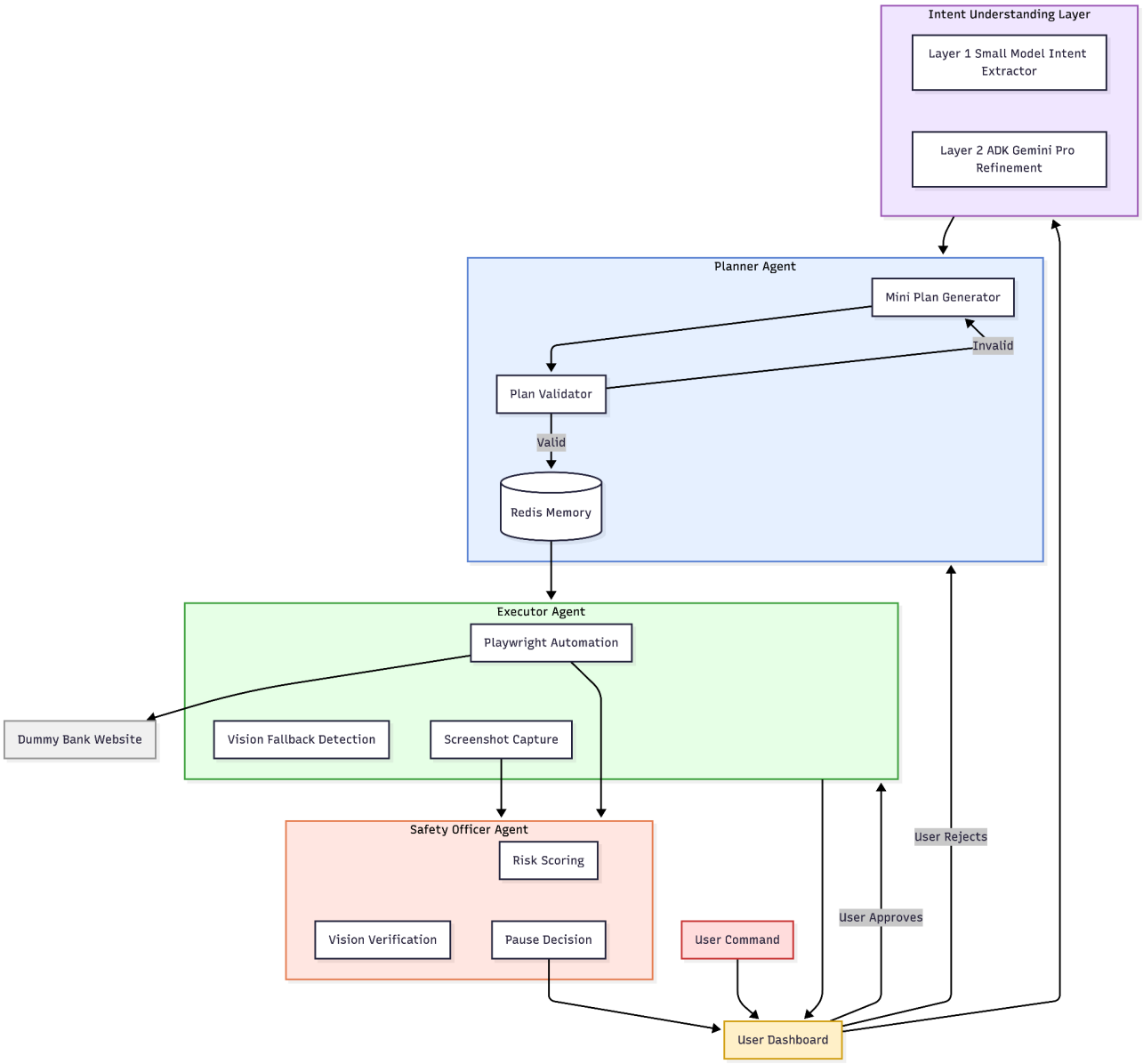
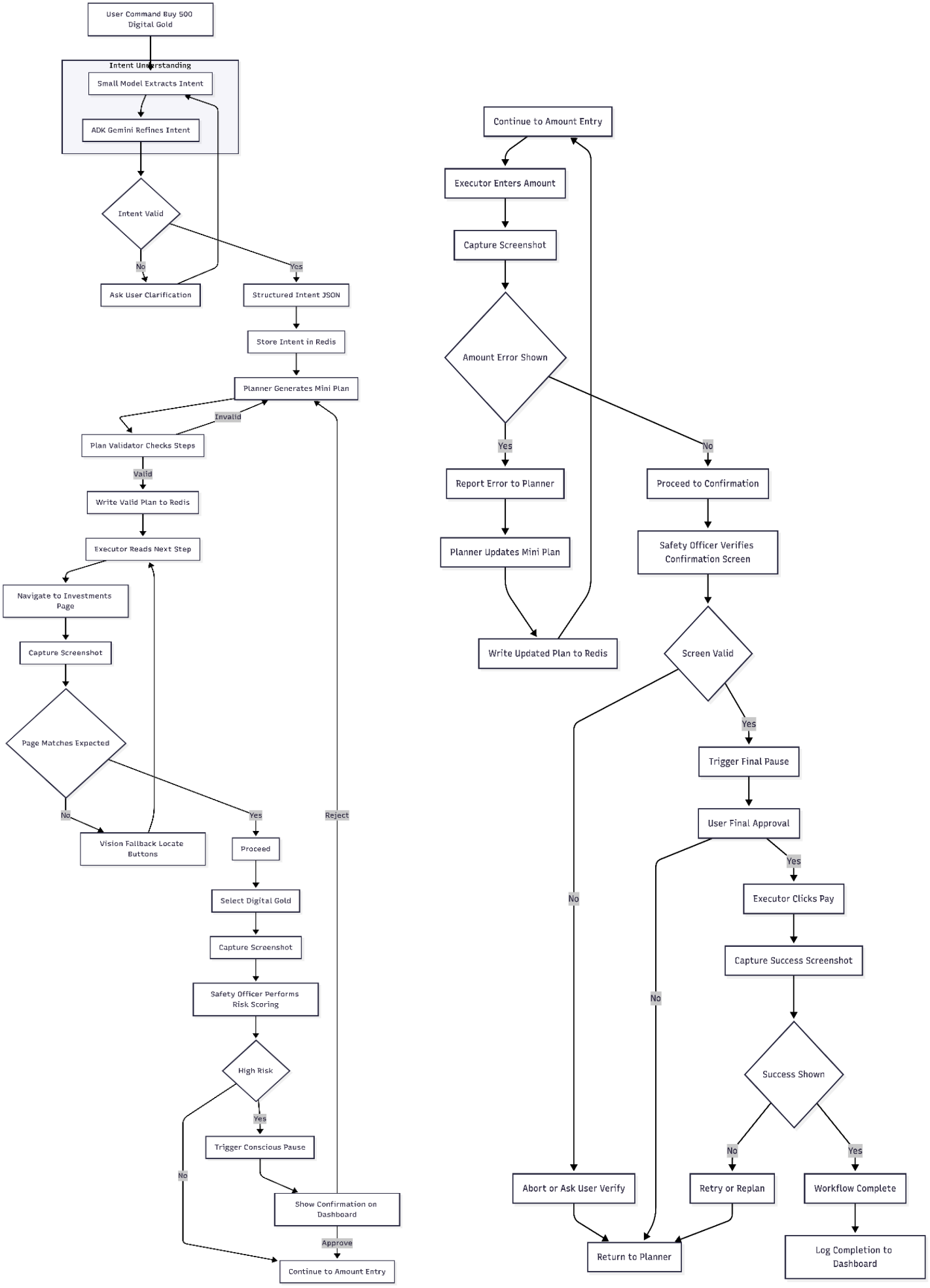
Team ID: Fina-250478 **Name**: Advita Shrivastava **Contact**: +918052407029, advitashrivastava09@gmail.com

Architecture Diagram: TriGuard Intel



Workflow Strategy

To illustrate the system end-to-end, consider the user command “Buy 500 digital gold.” The **Intent Understanding Layer** first performs a two-stage parse where a small model extracts the action parameters and Google ADK with Gemini Pro validates and structures them. The **Planner Agent** then generates a mini-plan detailing each required step- navigating to the Investments page, selecting Digital Gold, entering the amount, preparing confirmation, and the **Plan Validator** ensures every action is safe, executable, and schema-conformant before writing it to Redis. The **Executor Agent** retrieves this plan and performs UI actions through Playwright, using Vision fallback when DOM selectors fail. At each critical stage, especially before amount entry and final payment, the **Safety Officer** monitors the live UI, performs risk scoring, and validates confirmation screens using both heuristics and Vision models. If any step is deemed high-risk, ambiguous, or inconsistent with expected UI patterns, the system activates a Conscious Pause, surfaced through the dashboard, requiring explicit user approval or modification. Once approved, execution resumes deterministically from the stored Redis state. Core USPs of the architecture include a multi-agent design with strict plan validation, vision-grounded UI verification, and an independent Safety Officer enforcing financial-grade guardrails. Below is a detailed flowchart (left flowchart top to bottom → ight flowchart)


User Command Buy 500 Digital Gold
Intent Understanding
Small Model Extracts Intent
ADK Gemini Refines Intent
Intent Valid
No
Yes
Ask User Clarification
Structured Intent JSON
Store Intent in Redis
Planner Generates Mini Plan
Invalid
Plan Validator Checks Steps
Valid
Write Valid Plan to Redis
Executor Reads Next Step
Navigate to Investments Page
Capture Screenshot
Page Matches Expected
No
Yes
Vision Fallback Locate Buttons
Proceed
Select Digital Gold
Capture Screenshot
Safety Officer Performs Risk Scoring
High Risk
Yes
Trigger Conscious Pause
No
Show Confirmation on Dashboard
Reject
Approve
Continue to Amount Entry
Continue to Amount Entry
Executor Enters Amount
Capture Screenshot
Amount Error Shown
Yes
No
Report Error to Planner
Planner Updates Mini Plan
Write Updated Plan to Redis
Proceed to Confirmation
Safety Officer Verifies Confirmation Screen
Screen Valid
Yes
No
Trigger Final Pause
User Final Approval
Yes
No
Executor Clicks Pay
Capture Success Screenshot
Success Shown
No
Yes
Retry or Replan
Workflow Complete
Abort or Ask User Verify
Return to Planner
Log Completion to Dashboard

“Conscious Pause” Mechanism

The Conscious Pause mechanism functions as a financial-grade circuit breaker. It detects high-risk or irreversible actions before execution, halts automation, and requires explicit user approval through the dashboard. No sensitive action proceeds without user confirmation.

I. Detecting High-Stakes Actions

High-risk steps are identified through a three-layer pipeline:

Layer 1 → Planner Risk Classification

When generating a mini-plan, the Planner assigns each step a risk level (e.g., Transfer, Pay, Buy, Invest = High Risk). High-risk steps are tagged with `requires_pause = true`.

Layer 2 → Safety Officer Real-Time Evaluation

During execution, the Safety Officer inspects the live webpage, call-to-action buttons, UI anomalies, transaction amount, beneficiary novelty, and user context. Any elevated risk immediately triggers a pause.

Layer 3 → Screenshot Verification (Anti-Spoof)

Before irreversible clicks, the system validates the confirmation screen using vision models. If the displayed amount, entity, or layout does not match expected patterns, the action is blocked and the user is alerted.

II. Stop-and-Confirm UI

When paused, the dashboard displays a confirmation modal showing the action summary, amount, entity, risk level, and a screenshot of the page being acted upon.
Users can:

- **Approve** – resume execution,
- **Reject** – return control to the Planner,
- **Modify** – provide corrected inputs.

Safeguards include delayed approval buttons, optional explanations, and color-coded risk indicators.

III. Key Strengths

- Mirrors real financial approval workflows
- **Independent** safety agent prevents LLM hallucination-driven actions
- Multi-layer gating eliminates single points of failure
- Vision-based authenticity checks mitigate phishing and UI manipulation
- Fully auditable with logs and screenshots for every pause event

IV. Potential Extensions (If Time Permits)

A. Behavioral Risk Profiling: The system learns user patterns (typical transaction amounts, timing, beneficiary frequency). Deviations trigger stricter pause conditions or multi-factor verification.

B. Transaction Simulation Mode: Before approval, the agent generates a predicted "success screen," showing fees, balance impact, and computed totals, helping users validate intent.

C. UI Difference Hashing for Anti-Phishing: Screenshots are hashed and compared against known-safe templates to detect manipulated or injected UI components.

D. Two-Factor Conscious Pause (2F-CP): High-value or unusual transactions require a second confirmation channel (OTP or secondary prompt in demo mode).

Technology Stack Selection

1. Intent Understanding Layer

Tools: Layer 1: Llama 3 / Gemini Flash (small model), Layer 2: Google ADK + Gemini Pro

Why:

- Two-stage parsing reduces inference cost by ~60–70%.
- Small model handles fast, cheap extraction; large model only used when required.
- ADK enforces strict schemas → prevents malformed intents and reduces hallucinations.
- Low latency and high accuracy for intent classification.

2. Planning Layer

Tools: Google ADK (Planner Agent) + Gemini Pro

Why:

- ADK supports deterministic multi-step planning with tool graphs.
- Built-in output validation ensures high reliability in financial workflows.
- Generates structured mini-plans → fewer execution errors.
- Safer and more consistent than prompt-based planning.

3. Execution Layer

Tool: Microsoft Playwright

Why:

- Faster and more stable than Selenium (auto-waiting, async architecture).
- Robust under SPA, dynamic DOM, popups, and latency variations.
- Fewer flaky selectors → lower maintenance burden.
- Production-grade speed and reliability for UI automation.

4. Vision Layer

Tools: Gemini Pro Vision / GPT-4o Vision + OpenCV

Why:

- Vision fallback identifies UI elements when DOM fails.
- Critical for dynamic UIs and LAM-style autonomy.
- Authenticity checks for confirmation screens → prevents spoofing/phishing.
- OpenCV diffing adds a low-cost, deterministic secondary check.

5. Memory Layer

Tool: Redis

Why:

- Microsecond read/write → ideal for real-time agent state.
- Supports multi-agent coordination (Planner, Executor, Safety Officer).
- Enables workflow resumption after crashes or pauses.
- Ensures deterministic state transitions.

6. Safety Officer Layer

Tools: Small LLM (risk rules) + Vision model (screen validation) + Redis context

Why:

- Independent safety evaluation prevents unsafe LLM-driven actions.
- Flags high-risk transactions using rules + heuristics.
- Vision verification ensures the confirmation screen is authentic.
- Provides enterprise-level gating before irreversible actions.

7. User Dashboard

Tools: React + WebSockets

8. Dummy Banking Website

Tools: HTML / CSS / JavaScript