



## PROJECT REPORT ON

### **AI-Powered Form Filling Assistant for Indian Citizen Services** Submitted By

P. Keerthana Reddy (247Y1A67E7)

Sk.Karishma Yaseen (247Y1A67E6)

Pottalapola Venkatesh (237Y1A67C4)

**COMPUTER SCIENCE AND ENGINEERING (Data**

**Science) Marri Laxman Reddy Institute of Technology and  
Management**

**Under the Guidance of  
Mr.P.Jagan Mohan**

Submitted to

**Intel® Unnati Program**

Supported by

**Edgate Technologies**

**In partnership with**

**Marri Laxman Reddy Institute of Technology and Management (MLRITM)**



**Date of Submission:** October 2025

## INDEX

Chapter	Title	Page
Chapter1	Introduction	1
Chapter2	Literature Survey	2-5
Chapter3	Analysis	6-7
Chapter4	Design	8-10
Chapter5	Implementation & Results	11-14
Chapter6	Testing and Validation	15-16
Chapter7	Conclusion & Future Enhancements	17-18

## FIGURES

Figure	Description	Page
Fig 1.1	Upload Interface	11
Fig 1.2.1	OCR Extension Process	11
Fig 1.2.2	Entity Mapping Example	11
Fig 1.2.3	Template Auto-Fill Outputs	12
Fig 2.1	Multilingual Voice Input	12
Fig 2.2		12
Fig 3.1	Top Anomalous Flows	13
Fig 3.2	Anomaly Score Distribution	13
Fig 4.1	Behavioural Cluster Analysis	13
Fig 4.2	Temporal Pattern Analysis	14
Fig 5	Real time Threat Assessment	14

# Problem Statement

## Description:

Indian citizens frequently face the tedious process of filling multiple government service forms at Seva Kendras for certificates, licenses, and welfare schemes. Manual entry leads to repetitive work, high error rates, and accessibility challenges for citizens with language or literacy barriers. This project aims to build an AI tool that auto-fills these forms based on uploaded documents (Aadhaar, PAN, voter ID, etc.), reducing manual effort and errors, and ensuring faster, more accurate, and inclusive digital form filling.

## Agenda:

- Automate extraction of citizen details from uploaded documents using OCR and NLP.
- Accurately map extracted entities to appropriate form fields.
- Enable multilingual and voice-based form filling.
- Provide review and edit options before submission.
- Ensure data privacy and low latency performance.

## ABSTRACT

*AI-Powered Form Filling Assistant for Indian Citizen Services* is designed to automate the manual and time-consuming process of filling government service application forms. Citizens often struggle with repetitive data entry while applying for certificates, licenses, or welfare schemes at Seva Kendras. This project leverages Artificial Intelligence (AI) technologies specifically Optical Character Recognition (OCR) and Natural Language Processing (NLP) to extract essential details such as name, date of birth, and address from official documents like Aadhaar, PAN, and Voter ID.

Once extracted, the system automatically maps the recognized entities to predefined form templates, generating completed digital forms ready for review, editing, and download. It supports multilingual inputs and voice-based interactions to ensure accessibility for all users, including those with literacy or language barriers.

The assistant is implemented as a web-based interface using Streamlit, integrated with OCR and NLP back-end models. The proposed solution significantly reduces human effort, minimizes data-entry errors, and accelerates citizen-service delivery. By combining AI automation with inclusive design, this project contributes to India's vision of digital governance and efficient citizen-centric service transformation.

# CHAPTER 1

## INTRODUCTION

### 1.1 Motivation

Indian citizens frequently face the tedious process of filling multiple government service forms at Seva Kendras for certificates, licenses, and welfare schemes. Manual entry leads to repetitive work, high error rates, and accessibility challenges for citizens with language or literacy barriers. The motivation of this project is to simplify and automate this process using AI, ensuring faster, more accurate, and inclusive digital form filling.

### 1.2 Problem

Citizens often need to fill in multiple government service forms at Seva Kendras (e.g., for certificates, licenses, and welfare schemes). This project aims to build an AI tool that auto-fills these forms based on uploaded documents (Aadhaar, PAN, voter ID, etc.), reducing manual effort and errors.

### 1.3 Solution outline

- Document Upload & Input
- OCR & Data Extraction
- AI Entity Recognition
- Smart Form Auto-Filling
- Review & Edit Interface
- Output & Compliance

### 1.4 Scope

The system will support Indian citizen service forms such as birth/death certificates, domicile certificates, and welfare applications. It will process PDFs or images, extract data using OCR, and fill pre-defined templates. Future extensions may include direct API submissions to government portals.

### 1.5 Objective

- **Reduce manual workload** — Auto-fill majority of fields using extracted data.
- **Improve accuracy** — Maintain >90% accuracy for entity extraction from documents.
- **Faster processing** — Complete form filling within 3–5 seconds per document.
- **Enhance accessibility** — Support multi-language UI and voice input for ease of use.

- **Minimize user errors** — Provide validation and correction suggestions automatically.
- **Scalable solution** — Support multiple government forms and future API integrations.

## CHAPTER 2

# LITERATURE SURVEY

### Literature Survey on Techniques Used

#### OCR and Text Detection:

- **Overview:** Detect and recognise text from images or scanned documents (both printed and handwritten) so it can be processed by downstream systems
- **Applications:** Essential for digitising citizen documents (IDs, certificates), extracting information from uploads, enabling editable/searchable content, and automating paper-based workflows
- **Recent Works:**
  - Generative OCR models (VISTA-OCR) combining detection + recognition in one unified model (2025).
  - OCR adaptation for Hindi using Vision-Language Models to improve Indic script recognition (2025).
- **Literature Note:** Recent research focuses on deep-learning and layout-aware OCR systems for document text extraction.

#### Handwritten Text Recognition (HTR):

- **Overview:** Conversion of handwritten documents into machine-readable text; handles large style/script variability.
- **Applications:** Enables extraction of handwritten fields on citizen-uploaded documents so they can be auto-filled into government forms.
- **Recent works:** 2025 survey on HTR methods; 2024 model combining handwritten text + layout analysis; India-specific review on multilingual HTR.
- **Literature Note:** Key reviews covering evolution of HTR, script diversity, and hybrid handwriting and print recognition.

#### Document Classification & Layout:

- **Overview:** Automatically identify the document type (classification) and detect structural regions (layout parsing) such as headers, tables, forms, and text blocks.
- **Applications:** Enables routing of documents to the correct workflow, supports correct extraction of fields based on structure, and helps in handling diverse form templates in government service settings.

- **Recent Works:**

- > New unified layout-detection models capable of handling many document types with high accuracy and speed.
- > Graph -based methods improving structural understanding of document elements.
- > Vision-Language models using reinforcement learning to better capture logical reading order and layout.

- **Literature Note:**

A 2024 survey “Document Parsing Unveiled” covers modular pipelines and end-to-end vision-language models for layout analysis and classification.

## **Named Entity Recognition (NER) & Entity Extraction:**

- **Overview:** Detects and extracts key information like names, dates, addresses, and ID numbers from text.
- **Applications:** Converts OCR output into structured fields for auto-filling government forms with correct labels.
- **Recent works:** Transformer-based and multilingual NER surveys (2023–2025) focusing on few-shot learning, nested entities, and Indian-language support.
- **Literature Note:** Deep-learning NER reviews highlighting evolution from CNN/RNN models to advanced LLM-based entity extraction.

## **Form Mapping & Key-Value Extraction:**

- **Overview:** The process of identifying labelled fields (keys) and their corresponding contents (values) in a form and mapping them to structured fields.
- **Applications:** The process of identifying labelled fields (keys) and their corresponding contents (values) in a form and mapping them to structured fields.
- **Recent works:** Work such as XForm Parser (2024) demonstrates a multilingual multimodal form parser for key-value extraction.
- **Literature Note:** A Region-based Approach to Form-style Document Understanding (2022) uses region prediction for KIE in forms.

## **Speech / Voice Input (ASR) for Indian Languages:**

- **Overview:** Converts speech in Indian regional languages into text for voice-based form filling.
- **Applications:** Helps users who prefer speaking instead of typing; improves accessibility for low-literacy and multilingual users.

- **Recent works:** Indian-language ASR studies using models like Wav2Vec2.0, Whisper fine-tuning, and phonetic-aware architectures addressing low-resource and code-mixing challenges.
- **Literature Note:** Surveys highlighting Indo-Aryan/Dravidian language complexity and transformer-based approaches improving recognition accuracy.

## **Privacy, Security & On-Device Inference:**

- **Overview:** Ensures user data is handled securely and inference (AI processing) can occur locally (on-device) to reduce latency and data exposure.
- **Applications:** Citizen documents contain sensitive personally identifiable information (PII) like Aadhaar numbers and addresses — local inference or strong encryption ensures privacy compliance and trust.
- **Recent works:** Research on federated learning + on-device AI shows can run offline with acceptable accuracy; privacy-preserving document analytics frameworks emerging.
- **Literature Note:** Studies on secure document processing: encryption-aware pipelines, edge-AI document understanding, and UIDAI / Indian gov standards for PII handling in citizen services.

## CHAPTER 3

### ANALYSIS

#### 3.1 Software Requirement Specification (SRS):

**Functional Requirements:** Upload PDF/image documents, extract text and entities (name, DOB, address, etc.). Auto-fill form templates, support multilingual and voice inputs.

Enable review, edit, and download.

**Non-Functional Requirements:** Accuracy > 90% in entity extraction, Latency 3–5s per document, Platform-independent web application, Secure and privacy-preserving processing.

**Architecture Overview:** Document/Voice Input OCR/NLP Extraction Entity Mapping Form Template Filling Review/Download.

#### 3.2 Data Flow:

User Input (Documents / Voice)



Pre-processing (Image cleanup / Speech-to-Text)



OCR + Handwritten Text Recognition



Named Entity Recognition (NER) & Entity Extraction



Document Classification & Layout Parsing



Key-Value Extraction & Form Mapping



User Review & Manual Corrections



Final Output (Auto-Filled Form / Download / Submission)



### 4.1 System Model

Architecture consists of:

- Frontend (Web App):  
UI for document upload, voice input, and preview/edit form.
- Backend (AI Engine):  
Python-based pipeline performing OCR, entity extraction, and mapping.
- AI/ML Modules:  
Pretrained OCR, HTR, and NLP models integrated with custom scripts.
- Database / Storage:  
Temporarily stores extracted data and generated forms (can be local or cloud).
- Security Layer:  
Encrypts data, ensures privacy, supports on-device or secure local inference.

### 4.2 Module Organisation:

- User Input Module: Handles upload of PDFs/images and accepts voice input.
- Pre-Processing Module: Enhances image quality (denoising, binarization) and converts audio to text.
- OCR & Handwritten Text Recognition Module: Extracts printed and handwritten data from scanned documents.
- Named Entity Recognition (NER) & Entity Extraction Module: Detects entities such as Name, DOB, Address, ID Number.
- Document Classification & Layout Parsing Module: Identifies document type (Aadhaar, PAN, etc.) and its structure.
- Form Mapping & Key-Value Extraction Module: Maps extracted data to the correct form fields automatically.
- User Review & Validation Module: Displays preview for corrections before final submission.
- Output Generation Module: Exports or downloads the filled form (PDF/online submission).
- Privacy & Security Module: Ensures encryption, secure data handling, and optional on-device processing.

## CHAPTER 5

### IMPLEMENTATION & RESULTS

The project is implemented as a web-based platform using Streamlit / Flask. The backend integrates OCR libraries (Tesseract, Easy OCR) and NLP models for entity recognition. Data extracted from uploaded documents are parsed and matched to corresponding government form templates. The interface displays an editable preview, ensuring transparency before download.

#### 5.1 Implementation Details & Code Outputs

To achieve the functionality described, several key components were built:

**1. Multilingual OCR Extraction:** The system uses easy ocr(as mentioned in ) for its strong support of Indian languages. A wrapper function processes the uploaded image file.

```
# Sample Python code for OCR module
import easyocr
import io
from PIL import Image

# Initialize reader for English, Hindi, and Telugu
reader = easyocr.Reader(['en', 'hi', 'te'])

def extract_text_from_image(image_bytes):
    """
    Extracts text from an image file stream.
    """
    try:
        image = Image.open(io.BytesIO(image_bytes))
        # readtext returns a list of (bbox, text, probability)
        results = reader.readtext(image, detail=0, paragraph=True)

        # Combine extracted text paragraphs into a single string
        full_text = "\n".join(results)
        return full_text
    except Exception as e:
        print(f"Error during OCR: {e}")
        return None
```

**2.NER and Entity Mapping:** After OCR, a custom Named Entity Recognition (NER) model (or regex patterns for high-confidence fields like PAN/Aadhaar) extracts key-value pairs. This structured data is then mapped to form templates.

#### JSON

```
/* Sample JSON output from the OCR+NLP module after
   processing a sample Aadhaar Card image.
*/
{
  "doc_type": "Aadhaar",
  "extracted_entities": {
    "Name": "Rohan V. Sharma",
    "DOB": "15/08/1990",
    "Gender": "Male",
    "Aadhaar_Number": "XXXX XXXX 1234",
    "Address": "H.No 123, ABC Colony, XYZ City, State - 500001"
  }
}
```

## TESTING AND VALIDATION

#### JSON

```
/*
   Sample template definition for a "Domicile Certificate" form.
   The keys (e.g., "applicant_name") are the form's field IDs.
   The values are an ordered list of entity names to check from
   the extracted JSON.
*/
{
  "form_name": "Domicile Certificate Application",
  "field_map": {
    "applicant_name": ["Name", "Full Name"],
    "date_of_birth": ["DOB", "Date of Birth"],
    "full_address": ["Address", "Permanent Address"],
    "gender": ["Gender"],
    "uid_number": ["Aadhaar_Number"]
  }
}
```

Testing was performed using multiple sample documents such as Aadhaar cards, PAN cards, and voter IDs. Unit and functional testing ensured correctness of data extraction and mapping. Multilingual validation confirmed accuracy across major Indian languages. Voice input testing verified correct field recognition and update.

## 6.1 Sample Test Cases

Below representative test cases used to validate the system's core functionalities.

Test ID	Module	Test Case Description	Expected Result
UT-01	OCR Module	Unit Test: Provide a low-quality PAN card image.	The system should extract the PAN number and Name with >80% character accuracy.
UT-02	NLP Module	Unit Test: Input text: "Name: Priya Singh DOB 01/01/1995".	Extract entities () function should return {"Name": "Priya Singh", "DOB": "01/01/1995"}.
FT-01	Form Mapping	Functional Test: Upload an Aadhaar card and select the "Domicile Certificate" form template.	The 'applicant name' and 'full address' fields in the web preview must be auto-filled with the data from the Aadhaar card.
FT-02	Multilingual	Functional Test: Upload a Voter ID card with the address in Telugu.	The system should correctly extract the Telugu text and populate the 'Address' field in the review interface.
VT-01	Voice Input	Validation Test: User clicks the 'Edit' button for the 'Address' field and speaks "Set address to 456, Jubilee Hills, Hyderabad" in English.	The text in the 'Address' field must update to "456, Jubilee Hills, Hyderabad".
NFR-01	Performance	Non-Functional Test: Upload 5 different documents (PDF and JPG) consecutively.	The average processing time (from upload to review screen) must be 5 seconds per document.

## CHAPTER 7

### CONCLUSION & FUTURE ENHANCEMENTS

The AI-Powered Form Filling Assistant successfully automates document-based data extraction and form completion for Indian citizen services. It reduces manual effort, improves accessibility, and ensures accuracy. Future enhancements include handwritten text recognition, DigiLocker API integration for auto-document retrieval, and real-time government form submissions through API channels.

#### Abstract:

The AI-Powered Form Filling Assistant for Indian Citizen Services is designed to automate the manual process of completing government application forms by leveraging Artificial Intelligence. The system uses Optical Character Recognition (OCR) and Natural Language Processing (NLP) to extract key information such as name, date of birth, address, and identification numbers from uploaded documents like Aadhaar, PAN, or Voter ID. It then auto-fills these details into predefined templates, minimizing manual effort and human errors. The assistant supports multiple Indian languages and offers voice-based input for accessibility. This innovation enhances efficiency at Seva Kendras and contributes toward India's vision of digital and inclusive governance.

#### Sample Outputs (Text-Based Representation)

The system demonstrates the following output stages during operation:

- Extraction of text data from uploaded Aadhaar/PAN/Voter ID documents.
- Auto-filled form generation with editable fields.
- Voice-based form filling demonstration.
- Multilingual data mapping and field population.
- Final downloadable government form preview.

The sections below illustrate text-based representations of the system outputs, corresponding to the figures mentioned in the report.

**Fig 5.1 – Document Upload Interface** This is the web interface (built with Streamlit) where the user is presented with:

1. An "Upload Document" button (accepts .jpg, .png, .pdf).
2. A dropdown menu to select the target "Government Form" (e.g., "Domicile Certificate", "Birth Certificate Application").
3. A "Process" button.

**Fig 5.2 – Extracted Data Preview** After processing an uploaded Aadhaar card, the system first shows the raw extracted data (derived from the OCR+NLP module ) for user verification:

## JSON

```
// Data displayed in the 'Extracted Data' review tab
{
  "doc_type": "Aadhaar",
  "extracted_entities": {
    "Name": "Rohan V. Sharma",
    "DOB": "15/08/1990",
    "Gender": "Male",
    "Aadhaar_Number": "XXXX XXXX 1234",
    "Address": "H.No 123, ABC Colony, XYZ City, State - 500001"
  }
}
```

**Fig 5.3 – Auto-filled Form Template Preview** The system then displays the target government form's fields, auto-filled based on the mapping logic. The user can edit any field.

Form Field (Domicile)    Extracted Value (Editable)

<b>Applicant Name</b>	Rohan V. Sharma
<b>Date of Birth</b>	15/08/1990
<b>Gender</b>	Male
<b>Full Address</b>	H.No 123, ABC Colony, XYZ City, State - 500001
<b>Father's Name</b>	[Empty - Not found in Aadhaar]
<b>UID Number</b>	XXXX XXXX 1234

**Fig 5.4 – Voice-Based Input in Hindi** This output demonstrates the voice-filling capability for an empty field.

- **User Action:** Clicks the 'Microphone' icon next to the "Father's Name" field.
- **User Voice Input (Hindi):** "पिता का नाम विजय शर्मा"
- **System Transcription:** "Pita ka naam Vijay Sharma"
- **Action:** The "Father's Name" text field is automatically updated with the value "Vijay Sharma".

**Fig 5.5 – Final Downloadable Filled Form (PDF Output)** After the user clicks "Confirm and Download," the system generates a final PDF. This output is a data-filled PDF file, representing the official government form (e.g., Form-6 for Voter ID registration or a state-specific Domicile application) with the verified data populated in the correct coordinates.

