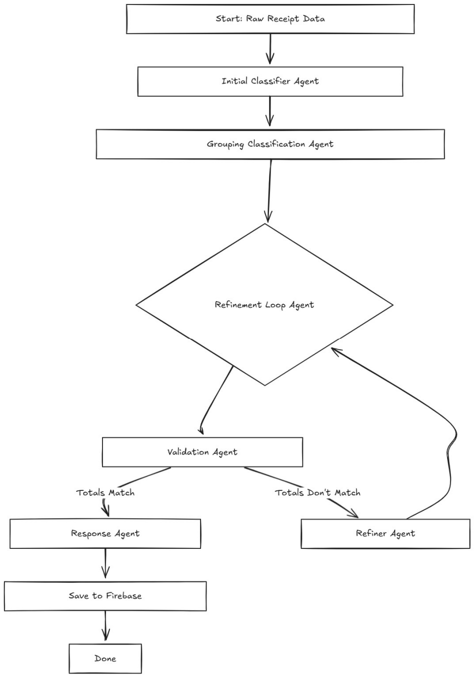
Milestone 3 (Week 9): Core Model Development & Integration

Group 5: Aliyyah Jackhan, Mohammed Aadil and Jonathan Chacko

1. Core Machine Learning Model(s) / AI Components

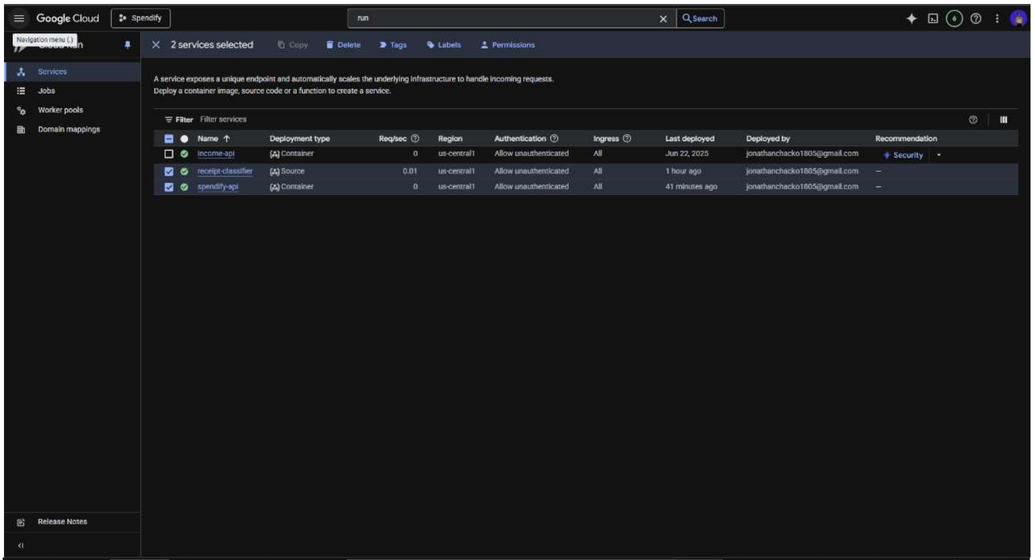
- **Vertex Document AI (Pre-Trained Model):**
  - Vertex has Document AI which Provides Us with Option of Training or Selecting a Pre-Trained Model to Use.
  - Since Expense Bills are available under the Pre-Trained Model we selected it and have the Option to Fine Tune it in the Future.
- **Classification**:
  - Implemented a highly structured ADK (Agent Development Kit) model pipeline for receipt item classification.
  - Pipeline uses strict, well-defined input and output formats for reliable and explainable classification results.
  - Switched from the previous Ollama-based LLM approach (Milestone 2) to the new multi-agent ADK pipeline for better control, structure, and validation.



- **Regression Model (Upcoming)**:
  - A regression model for spend prediction is planned and will be integrated in the next stage (along with the Dashboard).

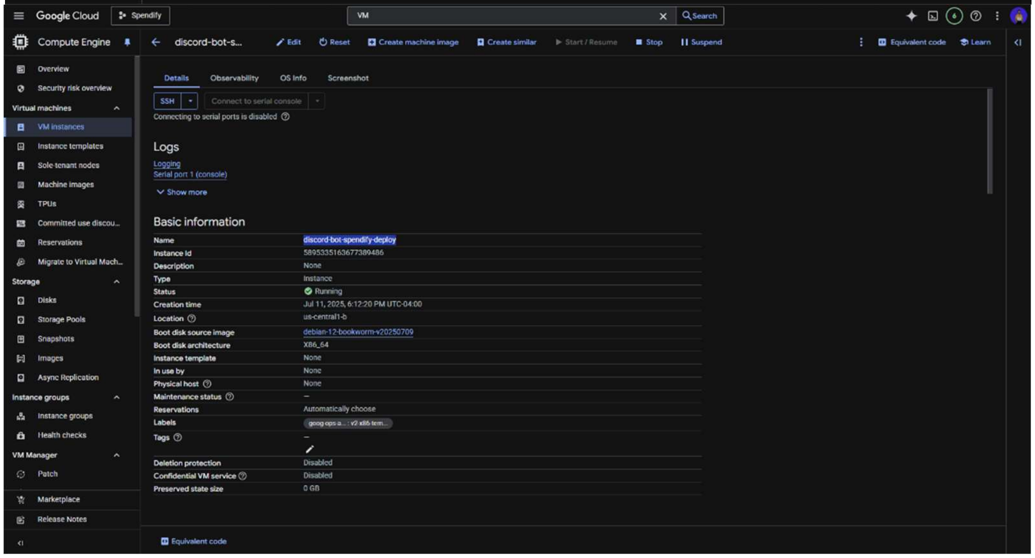
2. System Integration

- **Integration Demo**:
    - The gcp_adk_classification.py module demonstrates seamless integration between the main application and the ADK classification pipeline.
    - End-to-end flow: Discord bot → Flask API (main_api.py) → GCP Document AI OCR → ADK agent pipeline → Firebase storage.
- **Upcoming Dashboard**:
    - A user dashboard to visualize classified and predicted data is planned for the next milestone.
- **Screenshots from the Demo of the Deployed system so far:**



A service exposes a unique endpoint and automatically scales the underlying infrastructure to handle incoming requests.
Deploy a container image, source code or a function to create a service.

		Name ↑	Deployment type	Req/sec	Region	Authentication	Ingress	Last deployed	Deployed by	Recommendation
□	✓	income-api	Container	0	us-central1	Allow unauthenticated	All	Jun 22, 2025	jonathanchacko1805@gmail.com	Security
☑	✓	receipt-classifier	Source	0.01	us-central1	Allow unauthenticated	All	1 hour ago	jonathanchacko1805@gmail.com	–
☑	✓	spendify-api	Container	0	us-central1	Allow unauthenticated	All	41 minutes ago	jonathanchacko1805@gmail.com	–



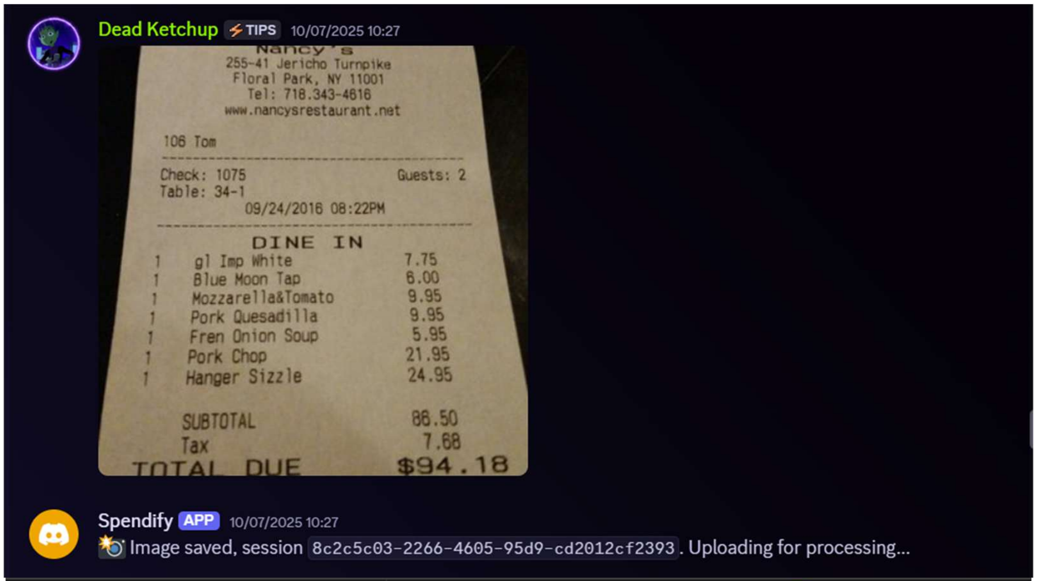
Connecting to serial ports is disabled

Logs

Logging
Serial port 1 (console)

Basic information

Name	discord-bot-spendify-deploy
Instance Id	5895335163677389486
Description	None
Type	Instance
Status	Running
Creation time	Jul 11, 2025, 6:12:20 PM UTC-04:00
Location	us-central1-b
Boot disk source image	debian-12-bookworm-v20250709
Boot disk architecture	X86_64
Instance template	None
In use by	None
Physical host	None
Maintenance status	–
Reservations	Automatically choose
Labels	goog-ops-a... : v2-x86-tem...
Tags	–
Deletion protection	Disabled
Confidential VM service	Disabled
Preserved state size	0 GB



Dead Ketchup ⚡TIPS 10/07/2025 10:27

Nancy's
255-41 Jericho Turnpike
Floral Park, NY 11001
Tel: 718.343-4616
www.nancysrestaurant.net

106 Tom

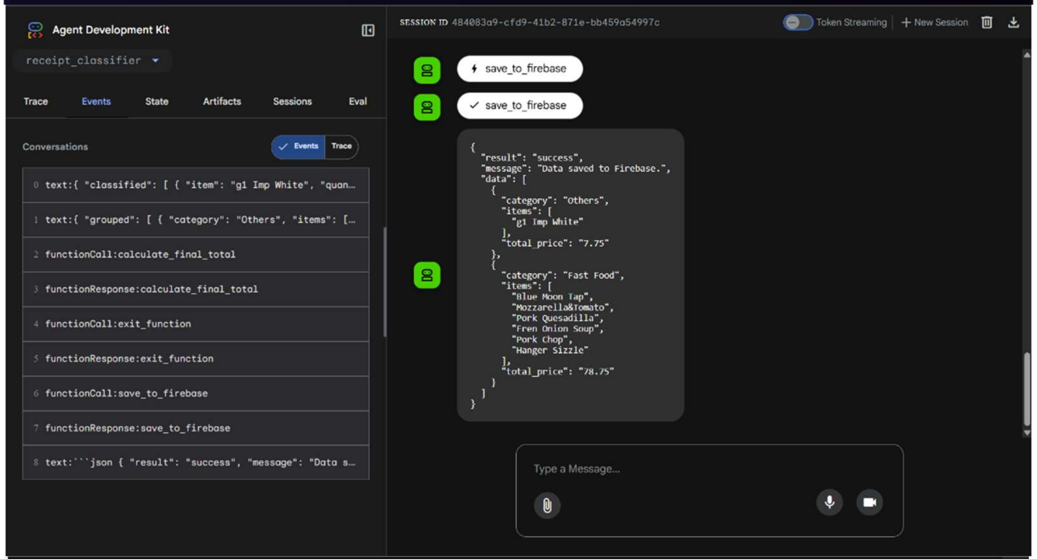
Check: 1075 Guests: 2
Table: 34-1
09/24/2016 08:22PM

DINE IN

1	g1 Imp White	7.75
1	Blue Moon Tap	6.00
1	Mozzarella&Tomato	9.95
1	Pork Quesadilla	9.95
1	Fren Onion Soup	5.95
1	Pork Chop	21.95
1	Hanger Sizzle	24.95
	SUBTOTAL	86.50
	Tax	7.68
	TOTAL DUE	$94.18

Spendify APP 10/07/2025 10:27

📸 Image saved, session `8c2c5c03-2266-4605-95d9-cd2012cf2393`. Uploading for processing...



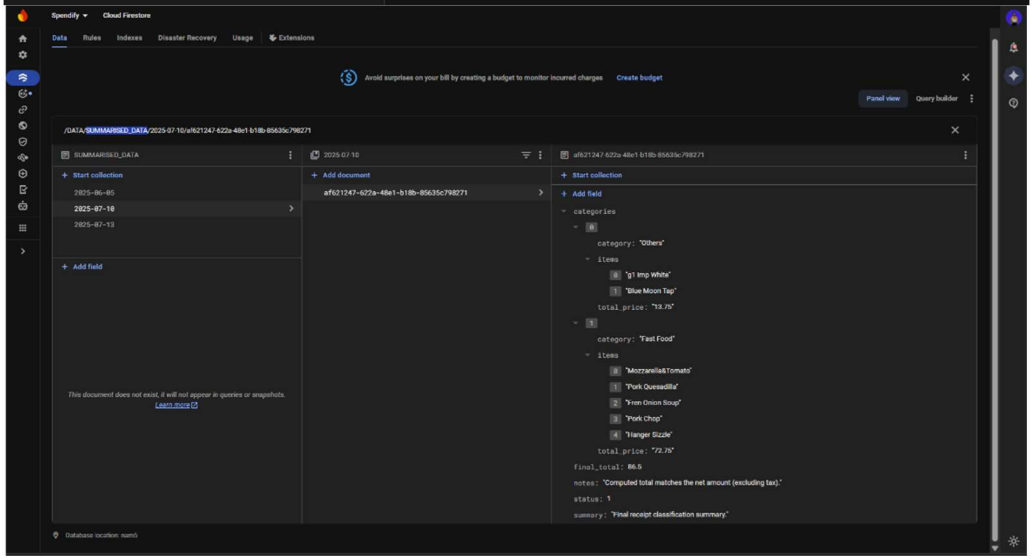
```
0 text:{ "classified": [ { "item": "g1 Imp White", "quan...
1 text:{ "grouped": [ { "category": "Others", "items": [...
2 functionCall:calculate_final_total
3 functionResponse:calculate_final_total
4 functionCall:exit_function
5 functionResponse:exit_function
6 functionCall:save_to_firebase
7 functionResponse:save_to_firebase
8 text:```json { "result": "success", "message": "Data s...
```

SESSION ID 484083a9-cfd9-41b2-871e-bb459a54997c

⚡ save_to_firebase

✓ save_to_firebase

```
{
  "result": "success",
  "message": "Data saved to Firebase.",
  "data": [
    {
      "category": "Others",
      "items": [
        "g1 Imp White"
      ],
      "total_price": "7.75"
    },
    {
      "category": "Fast Food",
      "items": [
        "Blue Moon Tap",
        "Mozzarella&Tomato",
        "Pork Quesadilla",
        "Fren Onion Soup",
        "Pork Chop",
        "Hanger Sizzle"
      ],
      "total_price": "78.75"
    }
  ]
}
```



Avoid surprises on your bill by creating a budget to monitor incurred charges

/DATA/SUMMARISED_DATA/2025-07-10/af621247-622a-48e1-b18b-85635c798271

SUMMARISED_DATA

- 2025-06-05
- 2025-07-10
- 2025-07-13

This document does not exist, it will not appear in queries or snapshots. Learn more

2025-07-10

- af621247-622a-48e1-b18b-85635c798271

af621247-622a-48e1-b18b-85635c798271

```
categories
  0
    category: "Others"
    items
      0 "g1 Imp White"
      1 "Blue Moon Tap"
    total_price: "13.75"
  1
    category: "Fast Food"
    items
      0 "Mozzarella&Tomato"
      1 "Pork Quesadilla"
      2 "Fren Onion Soup"
      3 "Pork Chop"
      4 "Hanger Sizzle"
    total_price: "72.75"
final_total: 86.5
notes: "Computed total matches the net amount (excluding tax)."
status: 1
summary: "Final receipt classification summary."
```

Database location: nam5

- The full pipeline was tested end-to-end:
  - Discord Bot uploads a receipt image
  - API receives and processes the image, runs OCR
  - Data sent to ADK classification pipeline (Initial Classifier → Grouping Agent → Validation/Refinement Loop → Final Response Agent)
  - Classification output is validated and stored in Firebase
- **Result:** API endpoint returns predictions successfully.

3. Initial Deployment Steps

Deployment guides and step-by-step instructions are provided in the following repository links. For each major component, use the commands below for deployment:

a. ADK Classification Agent

- Guide: deploy-adk.md
- **Cloud Run Deployment Command:**

```
adk deploy cloud_run \
  --project=$GOOGLE_CLOUD_PROJECT \
  --region=$GOOGLE_CLOUD_LOCATION \
  --service_name=$SERVICE_NAME \
  --app_name=$SERVICE_NAME \
  --port=8080 \
  --log_level=info \
  --with_ui \
  ./receipt_classifier
```

b. Flask API

- Guide: deploy-api.md
- **Build & Deploy to Cloud Run:**

```
gcloud builds submit --tag gcr.io/$GOOGLE_CLOUD_PROJECT/$SERVICE_NAME .
gcloud run deploy $SERVICE_NAME \
  --image gcr.io/$GOOGLE_CLOUD_PROJECT/$SERVICE_NAME \
  --platform managed \
  --region $GOOGLE_CLOUD_LOCATION \
  --allow-unauthenticated
```

c. Discord Bot

- Guide: deploy-bot.md
- **VM Deployment Command:**

```
ssh -i ~/.ssh/gcp_key user@VM_IP
tmux new -s discordbot
python3 bot.py
```

Each document contains Dockerfile usage, environment variable setup, GCP Cloud Run deployment, and troubleshooting tips specific to each core component.

4. Progress, Planning & Next Steps

From Milestone 2 → Milestone 3

- Major change: Replaced Ollama/LLM-based classification with the robust ADK model pipeline.
- Improved reliability, structure, and validation by enforcing strict input/output schemas and a multi-agent review/refine process.

Detailed Task Breakdown (Next 3 Weeks, by Role)

- **Regression model implementation & integration (Aliyyah):**
  - Predict user spend (time series/regression)
  - Integrate into existing backend pipeline
- **Dashboard development (Aadil):**
  - Build initial UI for receipt and prediction visualization
  - Connect dashboard to Firebase/API
- **Deployment & system integration (Jonathan):**
  - Deployment of Regression Model and Dashboard
  - Full system end-to-end testing

5. Challenges Faced to Achieve Milestone 3

- Ollama proved inaccurate and unreliable for classification, which led us to adopt GCP ADK as the primary classification engine.
- Swapping from Ollama to ADK allows us to access the larger computational power of GCP and benefit from a more scalable, production-grade environment.

- Integrating the highly structured ADK model pipeline with strict input/output, which required reworking the entire classification workflow.
- GCP ADK is an amazing tool for building agent pipelines, but it lacks comprehensive documentation and real-world examples, which significantly slowed down development.
- Deployment was difficult for GCP ADK for the same reasons as above, with a lot of trial and error required to achieve a working cloud setup.
- Ensuring compatibility and smooth data handoff between Discord Bot, Flask API, GCP Document AI, and ADK pipeline.
- Debugging Firestore data serialization and managing Firestore schema changes.
- Handling OCR inconsistencies in receipt formats from GCP Document AI.
- Adapting deployment strategies (switching from Ollama to ADK), requiring updates to containerization and cloud setup.

6. Remaining Challenges

- **Regression Model:** Need to design and validate regression predictions; may need additional user data.
- **Dashboard:** Ensuring robust connection between dashboard and backend, and handling real-time updates.
- **Validation:** Ensuring ground-truth data for accurate model validation (classification & regression).
- **Performance:** Scaling API/bot for multiple concurrent users; latency testing on cloud.