# Project Canvas: AI-Powered Amazon Shoping Assistant

Prepared by: Likhitha Guggilla

Date: October 4, 2025

Version: 1.0

### 1. Problem Statement

Navigating a vast e-commerce platform like Amazon presents significant challenges for customers. They often face information overload when trying to choose the best product, spending considerable time sifting through thousands of product descriptions, specifications, and often contradictory customer reviews. This process is inefficient and can lead to choice paralysis or poor purchasing decisions. There is a clear need for an intelligent, interactive solution that can instantly synthesize this vast amount of information, provide personalized guidance, and streamline the shopping process from discovery to checkout.

This project aims to build an AI-powered chatbot assistant that directly addresses these gaps by providing immediate, context-aware answers, summarizing reviews, comparing products, and executing tasks on the user's behalf.

### 2. Data & Knowledge

The assistant's knowledge will be sourced from publicly available Amazon product datasets. The primary dataset will be the **Amazon Reviews and Metadata 2023** collection, which provides a rich source of textual information.

* **Source:** <https://amazon-reviews-2023.github.io/main.html>
* **Data Fields:**
  + **Product Metadata:** Product titles, detailed descriptions, features, specifications, price, and category.

|  |  |  |
| --- | --- | --- |
| Field | Type | Explanation |
| main\_category | str | Main category (i.e., domain) of the product. |
| title | str | Name of the product. |
| average\_rating | float | Rating of the product shown on the product page. |
| rating\_number | int | Number of ratings in the product. |
| features | list | Bullet-point format features of the product. |
| description | list | Description of the product. |
| price | float | Price in US dollars (at time of crawling). |
| images | list | Images of the product. Each image has different sizes (thumb, large, hi\_res). The “variant” field shows the position of image. |
| videos | list | Videos of the product including title and url. |
| store | str | Store name of the product. |
| categories | list | Hierarchical categories of the product. |
| details | dict | Product details, including materials, brand, sizes, etc. |
| parent\_asin | str | Parent ID of the product. |
| bought\_together | list | Recommended bundles from the websites. |

* + **Customer Reviews:** User-submitted reviews, ratings, and comments.

|  |  |  |
| --- | --- | --- |
| Field | Type | Explanation |
| rating | float | Rating of the product (from 1.0 to 5.0). |
| title | str | Title of the user review. |
| text | str | Text body of the user review. |
| images | list | Images that users post after they have received the product. Each image has different sizes (small, medium, large), represented by the small\_image\_url, medium\_image\_url, and large\_image\_url respectively. |
| asin | str | ID of the product. |
| parent\_asin | str | Parent ID of the product. Note: Products with different colors, styles, sizes usually belong to the same parent ID. The “asin” in previous Amazon datasets is actually parent ID. **Please use parent ID to find product meta.** |
| user\_id | str | ID of the reviewer |
| timestamp | int | Time of the review (unix time) |
| verified\_purchase | bool | User purchase verification |
| helpful\_vote | int | Helpful votes of the review |

* **Scope:** To ensure manageability for this project, the initial phase will focus on a specific, well-defined subset of the data from the 'Clothing\_Shoes\_and\_Jewelry' category. This focused dataset will be indexed and used to build and evaluate the first versions of the assistant.

### 3. AI Approach & Methodology

The core of this project will be a **Retrieval-Augmented Generation (RAG)** **and Agentic** system. This approach ensures that the assistant's responses are grounded in the actual product data, minimizing inaccuracies and "hallucinations."

* **Retrieval:** Product metadata and reviews will be processed and converted into numerical vector embeddings using a sentence-transformer model. These embeddings will be stored and indexed in a Vector Database (Qdrant). When a user asks a question, the system will search this database to retrieve the most relevant chunks of information.
* **Generation:** The retrieved information, along with the user's original query, will be fed into a Large Language Model (LLM) (e.g., a model from the Gemini, Llama, or GPT families). The LLM's role is to synthesize this context into a coherent, natural-language response.
* **Prompt Engineering:** We will develop and refine system prompts to guide the LLM's behavior, ensuring it adheres to its persona as a helpful shopping assistant and provides accurate, concise answers.
* **Agentic Functionality:** For task-oriented features (e.g., adding to cart/purchasing the order/returning the item), the system will be enhanced to function as an "agent." The LLM will be granted access to a set of predefined "tools" (API functions) that it can decide to call based on the user's intent.

### 4. High-Level Architecture Outline

### A computer screen shot of a diagram AI-generated content may be incorrect.

### Link to the flowchart: <https://github.com/LikhithaGuggilla/ai_engineering/blob/main/system-architecture.png>

The system will be composed of several key, interacting components:

1. **Frontend (Streamlit UI):** A simple, clean chat interface where the user interacts with the assistant.
2. **Backend (FastAPI)/ Orchestrator:** The central logic hub of the application. It will:
   * Receive user queries from the frontend.
   * Recognize user intent (is it a question or a task?) and generate multi-step execution plan.
   * Invoke the Retrieval module to fetch relevant context from the Vector DB.
   * Invoke the LLM to generate a response.
   * Execute tools/APIs for task-oriented requests using MCP/A2A communication protocols.
3. **Vector Database (Qdrant):** An indexed, searchable knowledge base containing the vector embeddings of all product information and reviews.
4. **LLM Service (Gemini, OpenAI, Groq):** An API endpoint for a pre-trained Large Language Model that handles the natural language generation.
5. **Tools / API Layer:** A set of functions that connect to external services. Initially, these will be mock APIs like addToCart(product\_id) and placeOrder(cart\_details). The Human-in-the-Loop (HITL) mechanism will be triggered here for sensitive actions.

### 5. Performance Metrics & Evaluation Rules

To measure the assistant's effectiveness, we will track the following metrics:

* **Performance Metrics**
* **User Satisfaction (CSAT):** A simple "thumbs up/thumbs down" feedback mechanism after each response to gauge user sentiment.
* **Latency:** The time from when a user sends a query to when they receive a response. We will aim for a response time of under 3 seconds.
* **Evaluation**
* **Response Quality:**
  + **Faithfulness:** How factually accurate is the generated response based on the retrieved context? (Scale 1-5)
  + **Answer Relevancy:** How relevant is the response to the user's question? (Scale 1-5)
* **Task Success Rate:** The percentage of times the agent successfully completes a requested task (e.g., correctly adding the specified item to the cart).

### 6. Resources & Stakeholders

* **Key Stakeholders:**
  + Project Lead / AI Engineer (You)
  + Bootcamp Instructors / Mentors
  + Beta Testers (Peers, potential users)
* **Technical Resources:**
  + **Foundational Models:** Access to a hosted LLM API (e.g., Google AI Studio, OpenAI, Groq).
  + **Software:** Python, Jupyter Notebooks, LangGraph/LlamaIndex/CrewAI framework, FastAPI, Qdrant, LiteLLM, Splx, Ragas, LangSmith, Docker

A screenshot of a computer

AI-generated content may be incorrect.

### 7. Risks & Mitigation

* **Inaccurate Responses (Hallucination):**
  + *Risk:* The LLM may generate information not present in the source data.
  + *Mitigation:* The RAG architecture inherently limits this risk. Strict prompting will instruct the model to only use the provided context.
* **Outdated Information (Out of the project scope):**
  + *Risk:* Product prices, availability, and reviews change frequently.
  + *Mitigation:* Implement a data ingestion pipeline that periodically refreshes the knowledge base from the source dataset but this is out of the project scope.
* **Handling Ambiguous Queries:**
  + *Risk:* Users may ask vague or unclear questions.
  + *Mitigation:* Program the assistant to ask clarifying questions when the user's intent is not clear.

### 8. Deployment

* The application will be containerized using Docker to ensure consistency across environments.

### 9. Timeline & Milestones

The project will be developed over a 8-week period, following an agile, sprint-based approach.

* **Week 1:** **Foundation & Basic Q&A.**
  + Set up the development environment.
  + Process and index a subset of the product data into a vector database.
  + Build a basic RAG pipeline to answer simple questions about products.
* **Week 2 & 3:** **Enhanced Retrieval & Summarization.**
  + Implement review summarization functionality.
  + Build a product comparison feature.
  + Refine retrieval strategies to improve response relevance.
* **Week 4 & 5:** **Agentic Capabilities.**
  + Develop and integrate "tools" for adding products to a cart.
  + Implement a mock checkout process with a Human-in-the-Loop checkpoint.
  + Refine intent detection to differentiate between questions and tasks.
* **Week 6 & 7:** **Personalization & Deployment.**
  + Implement a basic personalization feature based on chat history.
  + Containerize the application and deploy it .
  + Conduct final testing and prepare for project presentation.