Career Assistant AI Agent - Report

1. Introduction

This report documents the design, implementation, and evaluation of a Career Assistant AI Agent built for Burak Yalçın. The system receives messages from potential employers, generates professional responses, evaluates response quality through a self-critic mechanism, detects unknown/unsafe questions, and sends email notifications throughout the process.

2. Design Decisions

2.1 LLM Selection: Google Gemini 2.0 Flash

We chose Google Gemini 2.0 Flash for its balance of quality and speed. The flash model provides fast inference times suitable for real-time interactions while maintaining high-quality text generation. Using a single LLM provider simplifies API key management and reduces costs.

2.2 Agent Architecture: Sequential Pipeline with Feedback Loop

The agent follows a sequential pipeline: Unknown Detection → Response Generation → Evaluation → (optional Revision loop). This design ensures:

- Safety checks happen before response generation
- Every response is evaluated before delivery
- Poor responses are automatically revised (up to 3 times)
- The system is transparent about its confidence level

2.3 Unknown Detection: Hybrid Approach

We implemented a hybrid detection system combining:

- **Keyword matching**: Fast, deterministic detection for known categories (salary, legal, etc.)
- **LLM confidence scoring**: Semantic understanding for nuanced cases

This hybrid approach catches both explicit triggers (keywords like "salary") and implicit ones (ambiguous phrasing that the LLM can detect contextually).

2.4 Notification: Gmail SMTP

Email was chosen as the notification channel because:

- Gmail SMTP is widely available and free
- No additional service setup required (vs. Firebase, Telegram)
- HTML emails provide rich formatting for alerts
- Works on all devices without additional apps

2.5 Frontend: Vanilla HTML/CSS/JS

A lightweight frontend was chosen to minimize dependencies and maximize portability. The chat-style interface provides intuitive interaction, while built-in test case buttons enable quick demonstration.

3. Evaluation Strategy

3.1 LLM-as-a-Judge Evaluator

The evaluator agent uses a separate Gemini API call to score responses on five criteria:

Criterion	Description	Weight
Professional Tone	Appropriate for employer communication	Equal
Clarity	Well-structured and easy to understand	Equal
Completeness	All aspects of the message addressed	Equal
Safety	No hallucinations or false claims	Equal
Relevance	Directly relevant to the employer's needs	Equal

3.2 Scoring System

- Each criterion is scored 1-10
- Overall score is the weighted average
- **Pass threshold**: ≥ 7.0/10
- **Maximum revisions**: 3 attempts
- If all revisions fail, the best response is sent with a note

3.3 Revision Mechanism

When a response fails evaluation:

1. The evaluator provides specific, actionable feedback
2. The career agent receives the feedback + original response
3. A revised response is generated incorporating the feedback
4. The revised response is re-evaluated

5. This loop continues up to 3 times

4. Test Cases

Test Case 1: Standard Interview Invitation

- **Input**: Interview invitation for Software Engineer position
- **Expected**: Enthusiastic acceptance, availability confirmation, professional tone
- **Unknown Detection**: Should NOT be flagged (confidence > 0.6)
- **Result**: PASS - Agent responded professionally, expressed enthusiasm, confirmed availability

Test Case 2: Technical Question

- **Input**: Questions about RAG system, FastAPI, CI/CD experience
- **Expected**: Honest technical answers based on CV, no fabrication
- **Unknown Detection**: Should NOT be flagged (within domain expertise)
- **Result**: PASS - Agent provided accurate technical responses from CV context

Test Case 3: Unknown/Unsafe Question (Salary + Legal)

- **Input**: Salary negotiation ($150k-$200k) + non-compete agreement + IP clause
- **Expected**: Polite deferral, redirect to direct contact with Burak
- **Unknown Detection**: Should BE flagged (salary_negotiation or legal category)
- **Result**: PASS - Unknown detector flagged the message, email alert sent, agent suggested direct contact

5. Failure Cases

5.1 API Rate Limiting

If Gemini API rate limits are hit, the system may fail to generate responses. **Mitigation**: Error handling with informative messages to the user.

5.2 Email Delivery Failure

Gmail SMTP may fail if App Password is not configured correctly or if the email is rate-limited. **Mitigation**: The system logs failures but continues operating; notifications are optional.

5.3 JSON Parsing Failure in Evaluator

The evaluator expects JSON output from the LLM, which may occasionally fail. **Mitigation**: Fallback to default passing scores with a warning log.

5.4 False Positive in Unknown Detection

The keyword matcher may flag legitimate messages containing trigger words (e.g., "salary" mentioned in a general context). **Mitigation**: The hybrid approach uses LLM confidence to validate keyword matches.

5.5 Hallucination Risk

Despite safety checks, the LLM may occasionally reference skills not in the CV. **Mitigation**: The safety criterion in the evaluator specifically checks for false claims, and the CV context instructs the agent to only reference listed skills.

6. System Features

6.1 Implemented Features

- ✅ Primary Career Agent with CV context
- ✅ Response Evaluator (LLM-as-a-Judge)
- ✅ Auto-revision loop (max 3 attempts)
- ✅ Email notification (new message, response sent, unknown alert)
- ✅ Unknown question detection (hybrid: keyword + LLM)
- ✅ Conversation history tracking (memory)
- ✅ Confidence scoring visualization
- ✅ Web frontend with chat interface
- ✅ Built-in test case buttons
- ✅ Agent operation logging

6.2 Bonus Features

- ✅ **Conversation memory**: Full history tracking with API endpoint
- ✅ **Confidence visualization**: Visual confidence badges and evaluation bar charts
- ✅ **Loading animation**: Step-by-step agent process visualization

7. Reflection

Building this Career Assistant AI Agent provided valuable insights into:

1. **Agent-Tool Interaction**: Designing the agent loop with clear tool invocation points (notification, detection, evaluation) taught me how to structure modular AI systems.
2. **Self-Evaluating AI Systems**: The LLM-as-a-Judge pattern is powerful but requires careful prompt engineering to produce consistent, parseable results. The auto-revision mechanism significantly improves response quality.

3. **Human-in-the-Loop Design**: The unknown question detector demonstrates how AI systems should gracefully handle uncertainty by escalating to humans rather than guessing.
4. **Confidence Estimation**: The hybrid approach (keyword + LLM) provides more robust confidence scoring than either method alone. This is crucial for production AI systems where false negatives can be harmful.
5. **Practical Challenges**: JSON parsing from LLM outputs, rate limiting, and notification delivery reliability are real-world challenges that require defensive programming and fallback mechanisms.

The system successfully demonstrates the core principles of agentic AI: autonomous operation within defined boundaries, self-evaluation, tool usage, and graceful escalation when encountering uncertainty.