Project Report – ShopAssist: AI-Powered E-commerce Chatbot

# 1. Introduction

ShopAssist is a full-stack e-commerce chatbot developed for Uplyft's Full Stack Intern Case Study – June 2025. The system leverages Flask (Python), HTML/CSS/JS for the web application and integrates Microsoft Power Platform Copilot Studio to deliver intelligent, conversational product search and customer support.

# 2. Technology Stack

- Frontend: HTML, CSS, JavaScript  
- Backend: Flask (Python)  
- AI Integration: Microsoft Power Platform Copilot Studio  
- Data: products.json (sample) & DummyJSON API  
- Deployment: Localhost (http://127.0.0.1:5000/)

# 3. Key Features

- Conversational Shopping through chatbot interface  
- AI-powered product search via natural language  
- Product browsing and detailed view  
- Shopping cart management with add/remove/update  
- Support for FAQs and instant help  
- Accessibility with screen reader and keyboard navigation

# 4. System Architecture

The system consists of three main layers:  
- Flask backend to manage routes and API endpoints  
- Frontend chat interface and product pages  
- AI-powered query processing using Copilot Studio with fallback to local JSON or DummyJSON API

# 5. Sample Queries

- "Show me waterproof watches under $100"  
- "Find me red running shoes for women"  
- "Add iPhone 9 to my cart"  
- "Remove the smartwatch from cart"  
- "What are the current deals?"

# 6. Results Obtained

The application was successfully tested on multiple scenarios. AI-powered search delivered relevant results based on user queries, and the fallback mechanism using local data/API ensured consistent performance. The cart management, category browsing, and support features worked smoothly.

# 7. Installation & Setup

1. Clone the repository: `git clone https://github.com/Avinash-yadav103/Uplyft\_Ecom\_bot`  
2. Navigate to project folder: `cd Uplyft`  
3. Create virtual environment: `python -m venv venv`  
4. Activate venv: `./venv/Scripts/activate`  
5. Install dependencies: `pip install -r requirements.txt`  
6. Start Flask server: `python index.py`  
7. Access: http://127.0.0.1:5000/

# 8. Dummy Data

The system uses a sample `products.json` file for local product browsing. If unavailable, it fetches data from DummyJSON API.

# 9. Accessibility

Accessibility features include keyboard navigation, ARIA roles, screen reader support, and high contrast UI.

# 10. License

This project is for educational and demonstration purposes only. For commercial use, contact the author.