Problem Statement 1:

Let AI Speak to Your Money

In a world where personal financial data is scattered across multiple accounts, platforms, and services, it's difficult for individuals to get a complete, real-time view of their financial health. AI assistants today can answer general queries, but without structured access to personal financial data, they can't deliver truly personalized, actionable insights.

Challenge:

Within 36 hours, create a prototype of an AI-powered personal finance assistant that can:

1. **Pull and process structured mock financial** data (assets, liabilities, transactions, EPF balance, credit score, investments). This can be simulated through JSON files or a mock API that mimics the Financial MCP (Multi-Channel Processing) Server.
2. **Allow natural-language conversations** where users can ask financial questions such as:
    - "Can I afford to take a vacation next month?"
    - "Why did my expenses increase last quarter?"
    - "What's my best option for repaying my loan faster?"
3. **Generate actionable insights** using an AI model (e.g., Gemini, GPT-4, Claude) — such as forecasting savings, detecting unusual spending patterns, and suggesting simple investment strategies based on the mock data.
4. **Demonstrate user control and privacy** by allowing the user to "grant" or "revoke" access to certain categories of data, even if it's simulated.

Requirements:

1. **Data Management**
    - Retrieve structured financial data from JSON files or a mock API that acts like a Financial MCP Server.
    - Parse, validate, and organize data into categories including: assets, liabilities, transactions, EPF/retirement balances, credit score, and investment portfolios.
      ✓ **Assets** → cash, bank balances, property, etc.
      ✓ **Liabilities** → loans, credit card debt, etc.
      ✓ **Transactions** → income, expenses, transfers.

✓ **EPF/Retirement Balance** → contributions, employer match, current balance.
✓ **Credit Score** → mock numeric value with rating (e.g., 750 = "Good").
✓ **Investments** → stocks, mutual funds, bonds, etc.

- Store processed data in a structured format for later analysis and insights.
- Users can control which financial categories the assistant can access. For example, a user may choose to grant access to **assets** and **transactions**, while revoking access to **credit score** and **EPF balance**. The system will then process insights only from the permitted categories. At any point, the user can update these permissions to reflect their preferences.

2. **Natural-Language Interaction**
   - Accept natural-language queries from the user (e.g., *"How much did I spend last month?"*).
   - Respond with direct answers, contextual explanations, and suggested actions (like creating a budget).
   - Maintain context for follow-up queries within a conversation session. So if you follow up with *"What about the last 3 months?"* it knows you're still talking about spending.

3. **AI-Powered Insights**
   - Provide actionable insights such as predict your future savings, detecting unusual spending, and even suggest smarter ways to repay debt or invest your money.
   - Ensure all insights are derived exclusively from data categories to which the user has granted access.
   - Present insights in clear, user-friendly language suitable for non-technical users.

4. **Privacy and User Control**
   - Simulate granting/revoking permissions for each data category. Let users decide what data the assistant can see — for example, they might share their **transactions** but keep their **credit score** private.
   - Ensure queries only use data the user has authorized.

Problem Statement 2:

Smart Bus Optimization Challenge

Urban bus systems in Indian Tier-1 cities (e.g., Bangalore, Delhi, Pune) rely on static timetables that fail to adapt to real-world conditions. This leads to bus bunching, under-utilized trips during off-peak hours, and unpredictable wait times. Transit agencies lack tools to forecast demand surges and adjust schedules in real time.

Challenge

Build a Smart Bus Management System prototype in 36 hours that makes city buses run smarter, on time, and with better passenger experience.

What Your System Should Do

- **Use Past Data**
  - Work with at least two types of data (ticket sales, passenger counts, GPS logs).
  - Clean the data: fix missing values, format timestamps, and remove wrong/outlier data.

- **Simulate Live Bus Updates**
  - Pretend buses are moving in real time (using a loop or stream).
  - Show how new data (like GPS location or bus occupancy) updates the schedule.

- **Fix Bus Problems**
  - Stop bus bunching (buses coming together at once).
  - Avoid empty trips by adjusting frequency in off-peak times.

- **Predict Passenger Demand**
  - Use a simple ML model or time-series method to guess how many passengers will ride at a given time.
  - Give short-term demand predictions for each route/hour.

- **Show Before vs. After**
  - Compare current schedule vs. optimized schedule.
  - Show improvements (like reduced waiting time or better bus usage).
  - Display results with simple charts or maps.

Requirements

• Use at least 2 data sources (e.g., ticket sales + GPS).

• Simulate real-time data feed (buses moving + passengers boarding).

• Build a scheduling engine (can be rule-based or ML-based) that updates bus timings automatically.

• Create a prediction model that forecasts ridership for the next few hours.

• Make a dashboard/UI that shows:

- Original vs. Optimized schedules
- Forecasted vs. Actual ridership
- Alerts (e.g., “Route 5 delayed – rescheduling now…”)

• Deploy as a working prototype (web, mobile, or CLI tool) with easy setup.

• **Bonus**: Show live buses on a map view with updated schedules.

Problem Statement 3:

Multi-Channel Digital Arrest & Fraud Scam Detection and Prevention

Digital arrest scams are a rapidly growing cybercrime trend where fraudsters impersonate law enforcement, government authorities, or financial institutions to coerce victims into making immediate payments or disclosing sensitive personal information. Scammers use **psychological manipulation**, **spoofed caller IDs**, **AI-generated voices**, **deepfake videos**, and **fake documents** to create a sense of urgency and fear. The victims are threatened with "digital arrest," imprisonment, or legal consequences unless they comply.

These scams are increasingly sophisticated and difficult for the average user to detect. Fraudsters exploit multiple communication channels: SMS, email, voice calls, video calls, and social media—making detection even more challenging.

Problem

Existing prevention methods are **reactive**, relying mainly on public awareness campaigns, media reporting, or user vigilance. These measures are insufficient against evolving tactics such as:

- **Financial loss**: Victims transfer large sums, sometimes their life savings.
- **Erosion of trust**: Legitimate government and banking communications lose credibility.
- **Jurisdictional complexity**: Transnational nature of scams makes prosecution difficult.
- **AI-driven deception**: Deepfake audio/video makes scams more believable, leaving victims with little chance of distinguishing real vs. fake interactions.

There is a **critical need for a proactive, intelligent, and multi-channel system** that can detect and flag potential digital arrest scams, including **text, audio, and video fraud attempts**. In real-time, empowering users with immediate alerts and protective measures.

Challenge

Build a **proof-of-concept Multi-Channel Digital Fraud Detection and Prevention System** that can:

1. Ingest and Analyze Communication Data

- **Text-based Data**: Simulated SMS, email, chat transcripts.
- **Voice-based Data**: Caller ID, call metadata (duration, frequency), and speech-to-text transcripts for keyword spotting.
- **Video-based Data**: Detect possible deepfake or manipulated content in video calls (e.g., AI-generated law enforcement officials).

2. Detect Scam Patterns using AI/ML

- Apply **NLP techniques** for text analysis and classification (legitimate vs. scam).
- Use **keyword spotting & sentiment analysis** to detect urgency, fear, and threats.

- Apply **deepfake/voice spoofing detection models** to flag synthetic audio or video.
- Perform **sender/caller reputation analysis** (e.g., suspicious email domains, spoofed phone numbers).
- Build classification models that adapt and improve with user-reported suspicious communications.

3. Real-Time Prevention & Alerts

- Trigger **immediate, clear, and actionable alerts** when potential scams are detected.
- Provide **protective advice** (e.g., "Do not share OTP," "Verify caller ID via official website," "This video may be AI-generated, verify source").
- Enable **call blocking, email filtering, or warning overlays** in real-time.

4. User-Friendly Interface

- A **dashboard or mobile/web app** that displays:
  - Detected scam attempts with explanations.
  - Highlighted **keywords/phrases or deepfake indicators** that triggered detection.
  - Educational resources on common scam tactics.
- A **manual reporting feature** for users to submit suspicious communications and help improve the detection model.

Requirements

- **Data Simulation/Collection**: Mock datasets for text, audio, and video communications (including both legitimate and fraudulent examples).
- **AI/ML Models**:
  - NLP models for scam text detection.
  - Speech-to-text + audio deepfake detection models.
  - Basic deepfake video detection techniques (e.g., blink rate, lip-sync mismatches, or pretrained classifiers).
- **Alerting Mechanism**: Instant user notifications with context and recommendations.
- **UI/UX**: A simple interface (web app, CLI tool, or mobile simulation) for displaying alerts and prevention tips.
- **Explainability**: Highlight risky patterns (phrases, audio markers, video inconsistencies) for user trust.
- **Deployable Prototype**: A single runnable solution with setup instructions.

Problem Statement 4:

Artificial Intelligence for Human Management in the Indian Air Force (IAF)

The Indian Air Force (IAF) operates one of the most diverse and skilled workforces in the country, consisting of pilots, engineers, technical specialists, ground staff, and administrative personnel. Managing this workforce is a mission-critical task that directly impacts national security, operational readiness, and overall efficiency.

Traditional human resource management methods within large defense organizations face challenges such as fragmented data systems, manual workforce allocation, and reactive decision-making. The complexity multiplies when factoring in training requirements, medical fitness, leadership potential, and deployment needs across dynamic operational environments.

Problem

The current system faces multiple bottlenecks:

- **Large-scale data handling**: Personnel records, training logs, medical history, performance reports, and mission-readiness data are stored in disparate systems, leading to inefficiencies in analysis and decision-making.
- **Workforce allocation inefficiencies**: Manual posting and deployment may lead to skill mismatches, underutilization of talent, or delays in filling critical roles.
- **Predictive challenges**: Identifying future training requirements, career progression opportunities, or attrition risks is difficult without advanced analytics.
- **Operational readiness issues**: Rapid deployment of the right personnel with the right skills at the right time is crucial, but hard to achieve with conventional systems.

This creates an urgent need for a secure, AI-driven human management system tailored for the IAF that can centralize personnel data, optimize workforce allocation, and support predictive decision-making at scale.

Challenge

Design and prototype an **AI-enabled Human Management System for the IAF** that can:

1. ***Centralize Personnel Data***
   - Integrate **structured and unstructured data** from multiple sources (skills, training history, medical records, mission readiness, and career progression).
   - Ensure **secure, role-based access** for different stakeholders (commanders, HR managers, medical officers, training departments).
2. ***Optimize Workforce Allocation***
   - Match personnel **skills, experience, and qualifications** to postings, missions, and operational roles.
   - Detect and predict **skill shortages or surpluses** across units.
   - Suggest **reskilling or cross-training pathways** to balance workforce capabilities.

*3. **Enable Predictive Analytics***

- Forecast **training needs** based on operational demands and evolving technology.
- Identify **future leaders** and design optimized career pathways.
- Detect **attrition risks** and recommend proactive retention strategies.
- Predict **mission readiness levels** for individuals and units.

*4**. Support Decision-Making***

- Provide **interactive dashboards** for commanders and HR managers.
- Deliver **explainable AI insights** behind recommendations (why a person is suited for a role, why a unit may be at risk).
- Generate **what-if simulations** (e.g., impact of retiring a group of officers, redeploying technical staff, or grounding pilots due to medical reasons).

Requirements

1. **Data Integration**
    - Must handle large, heterogeneous datasets from different IAF systems.
    - Include both structured data (personnel records, medical scores, mission logs) and unstructured data (training feedback, performance reports, text-based evaluations).
2. **AI Models**
    - **ML/DL algorithms** to analyze integrated datasets.
    - Models for classification (leadership potential), clustering (skill grouping), and prediction (attrition, readiness).
    - Incorporation of **natural language processing (NLP)** for unstructured text analysis.
3. **Security & Privacy**
    - Strict compliance with **military-grade cybersecurity standards**.
    - **Data encryption, multi-factor authentication, access control, and secure transfer protocols**.Privacy-preserving AI techniques (e.g., role-based access, federated learning if feasible).
4. **Dashboard / User Interface**
    - User-friendly, role-specific, and interactive dashboards.
    - Modules to display:
        - Workforce analytics (distribution of skills, experience levels).
        - Predictive career insights and training recommendations.
        - Mission readiness visualization (individual, unit, and organizational level).
5. **Scalability**
    - Capable of scaling to **thousands of personnel records** across all IAF units.
    - Modular architecture for future integration with Air HQ and defense analytics platforms.

Problem Statement 5:

AI/ML Based Prediction of Thunderstorm and Gale Force Wind over an Airfield

Thunderstorms and gale force winds pose significant threats to airfield operations, affecting flight safety, mission readiness, and ground operations. Traditional forecasting methods rely heavily on numerical weather models and manual observations, which may not capture localized, short-duration weather events with sufficient accuracy.

Challenges

- High variability: Thunderstorms and wind gusts are highly localized and evolve rapidly, making prediction difficult.
- Short lead time: Conventional forecasts often provide insufficient warning for airfield operations to take preventive measures.
- Complex data sources: Weather data is multi-dimensional (satellite imagery, radar data, ground sensors, historical records) and difficult to process manually.
- Operational risk: Missed or inaccurate forecasts can lead to flight delays, equipment damage, or safety hazards. There is a critical need for an AI/ML-driven system that can analyze multiple weather data streams in real time and provide accurate short-term predictions of thunderstorms and gale force winds specific to airfields, enabling timely and effective operational decisions.

AI/ML Weather Prediction & Alert System for Airfields

- Develop in 36 hours an AI-powered system to collect, process, predict, and alert severe weather events impacting airfield operations.

Requirements

1. Data Ingestion & Processing

- The system pulls data from multiple sources:Radar (DWR): storm cells, wind velocity. Satellite Imagery: cloud movements, IR/water vapor. Weather Stations (AWS): temperature, humidity, pressure, wind. Historical Records: past events for model training.

- Pipelines continuously gather, clean, and sync data. Missing values are imputed, noisy signals filtered, and all readings aligned on a common timeline. A scalable storage (time-series DB/data lake) maintains real-time + historical data.

2. Weather Prediction Models

- Two forecast horizons:
- Nowcasting (0–3 hrs): for immediate airfield safety.
- LSTMs for wind gusts, CNNs for radar/satellite patterns, ensemble models (Random Forest/GBM) for storm likelihood.
- Medium-term (up to 24 hrs): for planning.
- Longer horizon models incorporating synoptic-scale data. Continuous retraining ensures accuracy improves with new data.

3. Alerts & Explainability

- Alert Engine: Monitors real-time feeds → triggers alerts if thresholds exceeded.
- Confidence Scores: Each forecast includes probability (e.g., "Thunderstorm: 85%").
- Explainability: Shows drivers (e.g., "Pressure drop + radar echoes in north sector").
- Storage: All alerts logged for audits and system refinement.

4. Dashboard Integration

- A user-friendly interface for air traffic control & ground staff.
- Key Features:
- Real-time maps with radar overlays, storm cells, wind arrows.
- Color-coded risk levels: Green = safe, Yellow = caution, Red = danger.
- Custom thresholds per airfield.
- Animated storm movement + impact zones.
- Explainability panel: Why the alert was issued in plain language.
- Alerts: flashing banners, audio, and drill-down details.

Example Scenario

- Thunderstorm Warning (80% in 2 hrs): Triggered by radar + pressure drop → orange alert → delayed landings.
- Severe Winds (65 km/h in 30 min): Red alert → ground crew secures equipment.

Problem Statement 6:

AI-Powered Document Classification and Intelligent Indexing

Organizations and institutions handle large volumes of unstructured data such as documents, reports, contracts, and emails. Traditional methods like manual tagging and keyword-based search are inefficient, error-prone, and slow. There is a pressing need for AI-driven solutions that can classify, index, and retrieve documents automatically, while ensuring security and data privacy.

Challenge:

In 36 Hours, Design and implement an AI-powered Document Classification and Indexing System that can:

1. **Classify Documents:**
   - Automatically categorize documents into predefined classes (e.g., Finance, HR, Technical Reports, Contracts, Invoices).
2. **Extract Metadata & Summaries:**
   - Capture essential details such as title, author, key topics, and entities to build an intelligent index.
3. **Enable Semantic Search:**
   - Provide fast and accurate retrieval of documents based on meaning rather than just keywords.
4. **Ensure Security:**
   - Leverage open-source AI/LLM models (e.g., Hugging Face transformers) with optional integration to private/secure LLM deployments to preserve confidentiality.
5. **Provide User Interface:**
   - Develop a frontend dashboard to upload, classify, and search documents seamlessly.

Backend Requirements

1. Document Upload & Storage

- Accept: **PDF, DOCX, TXT**, Store files **locally**, Save metadata (**filename, date, uploader**) in **SQLite DB**

2. Document Classification

- Categories: **Finance, HR, Legal, Contracts, Technical Reports, etc**

3. Metadata Extraction

- **Title**: First bold line / sentence
- **Author**: Look for "Author:", "By:", emails
- **Date**: Regex-based detection

- **Entities**: Use **spaCy** or regex for names, orgs, amounts

4. Summarization

- **Extractive** approach
- Sentence scoring via **TF-IDF / word frequency**
- Pick **top 3–5** sentences

5. Semantic Search

- Generate embeddings using **SentenceTransformers**
- Store in **FAISS** or in-memory
- Use **cosine similarity** for ranking results

6. Security & Access Control

- **Login system** (hardcoded/simple DB)
- **Role-based access**:
  - HR → HR documents only
  - Finance → Finance documents only
- Store **access logs** (uploads, views)

Frontend Requirements

1. Dashboard

- Upload button
- Document list with: **Title, Category, Author, Upload Date**

2. Document View

- Show:
  - Metadata (Title, Author, Date)
  - Extracted **summary**
  - **Download** original file

3. Search & Filters

- **Search bar** for semantic search
- Filters: **Category, Author, Date**
- Results ranked by **relevance**

4. User Authentication

- **Login page** with username/password
- Show documents based on **user role**

Problem Statement 7:

AI/ML-Enhanced Calibration Platform

Calibration of sensors and instruments is a critical requirement in aerospace, defense, weather monitoring, and industrial IoT. Traditional calibration methods are often manual, static, and lack adaptability, making
them inefficient for modern, data-intensive environments. With advances in AI, ML, and data analytics, calibration platforms can become intelligent, enabling real-time corrections, anomaly detection, and
predictive insights that improve reliability and efficiency.

Challenge:

Design and implement an AI/ML-powered Calibration Platform that can:

1. Provide a GUI dashboard for monitoring, visualization, and comparison of calibration data.
2. Compute offset differences and apply correction algorithms.
3. Integrate AI/ML techniques for anomaly detection, drift prediction, and adaptive calibration.
4. Generate calibration reports (CSV, Excel, PDF) for record-keeping and decision-making.
5. Support scalability for future expansion into predictive calibration and cloud-enabled integration.

Requirements:

- **Computation Engine:**

Compute offset differences between measured and ideal values.
Apply correction algorithms (linear, polynomial, or custom).
Filter out noisy or invalid readings.
Layman Example: Imagine a scale that shows 1 kg too heavy. The platform notices this and automatically fixes the reading.

- **AI/ML Modules**

Anomaly Detection: Find readings that are unusual.
Drift Prediction: Predict future sensor inaccuracies using time-series forecasting.
Adaptive Calibration: Learn from past corrections to improve future adjustments.

Unusual Sensor Readings – Explanation

In software terms, a sensor reading is considered unusual (anomaly) when it does not behave like the majority of past or expected readings. This can happen in several ways:

1) **Out-of-Range Values**
   Each sensor has a normal range of operation. If a reading falls outside this range, it's unusual.
2) **Sudden Jumps (Spikes)**
   If a sensor value changes too quickly, it may be an error.
3) **Stuck Values**
   A sensor may get stuck showing the same value repeatedly, even though conditions should change.
4) **Noisy or Erratic Fluctuations**
   If readings jump randomly up and down without reason, they may be faulty.
5) **Deviation from Predicted Trend**
   AI/ML models can predict the next reading based on past values. If the actual reading is too different from the prediction, it's unusual.

How AI/ML Helps

Statistical methods: Mean, standard deviation → flag values too far away.

Machine Learning models: Trained on normal data → automatically detect unusual patterns.

Time-series forecasting: Predict next reading → check if actual value is reasonable.

- **Dashboard/UI: Show real-time simulated readings.**

Display charts, trends, and alerts.Color-coded alerts: red for critical anomalies, yellow for warnings, green for normal readings.Use Python GUI frameworks: Tkinter, PyQt or web frameworks like Flask, Dash.

- **Reporting: Export readings and AI insights to CSV, Excel, and PDF.**

Reports should include: Corrected readings, Detected anomalies, Drift predictions, Summary charts

- **Alerts & Notifications**

Allow setting thresholds for each sensor. Automatically alert when readings are outside limits.

- **Historical Data Analysis**

Store all simulated readings, corrections, and anomalies in CSV files or SQLite database.
Display historical trends and allow comparisons between calibration cycles.

- **Predictive Insights**

AI predicts which sensors may fail or need recalibration soon. Suggest maintenance schedules based on predicted drift.

- **Scalability:**

Modular design for future extension to real sensors. Data storage in CSV, JSON, or SQLite for portability.

For this hackathon, students may simulate or use open datasets from common sensors such as temperature, pressure, vibration, or humidity, which are widely used in aerospace, defense, weather monitoring, and industrial IoT applications.

Problem Statement 8:

AI/ML-Based Cattle Milk Yield and Health Prediction

Dairy farming is a vital sector in agriculture, providing milk and related products that form a staple of human nutrition. However, one of the biggest challenges faced by farmers is the ***fluctuation in milk yield*** *from cattle. The amount of milk produced by a cow or buffalo is not constant and depends on multiple factors such as the quality of food consumed, daily activity levels (e.g., walking and grazing), environmental conditions, seasonal variations, and most importantly, the health of the animal.*

In many cases, a ***sudden drop in milk yield*** *can act as an early warning indicator of stress, nutritional imbalance, or the onset of diseases such as mastitis, foot-and-mouth disease, or digestive disorders. Traditionally, farmers rely on manual observation and veterinary visits to detect such issues, which often results in delays and economic losses. Modern data-driven methods using* ***Artificial Intelligence (AI) and Machine Learning (ML)*** *provide an opportunity to predict milk output more accurately, detect health problems earlier, and improve overall dairy farm management.*

Challenge:

The challenge is to design and implement an AI/ML-powered Cattle Monitoring Platform that uses data about cattle activity, feed, and health to provide two major capabilities:

1. **Milk Yield Prediction:**
    - Build an ML model to estimate how much milk each cow will produce under given conditions.
    - The model should take into account variables such as feed quality, type of fodder, walking/exercise levels, weather, age, and historical yield data.
    - This prediction can help farmers plan logistics, optimize feed, and maximize output.
2. **Disease Detection and Diagnosis:**
    - Develop a second ML model to analyze cattle data and detect potential diseases or health issues that may cause a drop in milk production.
    - By linking patterns in milk yield reduction with veterinary health data, the model can predict the likelihood of conditions such as mastitis, digestive disorders, or mineral deficiencies.

3. **Integration with Frontend:**
    - Provide an easy-to-use dashboard or mobile app that allows farmers to input data, view predictions, receive disease alerts, and access recommended preventive or corrective measures.

Requirements:

1. **Data Inputs:**
    - **Animal-related data like** Breed, age, weight, lactation stage, parity (number of times calved), Historical milk yield records (daily/weekly), Reproductive cycle information etc..
    - **Feed and nutrition data like** Type of feed (green fodder, dry fodder, concentrates, supplements), Quantity consumed daily, feeding frequency etc.
    - **Activity & behavioral data like** Walking distance, grazing duration, rumination time, Resting/sleeping hours etc..
    - **Health data like** Veterinary records, vaccination history, disease occurrence, Body temperature, heart rate, activity alerts (if IoT sensors are used) etc..
    - **Environmental data like** Ambient temperature, humidity, seasonal conditions, Housing conditions (ventilation, cleanliness) etc..
2. **Model 1 – Milk Yield Prediction:** Regression/forecasting model to predict milk output per cattle per day based on input variables.
3. **Model 2 – Disease Detection:** Classification model to diagnose potential cattle diseases or health conditions responsible for reduced yield.
4. **Frontend/UI:**

    Interactive dashboard (web or mobile) to:
    - ✓ Input data (feed, activity, health observations).
    - ✓ Display predicted milk yield.
    - ✓ Show disease likelihood and recommended preventive/corrective actions.
5. **Reporting:**
    - ✓ The system will generate **comprehensive farm-level Exportable reports** summarizing milk yield trends, feed utilization, animal health records, and disease risk analysis. (CSV/Excel/PDF).
    - ✓ Exported insights will enable better decision-making in areas like feed planning, health management, disease prevention, and overall farm productivity.

Problem Statement 9:

AI/ML-Based Solar Power Generation Prediction

Solar energy has emerged as one of the most promising renewable energy sources, offering a clean and sustainable alternative to fossil fuels. However, the actual power generated by a solar panel depends on multiple dynamic factors such as the geographical location, available sunlight, time of year, panel orientation (tilt and azimuth angles), shading effects, and weather conditions.

Challenges with Current Methods:

- Traditional estimation methods rely on fixed formulas or manual calculations.
- These methods often fail to capture real-world variations.
- This leads to **overestimation or underutilization** of solar systems.

With the advancement of **AI and ML**, it is possible to analyze large amounts of environmental and system data to provide accurate, adaptive, and site-specific solar power predictions.

Challenge

In 36 hours, design and implement an **AI/ML-powered Solar Power Prediction Platform** that can:

1. **Predict Solar Power Output**
    - Build an ML model that predicts the expected electricity generation from solar panels based on environmental and physical parameters.
2. **Incorporate Multiple Factors**
    - Consider variables like sunlight availability, geographic coordinates, panel tilt/angle, surface area, weather conditions (cloud cover, temperature, humidity), and seasonal variations.
3. **Provide Actionable Insights**
    - Suggest the optimal orientation and placement of panels to maximize power generation.
4. **Integrate with Frontend**
    - Deliver a simple, user-friendly dashboard where users can input site details and view predicted solar output along with recommendations.

Requirements

Data Inputs & Sourcing

- Geographic Data: User-provided location (e.g., latitude, longitude, or a city name) and panel physical characteristics (surface area, tilt, and azimuth angles).
- Environmental Data: solar irradiance, temperature, humidity, wind speed, and cloud cover. Data should be available in both historical and forecast formats.

ML Model

- The model must be a regression-based or forecasting model capable of predicting continuous values (kWh or W).
- The chosen model must be able to handle diverse input parameters and identify complex, non-linear relationships between them.
- The platform should support model retraining as new data becomes available to continuously improve prediction accuracy.

Frontend/UI

- Interactive Dashboard: A user-friendly web interface where users can input their location and panel details.
- Visualization: The dashboard should visually display the predicted solar output using charts (e.g., line charts for time-series, bar charts for daily totals).
- Actionable Recommendations: Display the optimal tilt and azimuth angles for the user's location, along with a comparison of the predicted output at the optimal angles versus their current configuration.

Reporting

- Downloadable Reports: The platform should allow users to export the predicted data in various formats like CSV or PDF for further analysis and reporting. The report should include daily, weekly, and monthly predicted generation values, as well as the recommendations for optimal panel placement.

Problem Statement 10:

Smart AI Farming Assistant

Title: Smart AI Farming Assistant – Farming Predictor + Farmer Chatbot

1. Farmers struggle because:
    - Weather is unpredictable (rain may come early/late or not at all).
    - Wrong crop selection leads to crop failure and losses.
    - Many farmers can't read English or use complex apps.
    - They don't know about changing market prices or government schemes.
    - Identifying crop diseases is difficult without expert help.
2. A single smart solution can:
    - Predict weather and recommend the best crops.
    - Talk to farmers in either gujarati or hindi language.
    - Guide them about markets, diseases, and schemes.(LLM/Recommendation)

Challenges

1. Weather prediction is tricky → system must be reliable even with limited data.
2. Farmers speak many languages/dialects → chatbot must handle them.
3. Poor internet in rural areas → app must work offline or with SMS/IVR.
4. Many farmers don't trust "complicated tech" → system must give simple and transparent advice.
5. Solution must run even on low-cost smartphones and basic phones.

Requirements

A) Data Generation & Prediction

1. Collect or generate data for:
    - Soil type and pH (10 types)
    - Rainfall (past + predicted)
    - Temperature (daily high/low/average)
    - Humidity
    - Sunlight hours
    - Wind speed/direction
2. Build AI/ML models to:
    - Predict rainfall (amount, duration, probability) and Temperature.
    - Suggest best crops for all types of soil for predicted weather

B) AI Chatbot for Farmers

The chatbot supports Indian languages like Hindi and Gujarati, with voice and text interfaces so all farmers can use it.

It gives farming guidance and plant growth tips such as the right time to plant, how much to water, best fertilizers, and easy explanations of soil test results,the chatbot will use AI to detect plant diseases and suggest suitable treatments, including natural remedies.

The chatbot explains government schemes in simple words—covering eligibility, benefits, and how to apply—and also connects farmers to a community Q&A forum for shared learning.

C) User Interface (UI)

Must be farmer-friendly UI and SMS interface for basic phones.
Offline Mode: Store essential info when there is no internet.

Problem Statement 11:

AI/ML-Based Fabric Selection and Supply Chain Optimization for the Indian Army

The Indian Army works in very different climates – from the freezing cold in Ladakh to the hot deserts of Rajasthan, the humid coastal areas, and the dense jungles. Soldiers need uniforms that can keep them safe and comfortable in all these places. Right now, uniforms are made using region-specific fabrics .This creates problems:

AI/ML can solve this problem by analyzing data about fabrics, climate, and logistics to recommend the best versatile fabric blend and make supply chains more efficient.

Problem

Design and develop an AI/ML-powered Fabric Selection and Supply Chain Optimization System that can:

1. Predict Fabric Performance
    - Check how fabrics behave in different conditions: insulation in cold, breathability in heat, durability, moisture absorption, etc.
    - Predict how well a fabric will perform in different Army terrains (cold desert, hot desert, humid coast, jungle).
2. Check Sustainability
    - Analyze whether fabrics are eco-friendly (recyclable, reusable, biodegradable).
    - Give each fabric a sustainability score.
3. Recommend Best Fabric
    - Suggest the best single fabric or fabric blend that can work across multiple climates.
    - Reduce the need for region-specific uniforms.
4. Optimize Supply Chain
    - Use AI/ML to forecast demand, availability, and cost.
    - Recommend ways to reduce inventory complexity and improve fabric distribution across Army bases.
5. Provide a Dashboard
    - A simple UI dashboard for Army decision-makers to:
        - Compare fabrics and their suitability in different climates.
        - View sustainability scores.
        - Get supply chain insights.
        - Export reports (CSV, Excel, PDF).

Requirements

- Data Inputs:
  - Climate/environmental data (temperature, humidity, altitude).
  - Fabric properties (weight, breathability, insulation, tensile strength, recyclability).
  - Past performance of Army uniforms.
  - Supply chain & logistics data (cost, stock, transport times).
- AI/ML Models:
  - Fabric Suitability Prediction Model (performance in climates).
  - Sustainability Scoring Model (eco-friendliness).
  - Supply Chain Optimization Model (forecasting demand & inventory).

Challenges

1. Multi-Objective Optimization: Balancing performance, cost, and sustainability in a single recommendation.
2. Dynamic Conditions: Fabrics behave differently in real battlefield conditions (sweat, long wear, movement, and extreme temperature shifts).
3. Sustainability Trade-Off: Eco-friendly fabrics may not be durable enough; balancing both is difficult.
4. Supply Chain Complexity: Army supply chains operate across remote, high-altitude, and border zones with unpredictable disruptions (weather, terrain, emergencies).
5. Real-Time Decision Making: System must provide fast recommendations usable in procurement planning.