General Instruction

1. Student Instruction

Basic Rules

1. Each team must consist of 1–4 students.
2. Every team must submit via GitHub Repository Link
   - Working Project (Frontend + Backend)
   - Final Presentation (PPT and Demo)
3. Project must be:
   - Original
   - Developed during the hackathon timeline
4. AI tools (ChatGPT, Copilot, etc.) are allowed for guidance, but:
   - You must understand your code.
   - You must explain your logic during judging.

Timeline Discipline

- No late submissions.
- Final Git commit time will be verified.
- Last-minute ZIP submissions without proper repository history will be rejected.

Project Expectations

Your project must include:

- Working UI
- Proper Backend/API
- Database integration
- Deployment (recommended and Preferable)

2. Git Repository & Team Development Rules

Mandatory Requirements

1. Repository must be created by the team leader within 24 hours of releasing the problem statement.
2. Repository name format: NavKalpana-TeamCode
   Example: NavKalpana-RICR-NK-0001
3. All team members must:
   - Must be the contributor in repo
   - Use individual GitHub accounts
   - Contribute via commits
   - Avoid single-person commits for team projects

Commit Rules

- Minimum 10 meaningful commits.
- Avoid a single bulk commit at the end.
- Commit messages must be clear.
  - Good Commit Example:
    - Added login authentication
    - Integrated payment API
    - Fixed cart validation bug

- Bad Commit Example:
  - Update
  - Final
  - Added
  - done

Bad Commit Messages will lead to negative marking

Branching Rule (Recommended)

- main – Final stable version
- All the Members must Create Separate Branches which further merges to the Main Branch
  - Example:
    - feature/login
    - feature/register
    - bug/update etc.

3. Recommended Project Folder Structure

Repository Name Example: *NavKalpana-RICR-NK-0001*

Standard Structure

- ❖ NavKalpana-TeamCode
  - frontend
  - backend
  - docs
    - problem-statement.pdf
    - architecture-diagram.png
    - api-documentation.md
    - presentation.pptx
  - README.md

4. README.md Must Include

- Project Title
- Team Members & Roles
- Problem Statement
- Tech Stack Used
- Installation Steps
- API Endpoints
- Screenshots
- Future Improvements

5. What Judges Will Verify

Technical Verification

- Git commit history
- Contribution by all members
- Code understanding

- Folder structure clarity
- Backend API functionality
- Database integration
- Error handling

Innovation Check

Judges will evaluate:

- Whether it solves a real problem
- Practical usability
- Scalability potential
- Meaningful use of AI (if included)

Viva Round Questions

Each member may be asked:

- Explain your module.
- Explain database schema.
- Explain API flow.
- Explain deployment steps.
- Explain one technical challenge you faced.

6. Important Warning

Immediate disqualification may occur if:

- Code is copied from the internet without modification
- Purchased templates are used without proper logic implementation
- Repository is created only on the final day
- Only UI is presented without functionality
- Fake data is shown without backend logic (unless clearly defined prototype)
- Only one member answers all questions

7. Bonus Considerations

- Clean UI/UX
- Proper Authentication
- Role-based access control
- AI/ML Integration (if required)
- Deployment (Vercel, Render, AWS, Azure)
- Clean code practices

Final Note

The hackathon is not only about coding. It is about teamwork, structured thinking, problem solving, and clear communication. Build a meaningful solution and be prepared to explain it confidently.

AI Career Intelligence Engine (ACIE)
AI-Powered Preparation-to-Placement Platform

1. Product Overview

The **AI Career Intelligence Engine (ACIE)** is an intelligent web-based system that integrates adaptive learning and interview readiness into one unified AI-driven platform.
The system connects:
Learning → Skill Development → Interview Performance → Career Readiness
It continuously adapts preparation strategies based on measurable performance signals.

2. Product Vision

To create a measurable, adaptive, and intelligent career preparation ecosystem that:

- Identifies academic weaknesses
- Generates personalized study plans
- Simulates real interviews
- Quantifies readiness
- Optimizes preparation dynamically

3. System Architecture

ACIE consists of two major operational layers:
Part 1 – Learning Intelligence Layer (~70%)
Handles academic mastery and preparation.
Part 2 – Interview & Career Intelligence Layer (~30%)
Handles interview simulation, readiness scoring, and placement optimization.

PART 1 – LEARNING INTELLIGENCE ENGINE

4. Authentication & User Management

Features:

- Secure email/password login
- JWT-based authentication
- Password hashing using bcrypt
- Session timeout handling
- Secure data storage

5. Resume Intelligence Module

The resume influences preparation strategy even before interviews.
Input

- PDF Resume
- Plain Text Resume

Processing

- Skill extraction
- Project detection
- Keyword-role mapping

- Experience detection

Resume Strength Score Formula

Resume Strength =
(Skill Relevance × 0.40) +
(Project Depth × 0.30) +
(Experience Indicators × 0.20) +
(Structure Score × 0.10)

Outputs

- Resume Strength Score (0–100)
- Missing skill suggestions
- Resume improvement recommendations

6. Adaptive Study & AI Assessment Execution Module

This module transforms the system into a fully interactive AI-powered academic execution engine.
The system:

- Detects weak areas
- Generates personalized quizzes
- Generates personalized assignments
- Allows students to attempt quizzes
- Allows students to submit assignments
- Evaluates submissions automatically
- Updates mastery and readiness dynamically

6.1 AI-Based Dynamic Quiz Generation

Objective

Automatically generate topic-specific quizzes based on:

- Weak topics
- Resume skill gaps
- Interview performance gaps
- Difficulty progression
- Past mistake patterns

Quiz Generation Logic

Inputs:

- Topic Mastery Score
- Mistake History
- Target Role
- Difficulty Level

Question Selection Formula:
Topic × Difficulty × Error Pattern × Concept Coverage

Quiz Types Supported

- MCQ (Single correct)
- MCQ (Multiple correct)
- Short answer
- Scenario-based technical questions

- Code-output prediction questions

Student Experience

Student clicks:

“Start Adaptive Quiz”

System displays:

- Quiz title
- Topic focus
- Difficulty level
- Time limit
- Total questions

During Quiz:

- Timer enabled
- Question navigation
- Flag for review option

After Submission:

System displays:

- Score (0–100)
- Topic-wise accuracy
- Correct answers
- Detailed explanation
- Mistake classification

Immediate Mastery Update

After quiz completion:

- Topic Mastery recalculated
- Weakness adjusted
- Study plan updated
- Risk score updated

6.2 AI-Based Dynamic Assignment Generation

Objective

Generate practical, skill-based assignments automatically.

Assignment Generation Inputs

- Weak topics
- Interview performance gaps
- Target job role
- Practical skill demand

Assignment Types

- Coding implementation task
- Mini project
- Case study analysis
- Analytical question
- Debugging task

- System design scenario

Example

Topic: React Hooks
Assignment:
“Build a dynamic form with validation using React Hooks.”
Topic: SQL Joins
Assignment:
“Design a relational schema and write optimized join queries.”

Student Submission Options

Students can submit assignments via:

- Embedded code editor
- File upload (ZIP / PDF)
- Text-based answer
- GitHub repository link

6.3 AI Assignment Evaluation Engine

After submission, the system evaluates based on:

- Logical correctness
- Concept application
- Code structure
- Completeness
- Efficiency (if applicable)

Outputs

- Assignment Score (0–100)
- Concept coverage report
- Mistake breakdown
- Improvement suggestions

Mastery Update After Assignment

Topic Mastery Formula:
(Quiz Score × 0.50) +
(Assignment Score × 0.30) +
(Consistency × 0.20)
Mastery is recalculated in real time.

6.4 Adaptive Learning Loop

After each quiz or assignment:
If mastery improves:

- Reduce risk level
- Move topic to moderate

If mastery remains weak:

- Increase difficulty gradually
- Assign targeted practice

- Schedule additional revision

Learning Cycle:
Learn → Attempt → Submit → Evaluate → Update Mastery → Adjust Plan → Repeat

6.5 Student Dashboard Enhancements

Dashboard Displays:

- Upcoming Adaptive Quiz
- Pending Assignments
- High Risk Topics
- Recommended Mini Projects
- Topic Mastery Heatmap
- Daily Study Plan
- Performance Trend

6.6 Adaptive Difficulty Scaling (Optional Advanced)

If last 3 attempts average > 75:

- Increase difficulty level

If last 3 attempts average < 50:

- Reduce difficulty level