



## Silver Spectrum TechFest 2025

A National Level 48-Hour Hackathon - 28, 29, and 30 October 2025

Powered By,



### Problem Statements to be given to registered participants

Domain	PS Code	Problem Statement
Gen AI for Business Applications	NH-SJC-PS-001	Autonomous supply-chain optimiser (agentic RAG)
	NH-SJC-PS-002	Agentic compliance assistant for finance
	NH-SJC-PS-003	Cross-functional marketing & finance campaign planner
Cyber Security	NH-SJC-PS-004	Phishing email detector, Threat intelligence summarizer, Log analysis + incident triage, Security policy assistant
	NH-SJC-PS-005	Spot the Insider Threat
	NH-SJC-PS-006	AI-Powered Cyber Threat Detection Dashboard
Artificial Intelligence / Machine Learning / Deep Learning	NH-SJC-PS-007	Clinical triage assistant, Medical literature summarizer, Medication adherence coach, Hospital resource scheduler, Radiology report helper
	NH-SJC-PS-008	AI teaching assistant that uses the Socratic method to teach sorting algorithms
	NH-SJC-PS-009	Banking Fraud Detection using Multimodal Transactions
Internet of Things (IoT)	NH-SJC-PS-010	AI-Powered IoT Coaching System for Sports Training
	NH-SJC-PS-011	Smart Home Intrusion Detection using Multimodal Sensors
	NH-SJC-PS-012	Defect Detection in Manufacturing using Multimodal Sensors
Innovation using Web 3.0	NH-SJC-PS-013	Front-Running Attack Detection & Mitigation System

## Domain-1

# Gen AI for Business Applications

### NH-SJC-PS-001. Autonomous supply-chain optimiser (agentic RAG)

Retailers manage huge volumes of purchasing, inventory, logistics and demand-forecasting data. Inspired by AIMultiple's description of **agentic RAG**, where an AI agent autonomously routes queries to appropriate databases and self-corrects, teams should build a multi-agent system that accepts natural-language questions from planners (e.g., "Which warehouse should ship March orders to our eastern region to minimise cost and meet service-level agreements?"). The system must:

- Use a routing agent to decide which data sources to query (inventory DB, ERP orders table, shipping-cost API, etc.), generate appropriate SQL/API calls and retrieve the data. The agent should perform multiple function calls and refine its approach if initial results are insufficient.
- Coordinate with a reasoning agent to compute optimal decisions—such as order–warehouse assignments or route schedules—and provide interpretable explanations.
- Present a concise report showing the data sources used, the reasoning steps and the final recommendation.

This task demonstrates how **agentic RAG** can enable autonomous, multi-turn information retrieval and decision-making in supply-chain management.

### NH-SJC-PS-002. Agentic compliance assistant for finance

Financial institutions face complex regulations. A standard chatbot may not access internal policies, whereas **RAG** allows a model to pull in enterprise-specific data and produce accurate outputs. Participants will build an agentic assistant that:

- Ingests regulatory documents, internal risk policies and recent compliance rulings. When a user asks a question—such as "How do the latest Reserve Bank of India guidelines affect our mortgage-underwriting criteria?"—a reasoning agent autonomously retrieves and synthesises relevant information from multiple sources and delivers a response with citations.
- Highlights ambiguities or conflicting regulations, flags potential risks and suggests follow-up questions.
- Demonstrates agentic features—autonomous decision-making and real-time adaptation—by refining its responses based on user feedback and evolving regulations.

This challenge underscores how **agentic AI** can augment compliance teams by offering context-aware answers and risk insights.

### NH-SJC-PS-003. Cross-functional marketing & finance campaign planner

The CIO report notes that AI agents automate repetitive tasks and adapt to changing conditions across domains like customer service, supply chain and IT operations. Planning a marketing campaign often requires coordination between marketing, finance and inventory teams. Participants will create a multi-agent platform that:

- Uses RAG to access CRM data (customer profiles), finance budgets and real-time inventory levels.
- Plans a product promotion by generating targeted customer segments, projecting revenue, checking budget constraints and suggesting discount levels.
- Deploys multiple agents: one agent proposes creative content, another agent validates financial feasibility, and a third checks inventory availability. Agents collaborate and negotiate, adjusting the plan autonomously.
- Presents the final campaign plan along with supporting data and explanations, adapting when underlying data changes.

## Domain-2

# Cyber Security

### NH-SJC-PS-004. Phishing email detector, Threat intelligence summarizer, Log analysis + incident triage, Security policy assistant

#### Pick a cybersecurity use case

- Phishing email detector + auto-responder: a GenAI agent analyzes incoming emails, flags phishing attempts, drafts safe responses.
- Threat intelligence summarizer: an agent retrieves the latest CVEs/threat feeds and summarizes risks for security teams.
- Log analysis + incident triage: ingest system logs, use RAG to identify anomalies, and recommend mitigation steps.
- Security policy assistant: generate or validate compliance policies automatically.
- User-friendly security coach: explain security alerts or incidents to non-technical users in plain language.

#### Build the GenAI workflow

- Use a GenAI framework such as LangChain, LangGraph, LlamaIndex, AutoGen, etc.
- Show how your system retrieves, analyzes, and generates insights or actions.
- Integrate at least one external data source or tool (threat feeds, logs, APIs, databases).
- Optionally, use RAG (Retrieval-Augmented Generation) or Agentic workflows (multiple cooperating agents for retrieval, reasoning, and action).

#### Host/Run it

- Deploy on AWS or Azure (Bedrock/OpenAI on Azure, Lambda/Functions, DynamoDB/CosmosDB, etc.) or run locally / cloud-agnostic.

#### Prepare a short demo

- 5–7 minute demo showing how the system works
- 1–2 page architecture diagram/slide deck explaining agents, tools, and state flow

#### Technologies & Resources

- GenAI frameworks: LangChain, LangGraph, LlamaIndex, Microsoft AutoGen, Hugging Face Transformers
- Data sources: CVE/NVD feeds, VirusTotal, Shodan, Syslog/Kibana, AWS GuardDuty, Azure Sentinel
- Cloud (optional): AWS (Bedrock, Lambda, DynamoDB, S3), Azure (OpenAI, Functions, CosmosDB, Sentinel), or on-prem APIs. Any open APIs or security datasets

#### Tips

- Use RAG to enrich LLM responses with up-to-date threat intelligence.
- Design agentic workflows where one agent retrieves threat data, another analyzes, another drafts actions/reports.
- Focus on privacy & security in your own implementation (no leaking of credentials).
- Show how your GenAI solution would fit into a real security operations workflow.

## NH-SJC-PS-005. “Spot the Insider Threat”

### **Background:**

In IT companies, thousands of employees access emails, cloud apps, code repositories, and sensitive client data daily. While firewalls and antivirus protect against outsiders, some of the biggest risks come from inside — like an employee accidentally uploading data to public drives, sharing passwords, or using personal USB devices.

### **The Challenge:**

Design a simple solution to detect and alert when there is a potential insider threat.

### **Your solution should:**

- Monitor basic activities such as unusual file uploads, bulk downloads, or logins from unexpected locations.
- Show real-time alerts like “Large data download detected from HR folder at 2 AM.”
- Keep employee privacy protected (avoid storing personal details unnecessarily).
- Provide a clean dashboard with a simple risk score: Low / Medium / High. [a simple dashboard/web page]
- Simulate 3–4 basic anomalies: like login at odd hours, large file download, or multiple failed login attempts.
- Build a basic rule-based detector (no need for advanced ML).

### **Bonus:**

- Add a feature to suggest quick actions: “Suspend suspicious session” or “Notify IT Security.”
- Show how the tool scales for a company with 1,000+ employees.

### **Goal:**

Help IT companies catch risky activities early and prevent data leaks — keeping both client data and company reputation safe.

## **NH-SJC-PS-006. AI-Powered Cyber Threat Detection Dashboard**

**Cybersecurity – Focused on proactive threat detection, visualization, and simulation using AI.**

Modalities:

- Text: Threat logs, system alerts, and incident reports.
- Image: Screenshots of attack patterns, phishing attempts, malware signatures.
- Video: Security camera feeds or recorded attack simulations for anomaly detection.

Detailed Steps to Implement:

1. Data Collection & Preprocessing:

- Aggregate threat logs from firewalls, IDS/IPS systems, and SIEM tools.
- Normalize and clean data for consistency across sources.

2. Threat Pattern Analysis:

- Use NLP models to extract insights from textual logs (e.g., suspicious IPs, attack vectors).
- Apply computer vision techniques to detect anomalies in images or video feeds.

3. Generative Modeling for Synthetic Scenarios:

- Implement GANs or diffusion models to create synthetic attack scenarios for training and stress-testing.
- Simulate phishing campaigns, DDoS attacks, and ransomware behaviors in a controlled environment.

4. Dashboard Development:

- Build an interactive dashboard using frameworks like Streamlit, Dash, or React.
- Include real-time threat visualization (heatmaps, timelines, attack graphs).
- Provide drill-down capabilities for detailed analysis of each threat.

5. Alerting & Recommendations:

- Integrate AI-driven alerts for high-risk patterns.
- Suggest mitigation strategies based on historical data and predictive analytics.

Expected Output:

A fully interactive dashboard that:

- Displays real-time threat visualizations (maps, graphs, timelines).
- Offers synthetic attack simulations for training and preparedness.
- Provides actionable insights and recommendations for security teams.

## Domain-3

# Artificial Intelligence / Machine Learning / Deep Learning

### NH-SJC-PS-007. Medical Assistant

#### Pick a medical use case

- Examples:
- Clinical triage assistant: intake patient symptoms, retrieve guidelines, generate preliminary triage notes.
- Medical literature summarizer: fetch and summarize latest research for doctors.
- Medication adherence coach: personalized reminders + explanations for patients.
- Hospital resource scheduler: agentic system that retrieves resource data, forecasts demand, and schedules staff.
- Radiology report helper: retrieve historical cases, help draft preliminary reports.

#### Build the GenAI workflow

- Use LangChain or LangGraph to define agents, tools, memory/state, and workflows.
- Integrate RAG or Agentic RAG to pull from medical guidelines, research papers, or hospital data.
- Show state propagation between nodes/agents.
- Include at least one external data source or service (e.g., PubMed API, FHIR server, hospital database).

#### Host/Run it

- Deploy on AWS or Azure (Bedrock/OpenAI on Azure, Lambda/Functions, DynamoDB/CosmosDB, HealthLake, Azure Health Data Services) or run locally/cloud-agnostic.

#### Prepare a short demo

- 5–7 minute demo showing how the system works
- 1–2 page architecture diagram/slide deck explaining agents, tools, RAG components, and state flow

#### Technologies & Resources

- GenAI frameworks: LangChain, LangGraph, LlamaIndex, Microsoft AutoGen, Hugging Face Transformers
- Medical data sources: PubMed, WHO guidelines, FHIR servers, open-source EHR data sets (e.g. MIMIC-III), hospital APIs (synthetic/test only)
- Cloud (optional): AWS (Bedrock, HealthLake, Lambda, DynamoDB, S3), Azure (OpenAI, Health Data Services, Functions, CosmosDB)

#### Tips

- Use RAG to enrich LLM responses with up-to-date medical knowledge.
- Agentic workflows: design one agent for retrieval (guidelines, literature), another for reasoning/answer generation, and others for acting or orchestrating tasks.
- Focus on privacy & security of patient data — only use de-identified or synthetic data for the hackathon.
- Show how your workflow could realistically fit into a clinical or hospital workflow.

## **NH-SJC-PS-008. Create an AI teaching assistant that uses the Socratic Method to teach sorting algorithms.**

**The assistant should:**

1. Ask probing questions instead of revealing answers directly
2. Adapt follow-up questions based on student responses
3. Guide students to discover answers through their own reasoning
4. Maintain focus on sorting algorithms
5. Handle the computational complexity efficiently

**Target Topic:** Data Structures and Algorithms - Sorting Algorithms

This topic is chosen because it is fundamental to computer science education, familiar to most software engineering students, and well-defined in scope, making it ideal for a focused prototype.

### **Example 1: Time Complexity Query**

Student Question: "What is the time complexity of bubble sort?"

Traditional Approach (NOT what we want): "The time complexity of bubble sort is  $O(n^2)$ ."

Socratic Approach (What your solution should do):

Step 1: "Let's think about how each algorithm works. Can you describe how merge sort divides the problem?"

Step 2: "Good! Now, how does that division strategy affect the number of operations needed?"

Step 3: "Now compare that with bubble sort - how many comparisons does it make for an array of size  $n$ ?"

Step 4: Guide the student to compare  $O(n \log n)$  vs  $O(n^2)$  on their own

### **Example 2: When/Why to Use an Algorithm**

Student Question: "When should I use quick sort?"

Traditional Approach (NOT what we want): "Quick sort is good for large datasets and has an average time complexity of  $O(n \log n)$ , but worst case is  $O(n^2)$  when the pivot is poorly chosen."

Socratic Approach (What your solution should do):

Step 1: "What factors do you think are important when choosing a sorting algorithm for a project?"

Step 2: "Interesting! How do you think quick sort performs on average compared to its worst case?"

Step 3: "What might cause quick sort to hit its worst-case scenario?"

Step 4: "Given what you've discovered, in what situations would quick sort be a good choice?"

### **Example 3: Implementation Details**

Student Question: "How does merge sort work?"

Traditional Approach (NOT what we want): "Merge sort divides the array into halves recursively until single elements remain, then merges them back in sorted order."

Socratic Approach (What your solution should do):

Step 1: "Have you heard of the divide-and-conquer strategy in algorithms?"

Step 2: "How do you think we could apply that strategy to sorting?"

Step 3: "If we keep dividing an array in half, what's the smallest piece we can get?"

Step 4: "Great! Now, if you have two sorted arrays, how would you combine them into one sorted array?"

Step 5: Guide the student to piece together the full merge sort process

#### **Example 4: Handling Incorrect Answers**

Student Question: "Is bubble sort O(n) time complexity?"

Traditional Approach (NOT what we want): "No, that's incorrect. Bubble sort is O( $n^2$ )."

Socratic Approach (What your solution should do):

Step 1: "Let's think about this together. In bubble sort, how many times do we need to pass through the array?"

Step 2: "And in each pass, how many comparisons might we make?"

Step 3: "So if we have n elements, and we make up to n passes, with up to n comparisons each pass, what does that tell us?"

Step 4: "Right! So would you like to reconsider your answer about the time complexity?"

#### **FINAL NOTE**

Remember: The goal is not just to build an AI that knows the answers, but one that teaches students how to find them. Your assistant should make students think, not just provide information.

## **NH-SJC-PS-009. Banking Fraud Detection using Multimodal Transactions**

Banking & Financial Services – Focused on fraud prevention using AI and multimodal data analysis.

#### **Modalities:**

Text: Transaction logs, account activity, and customer details.

Audio: Voice recordings from customer service calls or verification calls.

Image: ID documents, scanned checks, and profile pictures for identity verification.

#### **Detailed Steps to Implement:**

##### **1. Data Collection & Integration:**

- Gather transaction logs from banking systems (amount, location, device info, time).
- Collect voice call recordings from customer support or verification processes.
- Acquire ID images (government-issued IDs, selfies for KYC) for authentication checks.

##### **2. Preprocessing & Feature Extraction:**

- Text: Use NLP to detect anomalies in transaction patterns (e.g., sudden large transfers, unusual geolocation).
- Audio: Apply speech-to-text and voice biometrics to verify identity and detect stress or suspicious behavior.
- Image: Use computer vision for ID validation (e.g., forgery detection, face matching with selfie).

### **3. Multimodal Fusion & AI Model:**

- Combine features from text, audio, and image into a unified representation using multimodal deep learning.
- Train fraud detection models using supervised learning and anomaly detection techniques.

### **4. Generative AI for Synthetic Fraud Scenarios:**

- Use GANs or diffusion models to create synthetic fraud cases for model training and stress testing.
- Simulate scenarios like fake IDs, voice spoofing, and transaction laundering patterns.

### **5. Fraud Detection Dashboard:**

- Build an interactive dashboard for fraud analysts:
- Real-time alerts for suspicious transactions.
- Multimodal evidence (transaction details, voice snippet, ID image).
- Synthetic fraud scenarios for proactive risk assessment.

### **6. Reporting & Compliance:**

- Generate automated fraud reports with confidence scores and recommended actions.
- Ensure compliance with banking regulations (KYC, AML, GDPR).

### **Expected Outcomes:**

A fraud detection dashboard that:

- Displays real-time alerts with multimodal evidence.
- Provides synthetic fraud scenarios for training and preparedness.
- Generates detailed reports for compliance and investigation.

## **Domain-4**

# **Internet of Things (IoT)**

### **NH-SJC-PS-010. Problem Statement: AI-Powered IoT Coaching System for Sports Training**

#### **Background**

Quality coaching is the difference between good and great athletes. However, personal coaches are expensive, not always available 24/7, and inaccessible to millions of aspiring athletes. What if technology could provide real-time, intelligent coaching that adapts to each athlete's needs, corrects their form, designs personalized training plans, and motivates them like a human coach would?

#### **The Challenge**

Create an intelligent IoT coaching system that acts as a virtual sports coach - not just tracking metrics, but actively training athletes through real-time guidance, technique correction, personalized workout planning, and adaptive feedback based on performance and progress.

#### **The Core Problem**

Athletes Need a Coach, Not Just Data

- Passive devices tell you what happened, but don't help you improve
- Human coaches are expensive and not always available
- Generic training apps don't adapt to individual weaknesses or provide real-time correction
- Athletes train alone without immediate feedback on form, pacing, or technique

#### **What a Real Coach Does (Your Device Should Too)**

1. Observes and corrects technique in real-time
2. Designs personalized training plans based on goals and current level
3. Provides motivation and encouragement during tough workouts
4. Adapts training based on fatigue, performance, and progress
5. Teaches fundamentals and advanced techniques progressively
6. Prevents injuries by catching poor form or overtraining
7. Gives tactical advice during practice sessions

## **NH-SJC-PS-011. Smart Home Intrusion Detection using Multimodal Sensors**

IoT (Internet of Things) – Leveraging connected smart devices for home security.

### **Modalities:**

Audio: Sounds captured by smart microphones (e.g., breaking glass, forced entry noises).

Image: Video frames or snapshots from smart cameras for motion and object detection.

Text: Metadata from sensors (timestamps, device logs, door/window status).

### **Detailed Steps to Implement:**

#### **1. Sensor Integration & Data Acquisition:**

- Connect smart cameras, microphones, and door/window sensors to a central IoT hub.
- Stream audio and video data in real-time; collect sensor logs for contextual information.

#### **2. Preprocessing & Feature Extraction:**

- Apply noise reduction and audio feature extraction (MFCCs) for sound classification.
- Use image processing and deep learning (CNNs) for detecting human presence or unusual activity.
- Combine sensor logs for event correlation (e.g., door opened + motion detected).

#### **3. Multimodal Fusion & AI Model:**

- Implement a fusion model that combines audio, image, and text features for robust intrusion detection.
- Use anomaly detection algorithms or supervised learning for classification of intrusion vs. normal activity.

#### **4. Alerting & Reporting System:**

- Trigger real-time alerts via mobile app or SMS when intrusion is detected.
- Generate incident reports with timestamp, detected modality, and confidence score.
- Attach visual evidence (snapshot or short video clip) for verification.

#### **5. Dashboard & User Interface:**

- Build a dashboard to monitor live feeds, intrusion alerts, and historical reports.
- Include analytics like intrusion frequency, time-of-day patterns, and device health status.

### **Expected Outcomes:**

- Real-time intrusion alerts delivered to the homeowner's device.
- Incident reports with detailed logs and visual/audio evidence.
- A user-friendly dashboard for monitoring and managing home security.

## **NH-SJC-PS-012. Defect Detection in Manufacturing using Multimodal Sensors**

Manufacturing – Leveraging AI and IoT for predictive maintenance and quality assurance.

### **Modalities:**

- Image: Visual inspection of machine parts and products using cameras.
- Audio: Sound patterns from machines (e.g., grinding, vibration noise) for anomaly detection.
- Text: Machine logs, operational parameters, and maintenance history.

### **Detailed Steps to Implement:**

#### 1. Data Acquisition & Integration:

- Collect image data from cameras installed on production lines for visual defect detection.
- Capture audio signals from microphones placed near machines to monitor abnormal sounds.
- Gather textual logs from sensors and PLCs (Programmable Logic Controllers) for operational context.

#### 2. Preprocessing & Feature Extraction:

- Image: Apply computer vision techniques (CNNs) to detect cracks, misalignments, or surface defects.
- Audio: Use signal processing (MFCCs, spectrograms) to identify unusual frequency patterns indicating wear or malfunction.
- Text: Parse logs for error codes, temperature spikes, or vibration anomalies using NLP.

#### 3. Multimodal Fusion & AI Model:

- Combine image, audio, and text features into a unified representation using multimodal deep learning architectures.
- Train classification models to categorize defects (e.g., mechanical, electrical, surface).

#### 4. Defect Classification & Alerting:

- Implement real-time detection and classification of defects.
- Trigger alerts for critical anomalies and schedule predictive maintenance tasks.

#### 5. Report Generation & Dashboard:

- Generate automated maintenance reports with defect type, severity, and recommended actions.
- Include visual evidence (images) and audio clips for verification.
- Build a dashboard for supervisors to monitor machine health and defect trends.

### **Expected Outcomes:**

#### 1. A defect classification system that:

- Detects and classifies defects using multimodal data.
- Provides visual and audio evidence for each detected issue.
- Generates maintenance reports with actionable insights.

#### 2. Additional features:

- Predictive analytics for future failures.
- Historical defect trend visualization.

## Domain-5

### Innovation using Web 3.0

#### NH-SJC-PS-013. Front-Running Attack Detection & Mitigation System

##### Problem:

MEV (Maximal Extractable Value) bots constantly front-run regular users by inserting transactions with higher gas fees, exploiting DeFi traders and NFT buyers. This leads to slippage, unfair trades, and financial losses for everyday users.

Build a middleware DApp or dashboard that:

1. Detects suspicious mempool behavior, including sandwich attacks, front-running, and other MEV patterns.
2. Shows flagged transactions, affected users, and potential slippage losses in real-time.
3. Optionally integrates a private transaction relayer (e.g., Flashbots) to allow users to submit transactions safely.
4. Bonus: Includes a simulation mode where users can test transactions before submitting to predict front-running risks.

##### Evaluation Criteria:

Metric	Description	Evaluation Criteria
Detection Accuracy	How well the system identifies front-running / sandwich attacks in the mempool	% of correctly flagged malicious transactions vs. total simulated attacks
User Impact Awareness	Clarity of information about affected users and potential losses	Quality of dashboard visualization, ease of understanding, actionable insights
Real-Time Performance	Speed at which transactions are analyzed and flagged	Time from transaction broadcast → detection (< few seconds ideal)