Space42 Bounty Starter Pack

This document outlines topics, tools, and general tips and tricks that participants are encouraged to familiarise themselves with before the competition, should they choose to do this bounty.

This document does not describe the bounty. You're going to have to wait till the event for that 😀.

Difficulty Level

Difficulty: Intermediate (Conceptual and design-focused)

This bounty is approachable for participants with basic programming knowledge. Over the course of the challenge, participants are expected to engage with concepts such as AI reasoning, explainability, and governance at a conceptual and applied level, rather than through advanced model development.

Recommended Team Composition

Teams with at least one member comfortable with programming are recommended.

Multidisciplinary collaboration will be especially valuable for system design, user experience, and presentation. A great code solution that solves the wrong business problem will not win.

Bounty Prizes

3 teams will be chosen to be awarded with prizes.

- 1st Place winner: 5000 AED
- 2nd Place winner: 3000 AED
- 3rd Place winner: 2000 AED

Topics encouraged to look at

1. Business Strategy and Operational Perspectives

Participants are encouraged to be familiar with business processes and operational challenges that can help shape solutions which are practical and impactful. A solution will fail if it doesn't align with the company's compliance needs or strategic goals.

Relevant areas to research:

- Process Mapping: Learn how to diagram a business workflow (Inputs → Process → Outputs) to identify bottlenecks in *any* department.
- Compliance & Privacy: Understanding that in a corporate environment, data security (GDPR, PII handling) is often more important than feature speed.
- Defining Success: How would a business measure the ROI of this tool? (e.g., Cost reduction vs. Quality improvement).

Resources:

- *IBM: What is Business Process Mapping?*
- *Harvard Business Review: The Real World Value of AI*
- *Zapier: How to Measure Automation ROI*
- Google Cloud: KPIs for Gen AI

2. Designing Trustworthy AI (User Experience)

Participants are encouraged to explore the unique challenges of designing interfaces for autonomous agents. Unlike standard apps, agents take actions on their own. The core challenge is: How do you design for trust?

They should understand how to handle "failure states" (when the AI gets it wrong) and how to design "Human-in-the-Loop" handoffs so the user always feels in control.
Topics to Look Into:

- Conversational UX: Scripting natural, empathetic, and clear dialogue.
- Explainability: Designing UI elements that show *why* the AI made a decision.
- Security & Robustness: Understanding Prompt Injection risks (e.g., users trying to trick the bot) and how to guard against them.

Resources:

- Google PAIR (People + AI Research): Guide to Explainability
- Voiceflow: Conversation Design Best Practices

- *Lakera: Guide to Prompt Injection*

3. Agent vs Chat Bot

Participants are encouraged to explore how "Agents" differ from standard chatbots. An agent doesn't just answer; it plans and uses tools to change the state of a system.

They should understand how to use function calling (or tool use) to let an LLM interact with a calendar, a database, or an email service.

Participants are encouraged to explore how "Agents" differ from standard chatbots. An agent doesn't just answer; it uses tools to change the state of a system.

- Tool Use: How to let an LLM interact with a calendar, database, or API.
- RAG (Retrieval Augmented Generation): How to give an agent access to private documents (like a PDF policy handbook).

Resources:

- LangChain Agents: Quickstart Guide – *How to build agents that use tools.*
- DeepLearning.AI: Functions, Tools and Agents with OpenAI – *A free short course on building agentic systems.*
- Microsoft AutoGen: Enabling Multi-Agent Chat
- Pinecone: What is RAG?
- LangChain: RAG Tutorial

4. Deployment

Participants are encouraged to explore how to package code in a web framework and deploy it locally or via cloud.

- **Streamlit / Gradio:** Great for building rapid AI prototypes with minimal frontend code.
- **FastAPI / Flask:** Essential for serving your Agent as an API that a frontend can talk to.

Resources:

- Building a Chat App with Streamlit and OpenAI
- Ollama: Get Started with Local LLMs
- Hugging Face: Open LLM Leaderboard

5. Presenting & Pitching solutions

Participants are encouraged to think about how to clearly communicate their solution to both technical and non-technical audiences.

Useful areas to explore include:

- Clearly defining the problem being addressed
- Explaining why the solution matters and who it helps
- Demonstrating system behavior through simple workflows
- Communicating trade-offs, limitations, and future extensions

Strong presentations often focus on clarity and value, rather than implementation details alone.

Resources:

- Y Combinator: How to Pitch Your Product
- Guy Kawasaki: The Only 10 Slides You Need in a Pitch

General Tips and Tricks

- AI-assisted development tools can be useful for ideation, exploration, and implementation.

- If using AI powered tools that have access to your repository, always use plan mode and reiterate the plan before you let the agent execute the commands.

- If you have a team from beforehand, set up the repository and look into git commands in case you are using version control. https://education.github.com/git-cheat-sheet-education.pdf

- Look into best practices for documenting APIs if you connect to external tools (e.g., Google Calendar, Slack, or a mock database)

- Learn how to use tools like Faker to generate realistic dummy data (fake names, roles, salaries) in case data is unavailable.