⭐ ELIAS – Day 11 (Friday) Official Work Plan

(8 PM – 10 PM Work Session)

Theme: Vision Phase 2 + UI Connection + Middle-Layer Integration

We are now entering the *second phase* of the project where pieces begin joining together:

- Vision → Authentication
- NLP → More intents
- Plugins → More variety
- UI → Starts connecting to backend

🧩 Team Roles for the Day

Member	Role	Focus Today
M S	Lead & Integrator	Connect UI to backend + routing cleanup
S S	NLP	Add classifiers + error fallback behavior
A K	Vision Engineer	Add face landmarks + authentication scaffolding
R A	Plugin Developer	Add second utility plugin
M A	Documentation	Architecture diagram + Day 11 log

🟦 1. M S — Lead & Integrator

Tasks

✅ Connect Streamlit UI → parse_intent() directly
✅ Create simple function API:

backend.parse(user_text) → returns intent + result

✅ Add a middle-layer:

src/controller.py

Purpose:

- Accept input
- Run NLP

- Route to plugins
- Return formatted output
- Add explainability logs

✅ Clean routing logic in app.py
✅ Support "response object" that the UI can show cleanly

Expected Output

- Streamlit UI takes user input → backend → output displayed
- Controller layer exists
- app.py becomes cleaner and modular

🟩 2. S S — NLP Developer

Tasks

✅ Add *fallback intent*:

if user says something unknown → detect sentiment / classify query type

✅ Implement basic classification using keyword groups:

- greeting
- thanks
- help
- confusion
- fallback search

✅ Enhance parse_intent() to return:

```
{
 "intent": "...",
 "entities": {...},
 "confidence": 0.6–1.0
}
```

✅ Add 2 more intents:

- system_status
- quick_math (extract numbers)

Expected Output

- NLP outputs *confidence score*
- NLP handles greetings, help, fallback
- Quick math now extractable

🟪 3. A K — Vision Engineer

Tasks

✅ Add face-landmark detection using Mediapipe
✅ Enhance detect_face() to return:

{"face_detected": True, "landmarks": <points>}

✅ Create authentication scaffold (not full logic):

src/vision_auth.py

Contains:

- function to capture face features
- function to compare features (dummy comparison for now)

✅ Test capturing your own face encoding and save as:

data/face_model/user1.json

Expected Output

- Landmark detection works
- Basic face signature extracted
- Authentication module structure created

🟧 4. R A — Plugin Developer

Tasks

✅ Create **second major utility plugin**
Choose one:

1. **System Info Plugin**
    - CPU, RAM, battery, OS
2. **Reminder Plugin**
    - Save reminders to file
3. **Music Player Plugin**
    - Play/pause local MP3
4. **Calculator Plugin**
    - Evaluate arithmetic

✅ Follow optimized structure:

```
def run(entities):
  return {ok:..., message:..., data:...}
```

Expected Output

- New plugin functional
- Clean modular code

🟨 **5. M A — Documentation & QA**

Tasks

✅ Create updated architecture diagram for:

- NLP
- Vision
- Plugins
- Controller
- UI

✅ Write **Day 11 log**
✅ Add "Vision Authentication — PART 2" doc
✅ Create checklist for final integration

Expected Output

- Clear system diagram
- Documented flow for every module
- All Day 11 work logged

🎯 **End-of-Day Expected Outcome**

By end of Day 11:

✅ Streamlit UI → backend connection live
✅ NLP improved with fallback + confidence
✅ Vision phase 2 working (landmarks + embedding)
✅ New plugin created
✅ Updated architecture documentation

System becomes **multi-layered and real**.