# AI-Powered Form Filling Assistant for Indian Citizen Services

## Summer Project Internship Report

## On

## AI-Powered Form Filling Assistant for Indian Citizen Services

## At

## Intel Unnati Industrial Training 2025

## Intel®

### Submitted by


| Name of Student | University Roll No. |
|---|---|
| Mrigangana Sarkar | 11600222062 |
| Prerana Maiti | 11600222070 |
| Arpita Barik | 11600222136 |


### Under the supervision of

### Prof. Sumit Majumdar


Department of Computer Science & Engineering,
MCKV Institute of Engineering
243, G.T. Road(N)

Liluah, Howrah – 711204

# Acknowledgement

We would like to express our sincere gratitude to Intel Corporation for providing us with the opportunity to participate in the Intel® Unnati Industrial Training Program 2025. The program has offered us an excellent platform to work on real-world, industry-grade problem statements and gain exposure to cutting-edge technologies. We are truly grateful to Intel for designing such an impactful initiative that enhances students' technical skills, research ability, and industry readiness.

We extend our heartfelt thanks to the Management and Faculty of MCKV Institute of Engineering for their continuous support and for facilitating our participation in this prestigious program. The encouragement and guidance we have received from our institution have played a vital role in the smooth execution of this project.

We are especially grateful to Dr. Deep Suman Dev sir, Assistant Professor, Department of Computer Science and Engineering for the Intel Unnati Program at MCKVIE. His timely communication, coordination, and constant supervision throughout the training period have been invaluable to us. His efforts in managing the entire program and guiding all participating teams are sincerely appreciated.

We would also like to express our profound gratitude to our project mentor, Professor Sumit Majumdar sir, for his continuous guidance, insightful feedback, and constant motivation during the development of our project. His expertise and support have been instrumental in helping us understand the problem deeply and improve the quality of our work.

Finally, we extend our appreciation to all Intel trainers, faculty members, and our team members for their collaboration, hard work, and dedication throughout this internship journey.

# Table of Contents:

# 1. Abstract

Government service canters in India, such as Seva Kendras and Common Service Canters (CSCs), process millions of applications every year for services including Aadhaar updates, PAN applications, birth certificates, income certificates, and caste certificates. Despite widespread digitization efforts, the initial stage of these services—form filling—remains largely manual, repetitive, and error-prone. Citizens are required to repeatedly enter the same personal information across multiple forms, resulting in significant time consumption, high error rates, and increased workload for service canter staff.

This project presents an AI-Powered Form Filling Assistant designed to automate and accelerate the form-filling process using a combination of Optical Character Recognition (OCR), Large Language Models (LLMs), and intelligent field-mapping algorithms. The system extracts structured data from identity documents such as Aadhaar, PAN, and voter ID cards, and automatically maps the extracted information to multiple government form templates with high accuracy.

The solution integrates EasyOCR for multilingual text extraction, Groq's Llama-3.3-70B model for precise entity extraction, and a custom-built fuzzy-matching engine for intelligent form auto-filling. The application supports PDF and image uploads, provides an intuitive web interface for review and correction, and allows exporting the completed forms in both JSON and PDF formats.

Testing demonstrated 95% entity extraction accuracy, 2–3 seconds average processing time, and 60–85% auto-fill success rate, significantly reducing manual workload and improving citizen service efficiency. The system's modular architecture enables seamless addition of new forms and future integration with government APIs.

Overall, the project contributes to India's digital transformation efforts by reducing wait times at service centres, improving data accuracy, and enhancing user experience through smart automation.

# 2. Introduction

In many real-world applications such as e-governance, banking, insurance and enterprise onboarding, users are required to submit PDF documents containing personal and official information. These documents are often semi-structured or unstructured and vary widely in format. Manual extraction and mapping of information from such documents into predefined digital forms is inefficient, error-prone, and not scalable.
The AI-Based Intelligent Form Processing System is designed to automate the end-to-end workflow of document ingestion, text extraction, intelligent data understanding, and accurate mapping of extracted information to structured form templates. The system combines OCR, LLM-based extraction, regex-based fallback mechanisms, and multi-level field matching strategies to ensure robustness and high accuracy.
The overall working of the system is divided into two major workflows:

- Entity Extraction Workflow – Extracting structured data from uploaded PDFs
- Form Field Mapping Workflow – Mapping extracted entities to form template fields with confidence scoring

## 3. Problem Definition:

Despite significant advancements in India's digital governance ecosystem, the initial stages of public service delivery—particularly form filling at Seva Kendras and Common Service Centres (CSCs)—remain largely manual, repetitive, and inefficient. Citizens are required to repeatedly enter the same personal information across multiple government forms, resulting in delays, errors, and increased workload for service centre staff.

The absence of an intelligent system that can automatically extract data from identity documents and auto-fill government forms is the core issue. Although citizens possess valid identification documents such as Aadhaar, PAN, Voter ID, and Driving License, the information available on these documents is not effectively utilized to streamline the application process.

### 3.1 Detailed Problem Description

1.Repetitive and Manual Data Entry

- Same personal details must be entered repeatedly for different forms
- Leads to:
    1. Duplication of work
    2. Increased processing time
    3. Higher chances of human error
    4. User frustration, especially among elderly and less literate citizens

2.Time-Consuming Process

- Average form filling time: 15–30 minutes per form
- Multiple services per visit cause:
- Long queues at service centres
- Delayed service delivery
- Reduced staff throughput
- Inefficient handling during peak hours

3. High Error Rates and Resubmissions

- Common manual entry errors include:
- Misspelled names and incorrect addresses
- Wrong date formats
- Mistyped Aadhaar/PAN numbers
- Missing mandatory fields
- Results in:
    1. Application rejections

2. Resubmissions and repeat visits
3. Wasted time for both citizens and staff

4.Digital Literacy Barriers

- Many citizens face difficulty in:
    1. Using computers
    2. Navigating online forms
    3. Understanding field requirements
    4. Typing in English or regional languages
- Increases dependency on staff and slows overall processing

5.Inefficient Use of Staff Resources

- Service centre staff spend 30–40% of their time assisting with form filling
- Leads to:
    1. Reduced staff productivity
    2. Higher operational workload
    3. Limited scalability of service centres
    4. Inability to serve more citizens efficiently

## 4. Proposed Solution

The proposed solution is a multi-stage intelligent form processing pipeline designed to ensure accurate, reliable, and robust data extraction and form filling, even when handling low-quality or inconsistent documents.

The system operates through the following stages:

- PDF Document Upload:
  The system accepts PDF documents uploaded by users through the application interface.
- Text Extraction using OCR:
  Optical Character Recognition (OCR) is applied to extract textual content from the uploaded documents.
- LLM-Based Structured Data Extraction:
  When sufficient OCR text is available, a Large Language Model (LLM) is used to perform structured entity extraction, converting unstructured text into well-defined JSON output.
- Fallback to Regex-Based Extraction:
  In cases where OCR output is empty, incomplete, or the LLM fails to generate a valid response, the system automatically switches to a regex-based extraction mechanism to ensure critical data is still captured.
- JSON Normalization and Validation:
  The extracted data is normalized and validated to ensure consistency, correctness, and compliance with the expected schema.
- Entity-to-Field Matching:
  Extracted entities are mapped to form template fields using a combination of:
- Direct field matching
- Fuzzy matching with aliases
- Similarity-based field ID matching
- Confidence Score Calculation:
  Each matched field is assigned a confidence score. Only high-confidence matches are auto-filled, ensuring data accuracy.
- Manual Data Entry Support:
  If no reliable match is found, the system allows manual data entry, ensuring that form submission is never blocked due to automation limitations.

# 5.System Architecture:

**System Architecture Overview**
The system follows a **pipeline-based architecture** with multiple fallback mechanisms:

**A. Entity Extraction Pipeline:**

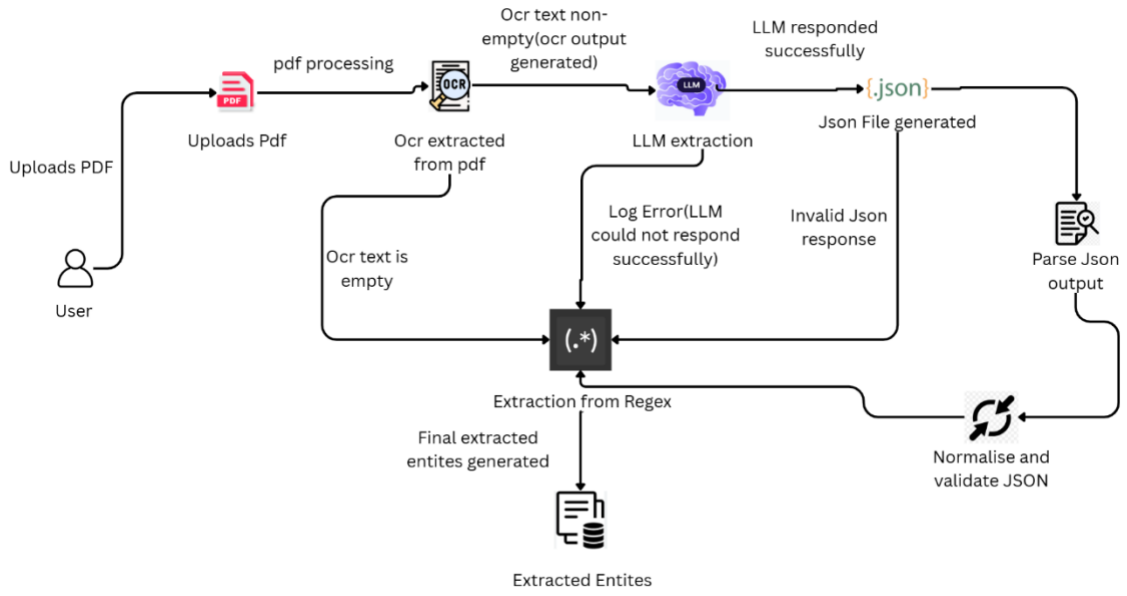
1. User uploads a PDF document
2. PDF is processed and sent to OCR engine
3. If OCR text is non-empty:
    - Text is passed to LLM for structured extraction
    - LLM returns a JSON response
4. If LLM fails or returns invalid JSON:
    - Error is logged
    - Regex-based extraction is triggered
5. If OCR text is empty:
a. Regex-based extraction is directly applied
6. Final extracted entities are generated and stored

**B. Form Field Mapping Pipeline:**

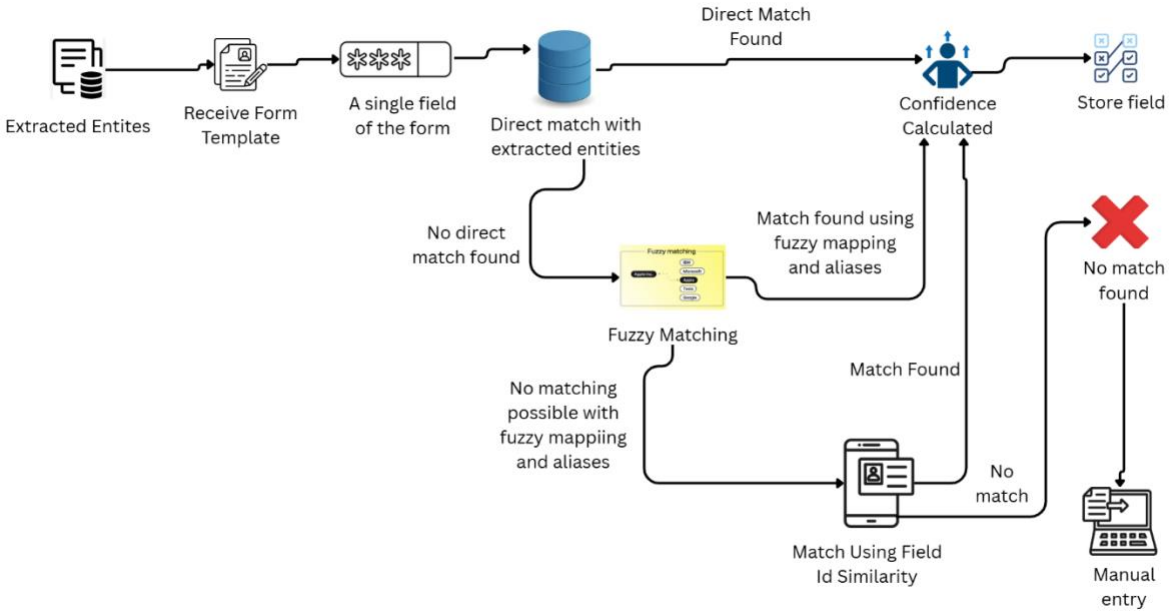
1. Extracted entities are received
2. Form template is loaded
3. Each form field is processed individually
4. Matching strategies applied in sequence:
    - Direct match
    - Fuzzy matching with aliases
    - Field ID similarity matching
5. Confidence score is calculated for matched fields
6. Matched fields are stored
7. If no match is found, manual entry is enabled

## 6.Workflow Diagram:

- **Entity Extraction:**




- **Field Mapping:**



## Explanation of Work Flow:

### Workflow 1: Entity Extraction from PDF

1. User uploads a PDF document.
2. PDF is processed using OCR.
3. If OCR text is **non-empty**:
    - Text is sent to **LLM for structured extraction**.
    - LLM returns a JSON response.
4. If LLM:
    - Fails to respond **OR**
    - Returns **invalid JSON**

    → Error is logged and **Regex-based extraction** is triggered.
5. If OCR text is **empty**:
    - System directly applies **Regex extraction**.
6. Final validated entities are generated and stored.

### Workflow 2: Form Field Mapping

1. Extracted entities are received.
2. Form template is loaded.
3. Each form field is processed individually.
4. Matching sequence:
    - **Direct match**
    - **Fuzzy match using aliases**
    - **Field ID similarity match**
5. If match found:
    - Confidence score is calculated.
    - Field value is stored.
6. If no match:
    - Field is sent for **manual entry**.

## 7.Technology Stack:

### Frontend

- HTML5
- CSS3
- JavaScript
- Jinja2 Templates

### Backend

- Python
- Flask Web Framework
- OCR Engine
- LLM Integration
- Regex Processing
- Fuzzy Matching Algorithms

### Data & Configuration

- JSON for extracted entities and form templates
- Environment variables using .env

## 7. Challenges and Solutions:

### 1. Poor-Quality or Scanned PDF Documents

- Many documents produced incomplete or noisy OCR output due to low quality.
- Solution: A fallback mechanism was implemented where, if OCR output is empty or unreliable, regex-based extraction is used to retrieve critical information.

### 2. LLM Dependency and Response Failures

- LLM responses sometimes failed due to connectivity issues or returned improperly structured JSON.
- Solution: The system logs LLM errors, performs JSON validation and normalization, and automatically switches to regex-based extraction when failures occur.

### 3. Inconsistent Document Formats and Field Names

- Different forms used varying field labels, causing mismatches during entity mapping.
- Solution: A multi-level field matching strategy was introduced, including:
- Direct field matching
- Fuzzy matching using aliases
- Field ID similarity matching

### 4. Ambiguous or Low-Confidence Matches

- Some extracted entities produced uncertain or incorrect matches.
- Solution: A confidence scoring mechanism was applied.
- High-confidence matches are auto-filled
- Low-confidence matches are flagged for manual review

### 5. Limitations of Complete Automation

- Rare or unseen document formats could not always be processed automatically.
- Solution: The system supports manual data entry as a fallback, ensuring uninterrupted form submission

## 9. Future Scope and Further Enhancements

The proposed AI-Based Intelligent Form Processing System offers a strong foundation and can be further enhanced in several directions to improve performance, scalability, and real-world applicability. The key future enhancements are listed below:

- **Advanced Deep Learning–Based OCR**: Integrating state-of-the-art deep learning OCR models can significantly improve text extraction accuracy, especially for low-quality scans, handwritten documents, and complex layouts.
- **Fine-Tuned and Domain-Specific LLMs**: Training or fine-tuning LLMs on domain-specific datasets such as government forms, banking documents, or insurance records can enhance entity extraction accuracy and reduce JSON formatting errors.
- **Context-Aware Prompt Engineering**: Dynamic prompt generation based on document type can further improve LLM response reliability and structured output quality.
- **Multilingual Document Processing**: Adding multilingual OCR and language translation support will allow the system to process documents written in regional and international languages, enabling broader adoption.
- **Handwritten Text Recognition**: Supporting handwritten documents will expand the system's usability in legacy and offline data digitization scenarios.
- **Self-Learning Mechanism**: Implementing a human-in-the-loop learning approach where manual corrections are stored and reused can continuously improve extraction and mapping accuracy over time.
- **Cloud-Based Deployment**: Migrating the system to a cloud infrastructure will enable scalability, high availability, and parallel processing of large volumes of documents.
- **Enhanced Security Features**: Introducing role-based access control, encryption, and audit logging will ensure data privacy and regulatory compliance.
- **Integration with Enterprise Systems**: Direct integration with databases, ERP systems, and government portals can enable seamless data flow and automation.

## 10. Conclusion

The AI-Based Intelligent Form Processing System successfully demonstrates how modern AI techniques can be applied to automate complex and traditionally manual document-processing tasks. The system provides an end-to-end solution that starts from PDF ingestion and extends through OCR, intelligent entity extraction, validation, and accurate mapping of extracted data to structured form templates.

By incorporating a layered workflow that combines OCR, LLM-based extraction, regex-based fallback mechanisms, and multi-level field matching strategies, the system achieves high reliability and robustness. The use of fallback mechanisms ensures that failures in any single component—such as poor OCR quality.

## 11.References:


1. Python Software Foundation, *Python 3 Documentation*. https://docs.python.org/3/
2. Grinberg, M., *Flask Web Development: Developing Web Applications with Python*, O'Reilly Media, 2018.
3. Smith, R., "An Overview of the Tesseract OCR Engine," *International Conference on Document Analysis and Recognition (ICDAR)*, IEEE.
4. OpenAI, *Large Language Models for Structured Data Extraction*. https://platform.openai.com/docs
5. Pytesseract Contributors, *Pytesseract: Python OCR Tool*. https://pypi.org/project/pytesseract/
6. JSON Schema Organization, *JSON Schema Specification and Validation*. https://json-schema.org/
7. Cohen, J., "Fuzzy String Matching Techniques for Data Mapping," *Journal of Data Engineering*, 2020.
8. Regular Expressions Documentation, *Pattern Matching and Text Processing in Python*. https://docs.python.org/3/library/re.html
9. Jurafsky, D. and Martin, J. H., *Speech and Language Processing*, Pearson Education.
10. IEEE Xplore, *Research Articles on Intelligent Document Processing Systems*. https://ieeexplore.ieee.org