Agentic AI in public administration

Goal

- Build an agentic assistant for Romanian e-gov scenarios
- Demo: Support at least 2 flows: **Carte de identitate (CI)** and **Ajutor social**
- Keep human-in-the-loop and reuse the same agents, chat widgets for a unified interface.
- Reuse existing national APIs, systems – just add LLMs and agents on top.

Agentic architecture: A2A (Agent-To-Agent)

- **Entry / Intent Agent** Human/Operator prompts → talks to LLM → decides domain, which agent to call next.

- **Domain agents**:
  - CI Agent
  - Social Aid Agent
  - Other use-cases outside the demo ..

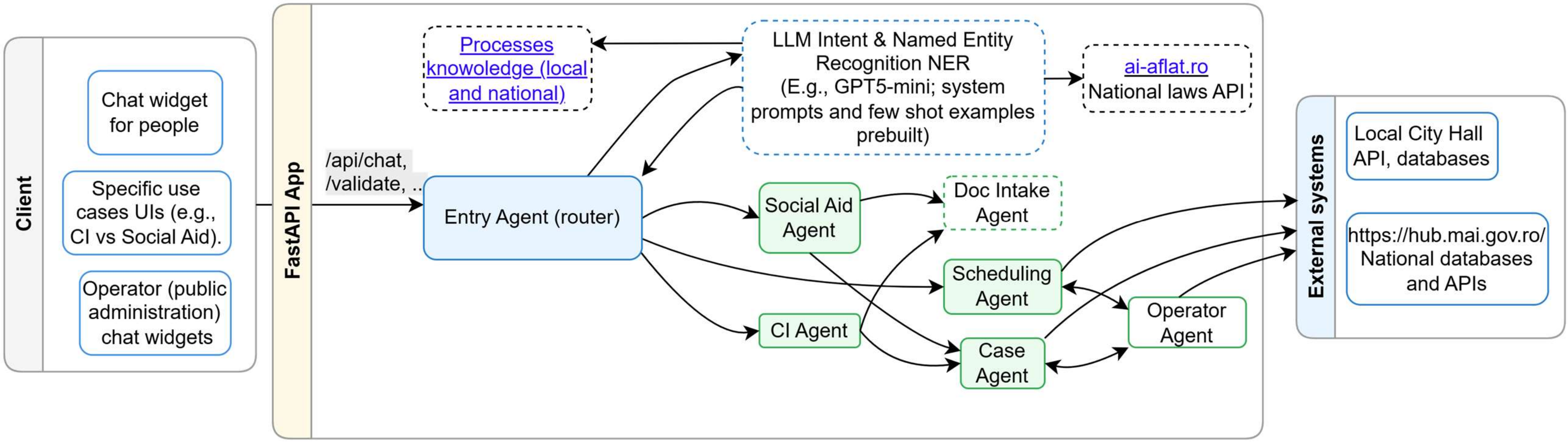
- **Shared service agents**:
  - Doc Intake Agent: OCR → canonical docs
  - Scheduling Agent
  - Case Agent
  - Operator Agent (the public administration operator)
  - Knowledge Agents

- The UI works as follows: it gets a sequence of steps to update panels in real time from the Agentic AI system.

Agents Overview

- **Entry Agent**
  - calls LLM to classify Romanian user message
  - extracts entities (CNP, slot_id, email)
  - routes to the right agent
- **CI Agent**
  - Handles the “Carte de identitate” case decides the eligibility, etc.
  - Handles requirements, missing documents, with user in the loop.
  - Speaks with the Scheduling Agent for finding a schedule.
- **Social Aid Agent**
  - Handles the Social Aid cases
  - checks social-specific documents, if all the eligibility conditions are met.
  - uses local regulations of the city where the user lives.
- **Knowledge Agents**
  - ai-aflat.ro – API to query national laws knowledge
  - Local city hall / government /etc resources by API or static documents handled by RAG
- **Doc Intake Agent**
  - reads uploaded documents through discussion with the users (both people and gov operators).
  - normalizes OCR labels with LLM and tries to create *canonical* documents
  - updates application statues after uploading by speaking with CI Agent/Social Aid Agent.
  - Respond to the users if clarifications must be made.
- **Case Agent**
  - Handles statuses of cases opened
  - Notify users for completion or clarifications needed
  - Handles payment operations
- **Scheduling Agent**
  - for CI it will use the government hub → https://hub.mai.gov.ro/
  - for social → local city hall slots, databases.
  - same agent, two backends even for the demo.
- **Operator Agent**
  - **This is the human operator on the public administration side.**
  - Handle cases and statuses through natural prompts, e.g., list/claim/complete tasks
  - Handles communications back to people for clarifications.

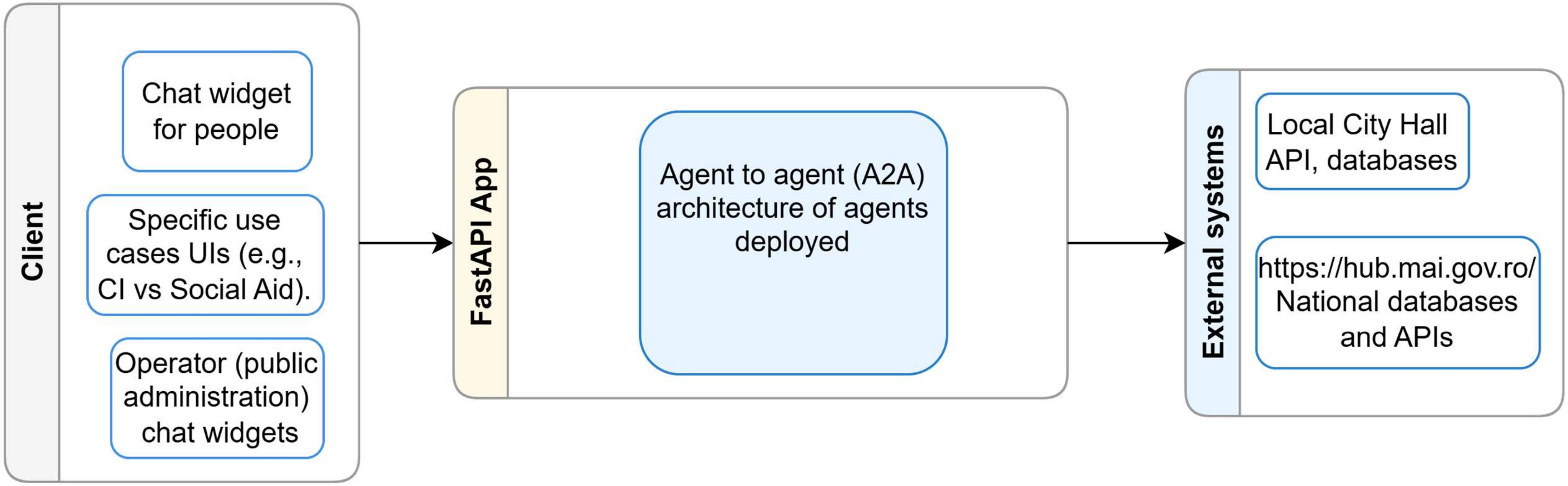
Agentic architecture: A2A (Agent-To-Agent)


Client
Chat widget for people
Specific use cases UIs (e.g., CI vs Social Aid).
Operator (public administration) chat widgets
FastAPI App
/api/chat, /validate, ..
Entry Agent (router)
Processes knowoledge (local and national)
LLM Intent & Named Entity Recognition NER (E.g., GPT5-mini; system prompts and few shot examples prebuilt)
ai-aflat.ro National laws API
Social Aid Agent
Doc Intake Agent
Scheduling Agent
CI Agent
Case Agent
Operator Agent
External systems
Local City Hall API, databases
https://hub.mai.gov.ro/ National databases and APIs

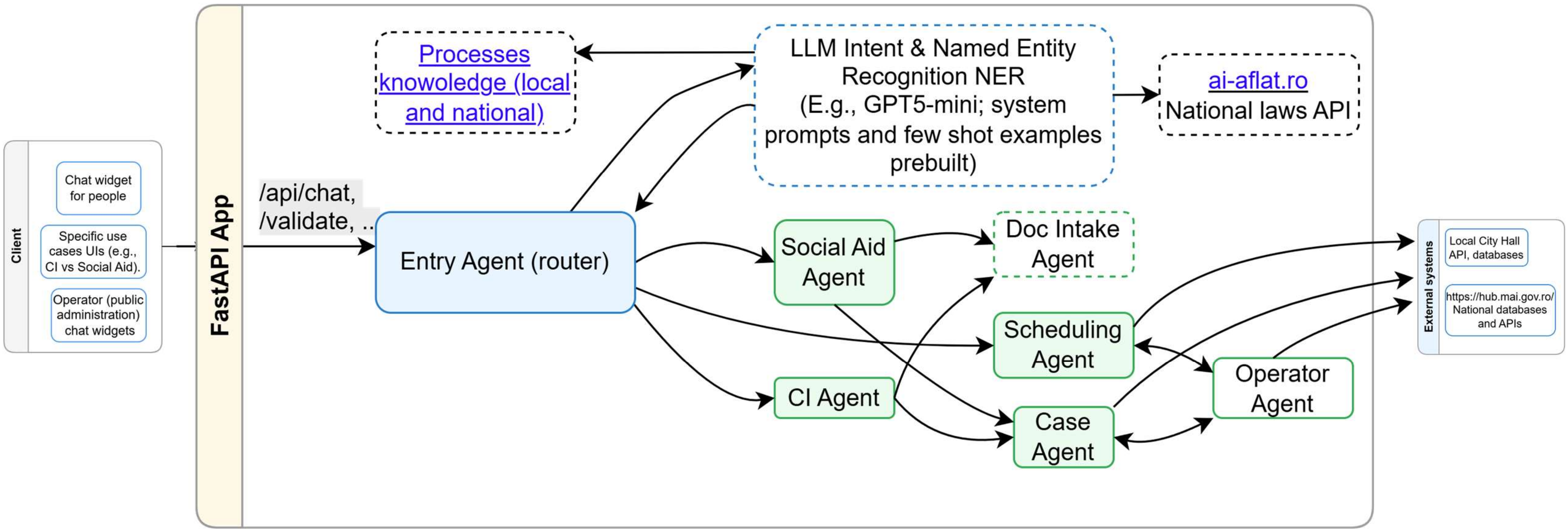
Notes:

- The dashed entities are reused in the current prototype. The idea is to share as much as possible the systems, including the externally deployed ones (on the right side)

OR.. Split in two : first speak as general systems


Client
Chat widget for people
Specific use cases UIs (e.g., CI vs Social Aid).
Operator (public administration) chat widgets
FastAPI App
Agent to agent (A2A) architecture of agents deployed
External systems
Local City Hall API, databases
https://hub.mai.gov.ro/ National databases and APIs

Then… present better the middle


Client
Chat widget for people
Specific use cases UIs (e.g., CI vs Social Aid).
Operator (public administration) chat widgets
FastAPI App
/api/chat, /validate, ..
Entry Agent (router)
Processes knowoledge (local and national)
LLM Intent & Named Entity Recognition NER (E.g., GPT5-mini; system prompts and few shot examples prebuilt)
ai-aflat.ro National laws API
Social Aid Agent
Doc Intake Agent
Scheduling Agent
CI Agent
Case Agent
Operator Agent
External systems
Local City Hall API, databases
https://hub.mai.gov.ro/ National databases and APIs

Motivation for LLMs: Regex/NLP vs LLM (per agent)

- Entry Agent
  - regex fails on: “am nevoie de ajutorul ăla de la primărie”
  - LLM: understands it’s social → intent: "social”
- CI Agent
  - regex fails on OCR labels like “certificat copil”, “extras CF”
  - LLM: maps to the correct labels: cert_nastere, dovada_adresa
- Social Aid Agent
  - regex fails on “venitul minim… ăla pentru cei cu probleme”
  - LLM: maps to social domain
- Scheduling Agent
  - regex/classic NLP expects “slot id/day/hour” something very clear
  - LLM can understand “ia primul de miercuri” and let the agent pick index 0 available.
- Operator Agent
  - regex /classic NLP expects “claim task 10”, where “10” is the ID, or something clear.
  - LLM can understand “preia tu tasku’ 10 pe numele meu”

- Check a full suite of examples by double-click here: LLMVsNLPCompare.html

Implementation status

https://github.com/unibuc-cs/AI-for-public-administration.git

- **Completed**
  - ✅ FastAPI app with 3 main UIs: user CI, user Social, operator
  - ✅ Local mock for Primărie and CEI-hub-like endpoints
  - ✅ Chat widget that accepts steps (toasts, slots, cases, tasks)
  - ✅ Agents already factored logically in code (entry, domain, operator)

- **To do**
  - 🔄 Swap regex intents with LLM intent + entity extraction
  - 🔄 Finalize social checklist JSON
  - 🔄 Move doc-intake normalization to LLM
  - 🔄 Add per-domain DB wiring (CI vs Social)
  - 🔄 Polish operator empty states (done)