Here’s a **complete, step-by-step workflow** of your system, from the user uploading a document to the AI-powered classification and display, including **how JSON and the Python API fit in**. I’ll use plain language, code snippets, and show how each part connects.

**1. User Uploads a Document (or Adds a Legal Case)**

**File(s):**

* resources/views/document/create.blade.php
* resources/views/legal/create.blade.php

**What happens:**

* User fills out a form (title, description, uploads a file).
* User clicks **Submit**.

**2. Laravel Controller Receives the Request**

**File(s):**

* app/Http/Controllers/DocumentController.php
* app/Http/Controllers/LegalController.php

**Key code:**

php

Apply to show.blade.p...

public function store(Request $request)

{

    $request->validate([

        'title' => 'required|string|max:255',

        'description' => 'nullable|string',

        'document\_file' => 'required|file|mimes:pdf,doc,docx,txt|max:10240'

    ]);

    $file = $request->file('document\_file');

    // ... next steps

}

* **Variables:**
* $request (all form data)
* $file (the uploaded file)

**3. Laravel Saves the File and Extracts Text**

**Key code:**

php

Apply to show.blade.p...

$fileName = time() . '\_' . $file->getClientOriginalName();

$filePath = $file->storeAs('documents', $fileName, 'public');

$extension = strtolower($file->getClientOriginalExtension());

$documentText = '';

if ($extension === 'txt') {

    $documentText = file\_get\_contents($file->getRealPath());

} elseif ($extension === 'pdf') {

    $parser = new \Smalot\PdfParser\Parser();

    $pdf = $parser->parseFile($file->getRealPath());

    $documentText = $pdf->getText();

} elseif ($extension === 'docx') {

    // ... PHPWord code to extract text ...

}

* **Variables:**
* $documentText (plain text from the file)

**4. Laravel Sends Text to the Python API (via JSON)**

**Key code:**

php

Apply to show.blade.p...

$response = \Illuminate\Support\Facades\Http::post('http://127.0.0.1:5050/classify', [

    'text' => $documentText,

]);

$category = $response->json('category');

$entities = $response->json('entities');

* **What’s happening:**
* Laravel sends a **POST request** to the Python API at http://127.0.0.1:5050/classify.
* The text is sent as **JSON**:

json

Apply to show.blade.p...

    { "text": "The full document text here..." }

**5. Python API (spacy\_api.py) Receives and Processes the Request**

**File:**

* spacy\_api.py

**Key code:**

python

Apply to show.blade.p...

@app.route('/classify', methods=['POST'])

def classify():

    data = request.get\_json()

    text = data.get('text', '')

    doc = nlp(text)

    entities = [{"text": ent.text, "label": ent.label\_} for ent in doc.ents]

*# Rule-based classification*

    lowered = text.lower()

    if any(word in lowered for word in ["contract", "agreement", "party", "witnesseth"]):

        category = "Contract"

*# ... more rules ...*

    else:

        category = "Uncategorized"

    return jsonify({"entities": entities, "category": category})

* **What’s happening:**
* The API receives the JSON, extracts the text.
* **spaCy** analyzes the text for named entities (like people, dates, organizations).
* The code checks for keywords to classify the document type.
* The API returns a **JSON response**:

json

Apply to show.blade.p...

    {

      "entities": [

        {"text": "Juan Dela Cruz", "label": "PERSON"},

        {"text": "July 1, 2024", "label": "DATE"}

      ],

      "category": "Contract"

    }

**6. Laravel Receives the JSON Response**

**Key code:**

php

Apply to show.blade.p...

$category = $response->json('category'); // e.g., "Contract"

$entities = $response->json('entities'); // e.g., [{text: "...", label: "..."}]

* **What’s happening:**
* Laravel now knows the document’s **category** and the **entities** found by the AI.

**7. Laravel Saves the Category to the Database**

**Key code:**

php

Apply to show.blade.p...

$document = Document::create([

    'title' => $request->title,

    'description' => $request->description,

    'file\_path' => $filePath,

    'category' => $category, // from Python API

    // ...other fields

]);

* **Database:**
* The documents table has a category column (see your migration files).

**8. Laravel Passes Data to the Blade View**

**Key code:**

php

Apply to show.blade.p...

return view('document.show', [

    'document' => $document,

    'entities' => $entities,

]);

* **Variables:**
* $document (with category)
* $entities (from Python API)

**9. Blade View Displays the Results**

**File(s):**

* resources/views/document/show.blade.php
* resources/views/legal/show.blade.php

**Key code:**

blade

Apply to show.blade.p...

<div>

    <strong>Category:</strong> {{ $document->category ?? 'Uncategorized' }}

</div>

@if(!empty($entities))

    <div>

        <strong>Named Entities:</strong>

        <ul>

            @foreach($entities as $entity)

                <li>{{ $entity['text'] }} ({{ $entity['label'] }})</li>

            @endforeach

        </ul>

    </div>

@endif

* **What’s happening:**
* The user sees the **AI-predicted category** and a list of **entities** found in the document.

**10. User Sees the Results in the UI**

* The document is now **auto-categorized** and **AI-analyzed**.
* The user can see the document’s type and important entities (like people, dates, organizations) right in the web app.

