grid view
5. **Adjustments**: Manually swap 2-3 problematic slots
6. **Publish**: Schedule pushed to all faculty and students' apps
7. **Ongoing**: Admin makes mid-week change for sick faculty, system auto-suggests replacement