Water monitoring: Prototype deadline prep

Meghan Carr

Prototype Scope

Overall Goal

- Demonstrate an end-to-end flow: Image upload → AI assistive suggestion → structured output saved for analysis
- Prove technical feasibility of AI-assisted wetland observation without requiring user expertise
- Produce a reviewable artifact that shows valid system design for MVP - not full feature completeness

---

Interpreting the 3 Guidelines

1) Backend only ✅ The backend handles file input, API call, response parsing.

- Focus on API logic, data handling, and AI interaction; UI can be minimal or simulated
- Prioritize correctness, clarity, and logging over polish
- Enables fast iteration and avoids frontend bottlenecks

2) Picture file upload in VS Code ✅You read a local image file and send it via the API.

- Local file upload simulates future web uploads without building auth or storage
- Keeps scope contained: single image, predictable format, repeatable tests
- Allows deterministic debugging of AI responses

3) AI response with ID

- AI returns *suggested biological labels*, not verified classifications
- Output must be structured (JSON), not free text
- Response is explicitly assistive, not authoritative

AI Identification Design

AI should suggest 2-3 species/genus members

Why:

- Single ID implies false certainty (especially with non-expert users)
- 2-3 options reflect how real ecological review works
- Supports your *human-in-the-loop* and *responsible AI* framing

Best practice for a prototype:

- Primary suggestion (most likely)
- 1-2 alternative possibilities
- Each with a confidence estimate or qualitative ranking

Even better (if time allows):

- Species *if confident*, otherwise Genus-level
- Explicit fallback: *"Unable to confidently identify. Do you want ideas for better photos?"*

Using LLaMA Vision — Where the API Fits

You are not training a model. You are calling one.

Here's the clean mental model.

Architecture (Prototype-Appropriate)

[Local Image File]
↓
[Backend Script / Service]
↓
[LLaMA Vision API Call]
↓
[Structured AI Response (JSON)]
↓
[Console Output / Saved Record]

API's Role

➡️ Transmits the image + instructions and returns structured suggestions; no storage, no training, no state.

- **The API is how your backend sends:**
  - **The image**
  - **A carefully written prompt (instructions + constraints)**
- **And receives:**
  - **The AI's structured response (IDs + confidence)**

You are using the API as:

> **"A remote, pre-trained perception service."**

Example: What the AI Response Should Look Like (Conceptually)

```
{
  "suggestions": [
    {
      "label": "Red-eared slider (Trachemys scripta elegans)",
      "rank": 1,
      "confidence": 0.62
    },
    {
      "label": "Painted turtle (Chrysemys picta)",
      "rank": 2,
      "confidence": 0.24
    }
  ],
  "disclaimer": "Suggestions only. Not a verified identification."
}
```

Coach's Advice (Straight Talk)

With 8 days:

- Do not chase UI
- Do not chase accuracy perfection
- Do not oversell automation

What *will* impress:

- Clean flow
- Clear AI boundaries
- Honest uncertainty
- Structured outputs

If you want, next I can:

- Draft the exact LLaMA Vision prompt
- Sketch the backend pseudocode
- Help you write the "Prototype Limitations" section (very powerful for reviewers)