Vertex AI Agent System Instructions

To complete your system, you need to configure the **Gemini Enterprise agent** to act as the intelligent interface. This involves setting **System Instructions** that define its persona, its knowledge of the ADDIE model, and how it should retrieve data from BigQuery.

Vertex AI Agent System Instructions

Copy and paste the following into the **System Instructions** (or "Preamble") field of your Vertex AI Agent:

> **Persona:** You are a senior **Instructional Design Assistant** powered by Gemini Enterprise. Your goal is to help users navigate the ADDIE (Analysis, Design, Development, Implementation, Evaluation) process by providing high-quality, approved prompts from our internal database.
>
> **Knowledge Base:** > * You are grounded in the `prompts_master` table within BigQuery.
>
> - You only provide prompts that are marked as `is_active = TRUE`.
>
> **Core Tasks:**
>
> 1. **Retrieve:** When a user asks for a prompt (e.g., "Give me a Design-phase prompt for learning objectives"), query the data store to find the most relevant, highest-rated match.
> 2. **Explain:** Briefly explain why the returned prompt is effective for that specific ADDIE phase.
> 3. **Refine:** If no exact match is found, use your internal reasoning to suggest a draft based on the user's context, but clearly state it is a "Generated Suggestion" and not an "Approved Master Prompt."
>
> **Constraints:**
>
> - Always cite the `prompt_id` and the `author` when returning a result.
> - If a user asks about prompt performance, refer to the `avg_rating` and `usage_count` metadata.
> - Maintain a formal, professional, and helpful tone.

Connecting the Agent to BigQuery

To make these instructions functional, you must link the Agent to your data:

1. **Data Store Creation:** In Vertex AI Agent Builder, create a **Structured Data Store** pointing to your BigQuery `prompts_master` table.
2. **Enable Grounding:** In the Agent's "Tools" or "Settings" pane, toggle on **Grounding** and select your new BigQuery data store.
3. **Permissions:** Ensure the Service Account for your Agent has the `BigQuery Data Viewer` role to access the tables.

Recommended Next Step

Would you like me to create a **Human-in-the-Loop Feedback Form** (using Google Apps Script) that allows users to rate these prompts directly from the Gemini chat interface?

Yes

To implement the **Human-in-the-Loop (HITL)** feedback mechanism for your Prompt Management System, you can use **Google Apps Script** as a bridge between your users and the BigQuery database. This setup allows users to provide ratings and corrections directly, which can then be used to continuously refine your AI models.

HITL Feedback Workflow

1. **Form Submission:** End-users submit feedback via a Google Form or a custom web app interface built with Apps Script.
2. **Processing:** An Apps Script trigger automatically captures the submission.
3. **Storage:** The script pushes the feedback, including a rating (1–5) and optional text, into your `usage_analytics` BigQuery table.
4. **Review Loop:** Administrators review low-rated prompts in the `prompts_master` table to make necessary improvements.

Implementation: Apps Script for BigQuery Feedback

You can deploy the following code within a **Google Apps Script** project. This script uses the **Advanced BigQuery Service** to insert user feedback directly into your analytics table.

JavaScript

```javascript
/**
 * Pushes human feedback from a form submission into BigQuery.
 * Requires: BigQuery API enabled in Google Cloud and Apps Script.
 */
function recordFeedback(e) {
  const projectId = 'your-project-id';
  const datasetId = 'prompt_mgmt';
  const tableId = 'usage_analytics';

  // Extract data from form submission event (e)
  const responses = e.values;
  const feedbackData = {
    usage_id: Utilities.getUuid(),
    prompt_id: responses[1], // Assumes Prompt ID is in the second column
    user_email: Session.getActiveUser().getEmail(),
    timestamp: new Date().toISOString(),
    feedback_rating: parseInt(responses[2]), // 1-5 rating
    feedback_text: responses[3], // Qualitative comments
    generation_successful: true
  };

  const request = {
    rows: [{json: feedbackData}]
  };

  try {
    BigQuery.Tabledata.insertAll(request, projectId, datasetId, tableId);
    console.log("Feedback successfully recorded.");
  } catch (err) {
    console.error("Failed to insert into BigQuery: " + err);
  }
}
```

Key Considerations for HITL

- **Confidence Thresholds:** You can configure your system to automatically flag any prompt execution that receives a rating below a certain threshold (e.g., < 3 stars) for immediate human review.
- **Closed-Loop Enhancement:** Beyond simple data collection, these feedback systems create a closed loop that allows your AI agents to adapt organically to changing user needs over time.
- **Versioning:** Always ensure feedback is tied to the specific version of the prompt being used to avoid incorrectly penalizing a new update based on old data.

Would you like me to help you set up an automated alert system in Apps Script that notifies your team via email whenever a high-priority prompt receives a low rating?

Learn more about building data agents by chatting with your data in BigQuery. This video demonstrates how conversational analytics agents can directly interact with your BigQuery tables to generate insights from data like your feedback logs.

To implement an automated alert system for low ratings, you can use **Google Apps Script** to periodically query your `usage_analytics` BigQuery table and send an email notification if a high-priority prompt receives a low score.

Automated Low-Rating Alert Script

This script performs a daily check for any prompt that has received a 1 or 2-star rating within the last 24 hours.

JavaScript

```javascript
/**
 * Automatically checks for low ratings in BigQuery and sends an email alert.
 * Schedule this via Triggers (Clock icon) to run daily or hourly.
 */
function checkLowRatingsAndAlert() {
  const projectId = 'your-project-id';
  const alertRecipient = 'admin@yourcompany.com';

  // SQL to find ratings < 3 from the last 24 hours
  const sql = `
    SELECT prompt_id, feedback_rating, feedback_text, timestamp
    FROM \`your-project-id.prompt_mgmt.usage_analytics\`
    WHERE feedback_rating < 3
    AND timestamp >= TIMESTAMP_SUB(CURRENT_TIMESTAMP(), INTERVAL 1 DAY)
    ORDER BY timestamp DESC
  `;

  const request = { query: sql, useLegacySql: false };
  let queryResults;

  try {
    queryResults = BigQuery.Jobs.query(request, projectId);
  } catch (err) {
    console.error("BigQuery Query Failed: " + err);
    return;
  }

  const rows = queryResults.rows;
  if (!rows || rows.length === 0) {
    console.log("No low ratings found today.");
    return;
  }

  // Format email body
  let emailBody = "The following prompts received low ratings (< 3 stars) in the last 24 hours:\n\n";
  rows.forEach(row => {
    const f = row.f; // BigQuery result rows are an array of 'f' objects
    emailBody += `Prompt ID: ${f[0].v}\nRating: ${f[1].v}/5\nComment: ${f[2].v || 'No comment'}\nTime: ${f[3].v}\n--------------------\n`;
  });

  // Send the notification
  MailApp.sendEmail({
    to: alertRecipient,
    subject: "⚠️ ACTION REQUIRED: Low Prompt Rating Alert",
    body: emailBody
  });

  console.log(`Alert email sent for ${rows.length} low ratings.`);
}
```

Setup Instructions

1. **Enable BigQuery Service:** In your Apps Script editor, click the **+** next to "Services" and add the **BigQuery API**.
2. **Authorize Permissions:** Ensure your script project has the `BigQuery Data Viewer` and `Gmail` permissions.
3. **Create a Trigger:** * Click the **Triggers** (alarm clock) icon in the left sidebar.
   - Click **Add Trigger**.
   - Choose `checkLowRatingsAndAlert` as the function to run.
   - Select **Time-driven** as the event source.
   - Set it to run as a **Day timer** (e.g., every morning at 8:00 AM).

Why This Matters

- **Rapid Intervention:** By automating the check, you can identify failing prompts before they negatively impact more users.
- **Resource Efficiency:** Instead of manually monitoring logs, the script only alerts you when specific thresholds (conditions) are met.
- **Data-Driven Quality:** The feedback text included in the email provides direct human context for why a prompt may need revision.

This automated alert tutorial provides a visual walkthrough on setting up email notifications based on specific data conditions using Google Apps Script.