



NIC, Odisha Gov.AI Exhibition

16th December, 2024

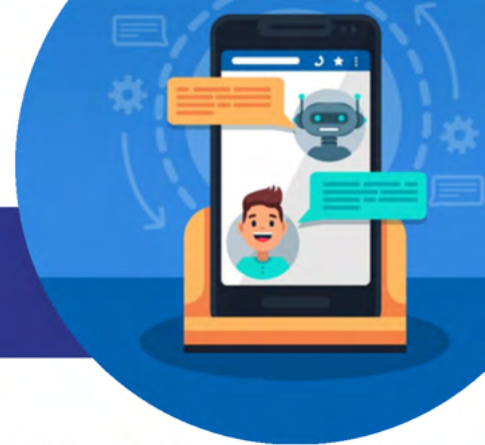
Dr. N. Seshagiri Memorial
Convention Hall



Open Source Chatbot F/W

Enhancing Interactions, Simplifying Solutions.

Team: Saroj Sekhar Behera, Software Developer
Hemlata Sah, Scientific Technical Assistant-A
Vijay Kumar, Scientific Technical Assistant-A



Brief Description of RASA:

Rasa is an open-source framework for building conversational AI applications, including chatbots, virtual assistants, and voice applications. It offers a robust platform for creating highly customizable and scalable solutions to handle natural language understanding (NLU) and dialogue management.

The Prototype:

We leveraged Rasa to create an advanced enterprise chatbot prototype. This architecture provides accurate, context-aware, and conversationally adaptive responses, ensuring seamless user interactions. The prototype demonstrates a fully customizable conversational AI, built entirely on open-source technologies, showcasing capabilities like intent recognition, entity extraction, and dialogue management. Designed for integration into secure environments, this chatbot eliminates reliance on external cloud services, ensuring data privacy and security. It validates the feasibility of using Rasa's open-source framework for building scalable, intelligent AI solutions tailored to specific organizational needs.

Tools and Technologies Used

RASA:

An open-source machine learning framework for building conversational AI, used for processing user input, determining intents, and managing the conversation flow.

Deployment:

 Localhost

Python:



Backend scripting language used to run the Rasa server and perform NLU and Core-based computations.

Development Framework:

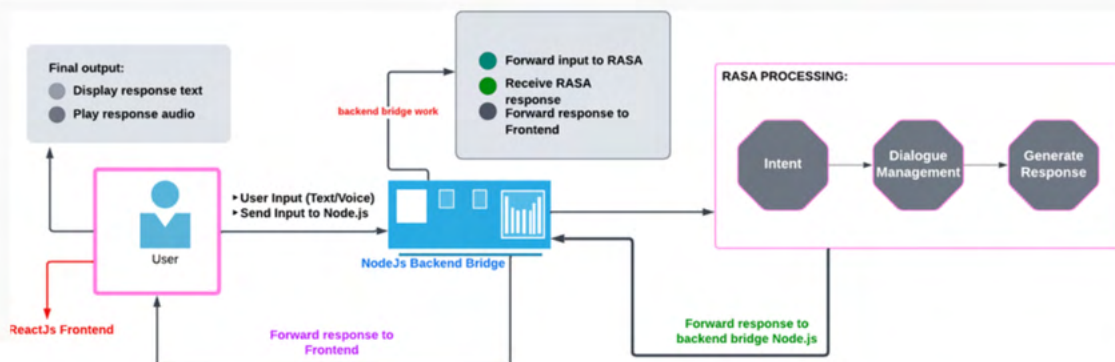


Back end: Node.js



Front end: React

Architecture:



Use Cases:

This Rasa chatbot project integrates advanced natural language understanding (NLU) capabilities with a robust frontend-backend architecture. The system can handle user interactions effectively and provide meaningful responses. Below are the primary use cases for the project:

Example:

- **User:** "What are the services offered by NIC?"
- **Bot:** "NIC provides e-Governance, cloud infrastructure, and cybersecurity services."

Interactive Voice Assistant

Description: With Speech-to-Text (STT) and Text-to-Speech (TTS) features, the chatbot acts as an interactive voice assistant.

Example:

- A visually impaired user can use voice input to ask questions and receive audible responses.

Large Language Model

Empowering Conversations, Unlocking Knowledge.

Team: Subhendu Sekhar Sahoo, Scientific Technical Assistant-A
Manoj Kumar Sethi, Scientific Technical Assistant-A



Brief Description of LLM:

Large Language Models (LLMs) combined with Retrieval-Augmented Generation (RAG) enhance the capabilities of traditional LLMs by integrating external, real-time knowledge through retrieval mechanisms. This allows the model to fetch relevant data from a knowledge base, providing richer, more contextually accurate responses. Unlike standard LLMs that rely solely on their training data, LLMs with RAG can access updated, domain-specific information when needed.

The Prototype:

We selected Langchain, ChromaDB and LLaMA to build an enterprise chatbot. This architecture allows for fast, accurate, and secure responses to complex queries while maintaining low operational costs and high adaptability. The prototype demonstrates a Retrieval-Augmented Generation chatbot built entirely on open-source technologies. This chatbot prototype is ideal for securely-sensitive environments, eliminating reliance on external cloud services and demonstrating the feasibility of integrating open-source tools for advanced AI solutions.

Tools and Technologies Used

Retrieval:

 ChromaDB for vector-based data storage and retrieval.

 LangChain to manage retrieval workflows.

Language Model:

 LLaMA for domain-specific response generation.

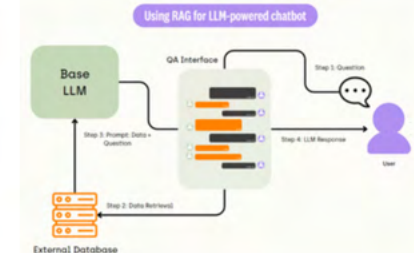
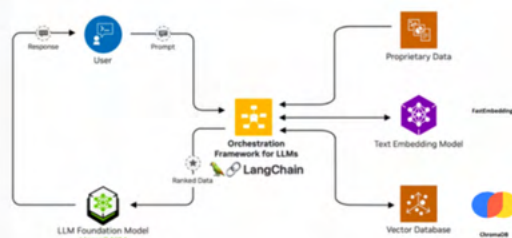
Development Framework:

 Python with PyTorch.

Deployment:

 Flask for RESTful APIs.

Architecture:



Use Cases:

Farmers often face challenges in understanding and accessing information about various government schemes tailored to their needs. The complex eligibility criteria, document requirements, and procedural steps are hard to navigate, especially in rural areas with limited digital literacy. An LLM with RAG-based chatbot can provide personalized, real-time advisory services to farmers, bridging the information gap and making government schemes more accessible.

Farmer's Query:

"What are the technical specifications and unit cost price of the Subjee Cooler?"

Chatbot's Response:

According to the provided context, the technical specifications and unit cost price of the Subjee Cooler are:

Technical Specifications:

The minimum net storage volume of the subjee cooler is 0.75 cum having a minimum dimension of 1 mtr X 1 mtr X 0.75 mtr. Unit Cost Price: The unit cost of the subjee cooler is Rs.50,000/-.

Small Language Model

Efficient Intelligence, Compact Power.

Team: Sumanta Garhnayak, Scientific Technical Assistant-A
Suresh Bairwa, Scientific Technical Assistant-A



Brief Description of SLM:

Small Language Models (SLMs) are designed to perform language-based tasks while requiring less computational power, memory, and time compared to Large Language Models (LLMs). These models are especially valuable in resource-constrained environments or where tasks are domain-specific, such as healthcare, legal, or education sectors.

The Prototype:

In the prototype we have demonstrated the setup and configuring a chat bot based on the Small Language Model completely deployed on-premise without any external dependencies. The prototype is designed and developed using all the open source technologies. This type of on-premise deployment is very helpful in case of security sensitive context.

It demonstrates that Small Language Models, when optimized and fine-tuned, can provide robust, scalable, and efficient solutions to meet modern AI demands in both enterprise and individual use cases.

Tools and Technologies Used

Language Model:



Phi2

Development Framework:



Python with PyTorch.

Deployment:



Flask

Flask for RESTful APIs.



Websocket for socket connection.

Architecture:



Use Cases:

Customer Support Automation

- Deploy an SLM to handle common customer queries via chatbots or automated email responses.
- Customize the SLM to incorporate domain-specific terminology and knowledge.
- Ensure compliance with data privacy regulations by keeping sensitive information on-premises.

HR and Talent Management

- Deploy SLMs for automated resume screening, filtering candidates based on tailored job descriptions.
- Provide real-time assistance to employees through virtual HR assistants.

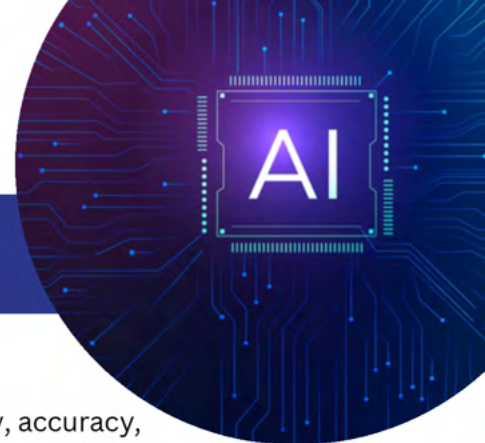
Educational Tools

- Develop on-premise virtual tutors for schools or universities, providing domain-specific knowledge for STEM or humanities subjects.
- Ensure secure handling of student data during interactions.



Guardrail AI

Team: Rajneesh Kant Pandey, Scientific Technical Assistant-A
Sanjay Samaria, Scientific Technical Assistant-A



Brief Description

As AI systems become integral to critical applications, ensuring their reliability, accuracy, and ethical behavior is paramount. Guardrails serve as mechanisms to define boundaries and enforce compliance within AI applications, ensuring that outputs are safe, relevant and aligned with organizational policies.

Different Open-Source Platforms for Implementing Guardrails :

Several open-source platforms provide robust frameworks for adding guardrails to AI applications. Some notable platforms include:

- **Guardrails AI / Guardrails Hub:** Offers a structured approach to validate, filter, and shape AI outputs, safety and reliability.
- **Lang Chain:** Designed to manage and control outputs from language models, integrating seamlessly with guardrail validators.
- **Rasa Guardrails:** Primarily for conversational AI, allowing safe and contextual responses through customizable rules.
- **Hugging Face Transformers:** Includes tools for fine-tuning and controlling AI model behavior.

Validators in Guardrails AI and Our Selection :

- Banned Words Validator
- Profanity Check Validator
- Redundancy Check Validator
- Reading level Validator
- Gibberish Text Validator
- Mentions Drug Validator

Tools and Technologies Used :



Framework: Guardrails AI for input and output validation.

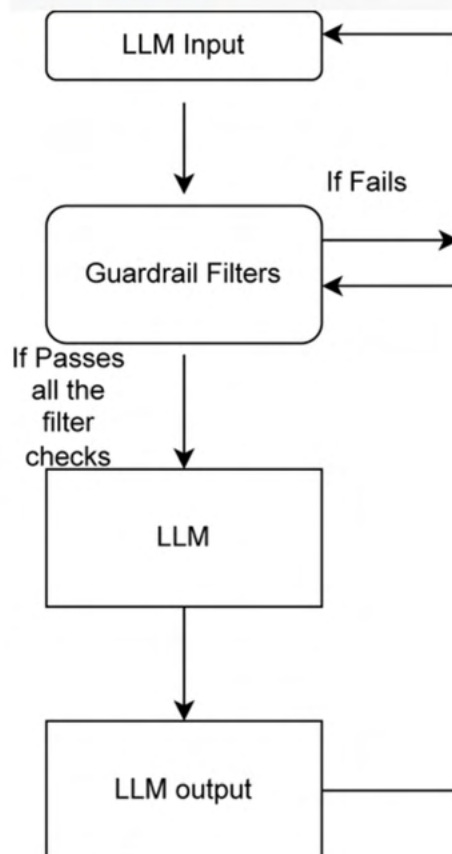


Development: Python, for creating the user interface and API endpoints.



Validation: Multiple custom validators configured to meet specific safety and relevance requirements.

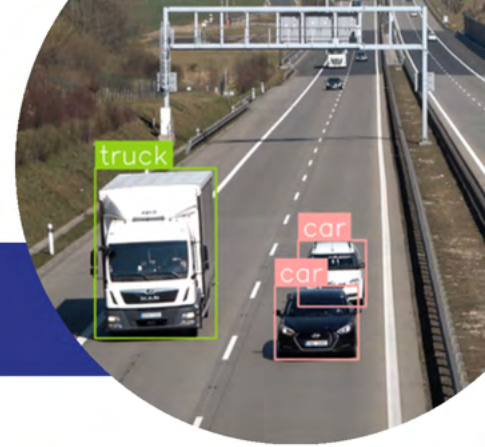
Architecture:



Billion Eye

Yolo8 based Incident reporting system

Team: Dibyatman Khadanga Trainee
Ashutosh Pattnaik Trainee



YOLOv8 and Object Detection

YOLOv8 (You Only Look Once, version 8) is a state-of-the-art model for real-time object detection, segmentation, and classification. It builds upon the efficiency and accuracy of previous YOLO versions while introducing improved architecture, faster inference speeds, and better handling of diverse object scales. YOLOv8 can process images and videos, identifying multiple objects within a single frame with bounding boxes and class labels. This makes it ideal for applications like autonomous driving, surveillance, and city problem detection. Its lightweight design and scalability allow seamless deployment across various platforms, including edge devices and cloud environments.

Prototype Overview

The YOLOv8-based prototype is designed to address urban infrastructure challenges by automating the detection of city problems like potholes, fallen trees, litter, and other issues using computer vision.

The system provides users with an intuitive interface, built using Gradio, where they can upload images of potential problems. The YOLOv8 model processes these images in real-time, identifying and categorizing the detected issues with bounding boxes and class labels.



Tools and Technologies Used:



YOLOv8: Core object detection model for real-time, high-accuracy detection.



Web Scraping: Used bing-image-downloader .



Gradio: Interactive GUI for uploading images and viewing detection results.



Manual Annotation: Created a custom, accurately annotated dataset for training.



GPython: Primary programming language for backend logic and integration.



Custom Database: Designed specifically for urban problem detection needs.

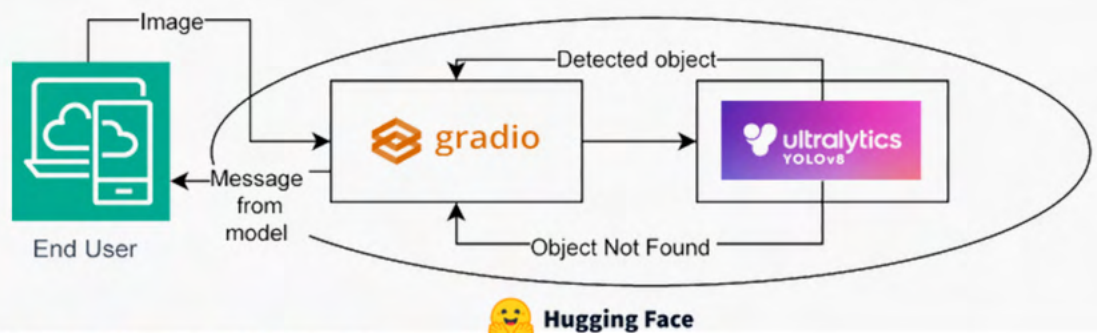


Hugging Face Spaces: Deployment platform for global accessibility.



Pillow (PIL): For image processing tasks like saving and handling images.

Architecture:



Use Case: Smart Incident Reporting System

Citizens use a mobile app to capture and upload images of issues like potholes, accidents, or fallen trees, including GPS and timestamp data. The YOLOv8 model detects anomalies in real time, and metadata helps identify the responsible authority. Notifications are sent to relevant departments for resolution, with progress tracking for transparency.

Vision Language Model

Combining Machine Vision and Semantic Processing Techniques



Team: Raj Anand, Scientific Technical Assistant-A
Shivam, Scientific Technical Assistant-A

Brief Description of VLM:

The Vision-Language Model (VLM) objective is to integrate and align visual and textual information effectively to enable various multimodal tasks. These objectives are critical for tasks such as image captioning, visual question answering (VQA), image-text retrieval.

The Prototype:

The VLM model was developed using llama3.2-vision 11b is a state-of-the-art vision-language model to improve the understanding and integration of visual and textual data. It employs innovative training techniques to align image and text representations effectively, enabling various multimodal applications such as image captioning, visual question answering (VQA).

Tools and Technologies Used:



Python: The Primary Programming Language used for scripting and automation.



Development Framework: Python with PyTorch.

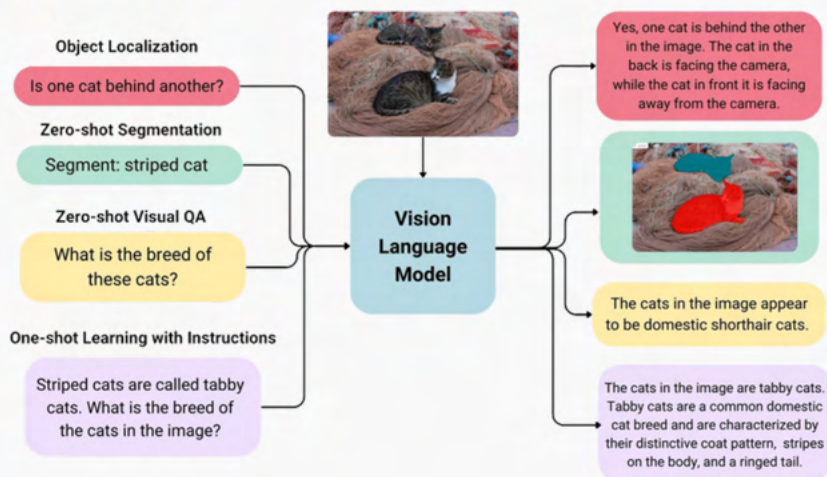


Deployment: Flask for RESTful APIs.



Vision Model: Llama3.2-vision 11b for domain-specific response generation.

Architecture:



User's Query:

"What is the color of hair of girl in the image?"

VLM's Response : "Blonde"



Use Cases:

- **Data Analysis:** Enables efficient analysis of visual and textual data for industries like media, healthcare, and e-commerce.
- **Accessibility Enhancements:** Generates captions and descriptions to aid visually impaired users.
- **Content Management:** Improves tagging, search, and organization of multimedia files.
- **Customer Engagement:** Enhances chatbot capabilities with visual understanding and contextual responses.
- **Education and Training:** Facilitates interactive learning by integrating visual and textual content.

AutoPlateX

Intelligent OCR for Vehicle Plate Recognition

Driving Innovation, Unlocking Smarter Vehicle Recognition

Team: K Rahul
Kondepudi Venkat Durga



Brief Description of the System

AI-Based Number Plate Detection combines YOLO (You Only Look Once) for efficient real-time detection of vehicle number plates with EasyPaddleOCR (powered by PaddleOCR) for accurate text extraction. This project is aimed at automating vehicle identification processes like toll collection, traffic rule enforcement, parking management, and security surveillance. The system integrates object detection and OCR capabilities to provide a seamless, scalable, and highly adaptable solution for smart city and industrial applications.

The Prototype:

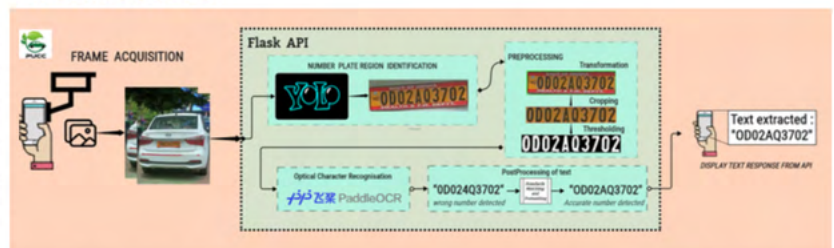
The prototype demonstrates a high-speed number plate detection and OCR system, integrating YOLO for bounding box generation and EasyOCR for text extraction, with preprocessing and postprocessing steps to enhance accuracy and usability. Preprocessing involves image normalization, noise reduction, and cropping irrelevant regions to focus on number plates. Detected bounding boxes from YOLO undergo further preprocessing, including perspective correction, grayscale conversion, and contrast enhancement, to optimize input for EasyOCR.

EasyOCR extracts alphanumeric text from the processed images. Postprocessing includes validating and correcting the extracted text using regular expressions, integrating the results into a database for applications like vehicle tracking or toll management, and triggering actions such as alerts for unauthorized vehicles or automated toll deductions. This workflow ensures robust and scalable performance for real-world scenarios like traffic management, parking systems, and security surveillance.

Use Cases:

- eChallan and PUC Verification:** Simplify the issuance of eChallans for traffic violations and automate the validation of Pollution Under Control (PUC) certificates. This ensures regulatory compliance, reduces the workload of officers, and minimizes errors in enforcement.
- Traffic Violation Management:** Automate the detection of violations such as overspeeding and signal jumping. Integration with the Indian eChallan system ensures accurate fine issuance, reduces manual data entry, and enhances transparency in traffic enforcement.
- Toll Collection on Highways:** Streamline toll operations using automated vehicle identification integrated with FASTag systems. This reduces congestion at toll plazas, minimizes manual intervention, and enhances the efficiency of India's highway network.
- Parking Automation in Urban Areas:** Enable smart parking systems for malls, offices, and public spaces by automating entry, exit, and billing processes. This helps tackle urban parking challenges and reduces delays in densely populated areas.
- Security and Surveillance:** Enhance security in high-risk locations like airports, government buildings, and border areas by identifying unauthorized or flagged vehicles in real-time. This aids law enforcement in ensuring safety and preventing security breaches.

Architecture:



Tools and Technologies Used

Object Detection :



YOLO(You Only Look Once) for number plate detection.



OpenCV for preprocessing and handling image/video inputs.

OCR Extraction:



EasyPaddleOCR for text recognition.

Development Framework:



Python with PyTorch for deep learning model implementation.

Deployment:



Flask for RESTful APIs.



Docker for containerization

AI-Powered Liveliness Detection System

Efficient Intelligence, Compact Power.

Team: Kondepudi Venkata Durga
K Rahul



Brief Description:

Our AI-powered liveliness detection system uses custom CNN models and pretrained architectures to deliver secure and real-time user authentication. A Convolutional Neural Network (CNN) is a type of deep learning model that processes and analyzes visual data, identifying complex patterns and features like textures and facial structures. The system integrates Liveliness Detection and Authentication methods to prevent impersonation and fraud. Leveraging CNN's ability to extract intricate features, it ensures high accuracy in detecting spoof attempts and natural expressions. Designed for scalability and real-time performance, it is ideal for applications like proctoring and authentication.

The Prototype:

We developed a robust AI-powered liveliness detection system for secure authentication and proctoring, integrating spoof detection, distance verification, eye blink detection, and smile detection into a real-time, accurate workflow. The system ensures enhanced security for LL test proctoring and face authentication, effectively preventing impersonation and fraud. It utilizes advanced models like Blaze Face for face detection, the Iris Model for eye landmark tracking, and a custom CNN model for spoof analysis, ensuring comprehensive user verification. The system analyzes natural blinking, smiling, and distance thresholds to detect liveliness and ensure authenticity. Built with open-source tools, it is scalable, cost-effective, and optimized for sensitive environments. This system delivers secure and efficient operations while maintaining adaptability across various use cases.

Tools and Technologies Used

Development Framework:

-  TensorFlow & Keras for model training and optimization.
-  TensorFlow.js for browser-based AI processing.
-  JavaScript and Python.

Models:

-  Based on Convolutional Neural Network

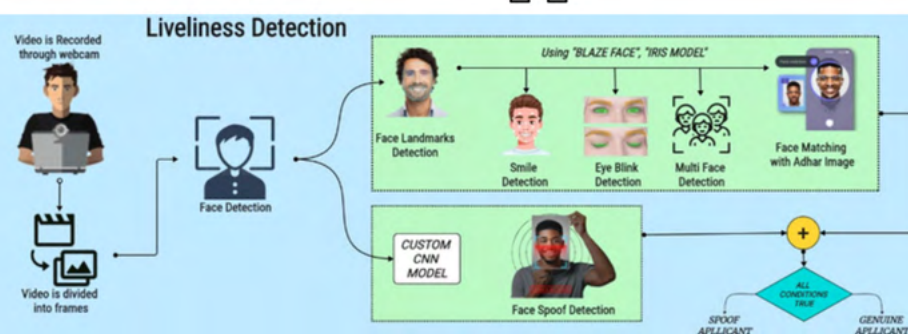
Development Environment:

-  Visual Studio Code for coding and debugging

Deployment:

-  Local Client.

Architecture:



Use Cases:

- Liveliness Verification: Ensures that a real user is present during authentication.
- Online Examination Proctoring: Prevents impersonation during online tests.
- Authentication Systems: Secures access to sensitive systems and online services, ensuring real-time, accurate user authentication.

Deployment in Sarathi:

This system has been successfully deployed in the Sarathi LL Vistless test services for face authentication and proctoring . By incorporating advanced AI models for liveliness detection, spoof detection, and distance verification, it ensures secure and accurate user authentication. This deployment plays a critical role in significantly reducing scam-related activities during exams and login processes, ensuring the integrity and security of the system.

Smart Water Meter Reader

Revolutionizing Water Consumption Monitoring

Team: Siddhartha Kumar Mandal, Scientist-C

Brief Description of the System

The **Smart Water Meter Reader** is a cutting-edge mobile application designed to revolutionize water meter reading for industries. By leveraging Optical Character Recognition (OCR) technology, the app automates the extraction of water meter readings, ensuring error-free and timely data reporting. This solution eliminates the challenges of manual meter reading, reduces human errors, and enhances operational efficiency. Currently adopted by over 1200 industries registered with the Industrial Water Resource Department, the app provides a robust and scalable solution for real-time water consumption monitoring and management.

The Prototype:

The prototype demonstrates a streamlined process for capturing and managing water meter data. Using OCR technology, the app enables users to scan water meters, extract the numeric readings, and send the data to a central server. Each reading includes critical metadata such as the timestamp, geographical coordinates (latitude and longitude), and the scanned image. This ensures accurate tracking of water usage while offering a seamless user experience. The prototype highlights the app's capability to reduce manual intervention, improve accuracy, and provide real-time data synchronization.

Architecture:

User Interaction Layer:

- Flutter-based mobile app for capturing images of water meters.

Backend Processing:

- Node.js server handles incoming data, performs validations, and stores readings in the MySQL database.

Data Transmission:

- Geolocation, timestamp, and images are transmitted along with readings for context and accuracy.

OCR and Preprocessing:

- Embedded OCR engine processes captured images to extract numeric readings, normalizing and validating the data.

Centralized Storage and Reporting:

- MySQL database supports robust reporting and analytics for regulatory and operational use cases.

Use Cases:

- Industrial Water Management:** Automates water consumption tracking for over 1200 industries, enabling precise billing and usage reporting.
- Error-Free Meter Reading:** Eliminates manual data entry, reducing errors and improving accuracy in water usage monitoring.
- Regulatory Compliance:** Simplifies submission of water usage reports, ensuring compliance with regulations.

Tools and Technologies Used:

Development Environment:



Flutter: Cross-platform mobile app framework for a responsive and intuitive user interface.

OCR Technology:



Tesseract OCR: For extracting numeric data from images of water meters.

Library:



Google's ML Kit

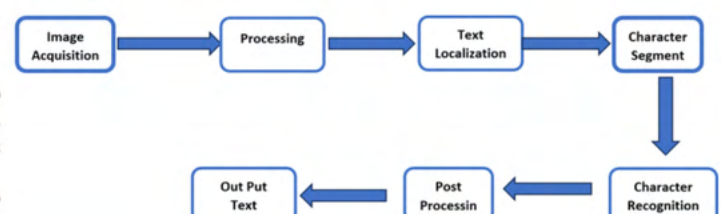
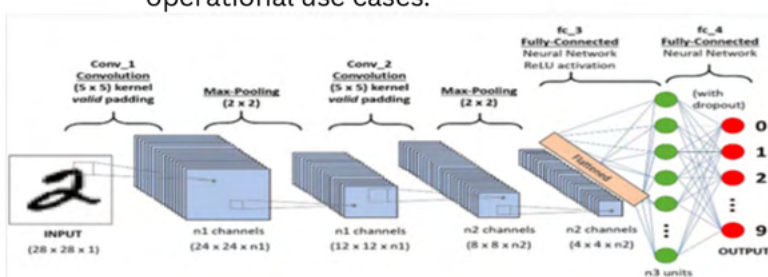
Backend Development:



Node.js: Backend framework for handling API requests and data processing.



MySQL: Database for secure and efficient storage of meter readings and metadata.



Yield Estimation Using DNN

Harnessing Deep Learning to Cultivate Precision and Sustainability in Agriculture

Team: Mr. Vinay Kumar Tiwari, Scientific Technical Assistant-A
Mr. Surendra Baskey, Scientific Technical Assistant-A
Mr. Vinay Kumar, Scientific Technical Assistant -A



DNN Overview

A Deep Neural Network (DNN) is a type of artificial neural network characterized by the presence of multiple layers situated between the input and output layers. Each layer comprises neurons that perform both linear and nonlinear transformations on the input data, thereby facilitating the learning of complex patterns and representations. In the context of this project, our DNN model is specifically formulated to predict seed production by analyzing a range of agricultural features.

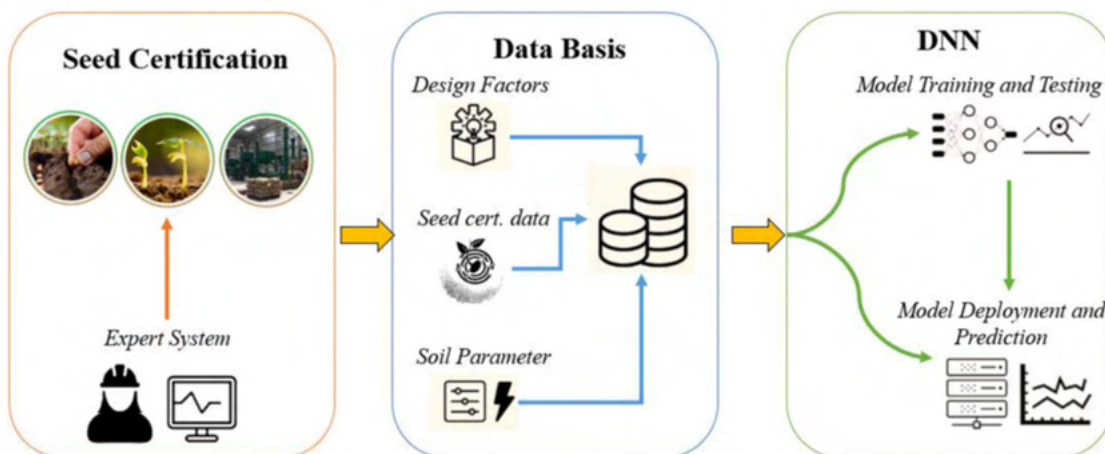
Prototype

The prototype developed aims to forecast seed production for different crops and their varieties across various seasons. With the help of data captured from the SATHI portal during the seed certification process, our team has created an advanced model that employs deep learning methods to estimate the amount of seeds produced. This model offers essential insights for farmers, agricultural planners, researchers, and policymakers, supporting them in making informed decisions, optimizing resource use, and enhancing agricultural productivity.

Tools and Technologies Used:



Architecture:



Contribution to Sustainable Development Goals (SDGs)



Enhances food security and sustainable agriculture through precise seed production predictions, contributing to hunger alleviation



Optimizes resource utilization and minimizes waste, thereby fostering sustainable production methodologies



Advances agricultural innovation and resilient infrastructure through the integration of sophisticated machine learning techniques

Use Cases

- **Farmers:** Benefit from accurate seed production predictions based on real-time and historical data for effective planning.
- **Agricultural Planners:** Leverage data-driven decisions to enhance agricultural productivity.
- **Researchers:** Analyze factors influencing seed production to foster the development of innovative techniques.
- **Policymakers:** Formulate sustainable agricultural practices to bolster food security.

Robotic Process Automation

Automating Tasks, Empowering Businesses

Team: Himanshu Ranjan, Scientific Technical Assistant-A
Naveen Lohan, Scientific Technical Assistant-A



Brief Description of RPA:

Robotic Process Automation (RPA) is a technology that uses software robots or "bots" to automate repetitive and rule-based tasks in business processes. It enables organizations to improve efficiency, accuracy, and cost-effectiveness by automating tasks such as data entry, report generation, and transaction processing. RPA bots interact with applications and systems in the same way a human would, but with greater speed and precision. This technology is widely used in industries like finance, healthcare, and customer service to streamline operations.

Model Description:

We selected TagUI, Python to automate the NIC Online Examination platform (Pbox). This automation was aimed at improving efficiency, reducing the manual effort of interacting with the exam portal, and testing various functionalities such as login, exam navigation, and submission. By automating these processes, the goal was to simulate user behavior, conduct exam-related actions, and help in repetitive task automation during an examination session.

Tools and Technologies Used:



Python: The primary programming language used for scripting and automation.



TagUI: Open-Source automation tool.



Web Browser: Used for running the automation script and interacting with the NIC online exam portal.

Architecture:



Use Cases:

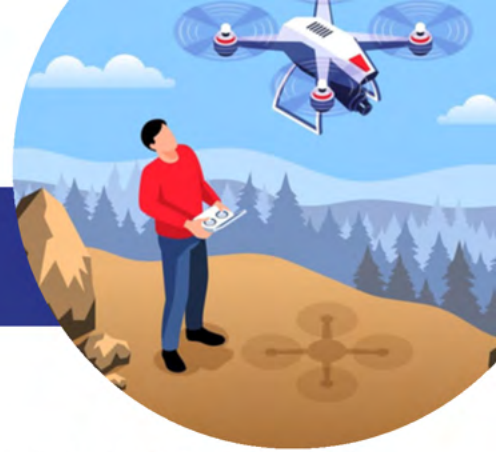
Manual work often takes a significant amount of time due to repetitive tasks and the need for constant attention to detail. This can lead to errors, as humans may overlook small details or become fatigued. By implementing RPA, these tasks can be automated, increasing efficiency and accuracy. Automation reduces human error, speeds up processes, and allows employees to focus on more strategic, value-added activities.

- **Invoice Processing in Finance :** RPA can automate the extraction of data from invoices, validate it against purchase orders, and enter it into an accounting system. For example, a company using RPA to process hundreds of invoices daily can significantly reduce errors and save time compared to manual data entry.
- **Customer Service Chatbots :** RPA can be used to handle customer inquiries by automating responses through chatbots. For instance, a telecom company can use RPA to answer common customer queries like balance inquiries, bill payments, and service activation, freeing up human agents to handle complex issues.
- **Supply Chain Management :** RPA can automate inventory tracking and order processing. For example, an e-commerce company can use RPA to monitor stock levels, automatically reorder products when inventory runs low, and update the warehouse management system, ensuring smooth and efficient supply chain operations without manual intervention.

PaliGemma RSVQA

Remote Sensing- Vision Question & Answer

Team: Ashutosh Pattnaik ,Trainee
Dibyatman Khadanga ,Trainee
Harshpreet Singh Sehgal ,Trainee



PaliGemma and Remote Sensing - Image Vision Question and Answer

PaliGemma2 is a pre-trained language model based on the Transformer architecture, designed for tasks like natural language understanding and generation. It can be fine-tuned for various applications like visual question answering . In the context of Remote Sensing Image Visual Question Answering (RS-IVQA), PaliGemma2 is adapted to process textual questions and generate answers based on insights from remote sensing images. RS-IVQA integrates computer vision and natural language processing to interpret satellite or drone imagery and answer user queries, making it a powerful tool for applications like disaster monitoring, land use analysis, and urban planning. This combination of PaliGemma2 and RSVQA enables automated and accessible geospatial data analysis.

Prototype Overview

The Remote Sensing Image Visual Question Answering (RS-IVQA) system uses a PaliGemma2 model to answer natural language questions about satellite or aerial imagery. By combining image processing and language understanding, it supports tasks like disaster management, urban planning, and environmental monitoring. Tools like Hugging Face and CLIP enable seamless integration and deployment, making it a powerful asset for researchers, governments, and organizations. The system provides users with an intuitive interface, built using Gradio, where they can upload images.



Potential Use Cases:

- **Disaster Management:** Quickly analyze satellite imagery to assess flood impact, detect landslides, or monitor wildfire spread.
- **Urban Planning:** Support decision-making by identifying land use patterns and infrastructure changes.
- **Infrastructure Monitoring:** Track real-time changes in roads, bridges, or buildings.
- **Traffic Management:** Analyze congestion patterns and incidents.
- **Agriculture:** Enhance crop monitoring and yield prediction through precise visual and textual data integration.
- **Environmental Monitoring:** Detect deforestation, waterbody changes, pollution levels, urban encroachment, and green cover health.
- **Disaster Preparedness:** Provide early warnings for floods, landslides, or urban heat zones.

Tools and Technology Used



PaliGemma2: Vision Language Model used for Remote sensing Image Question and Answer



Gradio: Interactive GUI for uploading images and viewing detection results.



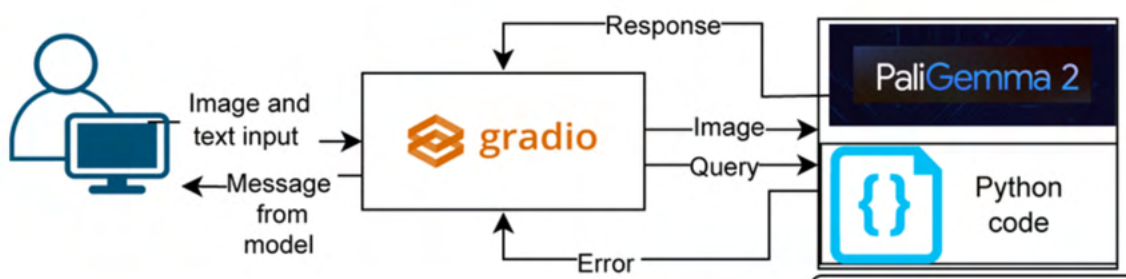
Pillow (PIL): For image processing tasks like saving and handling images.



Google Colab: For running the code on a cloud-based environment with GPU/TPU support



Python: Primary programming language for backend logic and integration.





National Information Centre, Meity



/National Informatics Centre



NICMeity



@NICIndia



nicmeity



<https://odisha.nic.in>

National Informatics Centre, Odisha State Centre, Unit IV, Sachivalaya Marg, Bhubaneswar-751001,
Tel: +91-674-2508438, Email: sio-ori@nic.in, Web: <https://odisha.nic.in>



इलेक्ट्रॉनिक्स एवं
सूचना प्रौद्योगिकी मंत्रालय
MINISTRY OF
ELECTRONICS AND
INFORMATION TECHNOLOGY
सत्यमेव जयते



एनआईसी
National
Informatics
Centre