Architecture Design

Machine Learning NLP Price Negotiation Chatbot

Sarampenta Praveen

Contents

Abstract **………………………………………………………...........**

1. Introduction **…………………………………………………….**
   1. Why this Architecture Design Document?**……….**
2. Architecture **……………………………………………………**
3. Architecture Design **…………………………………………..**
   1. Data Collection **……………………………………..**
   2. Data Description **……………………………………**
   3. Components **………………………………………..**
   4. State management **…………………………………**
   5. Routing **………………………………………………**
   6. Styling **……………………………………………….**

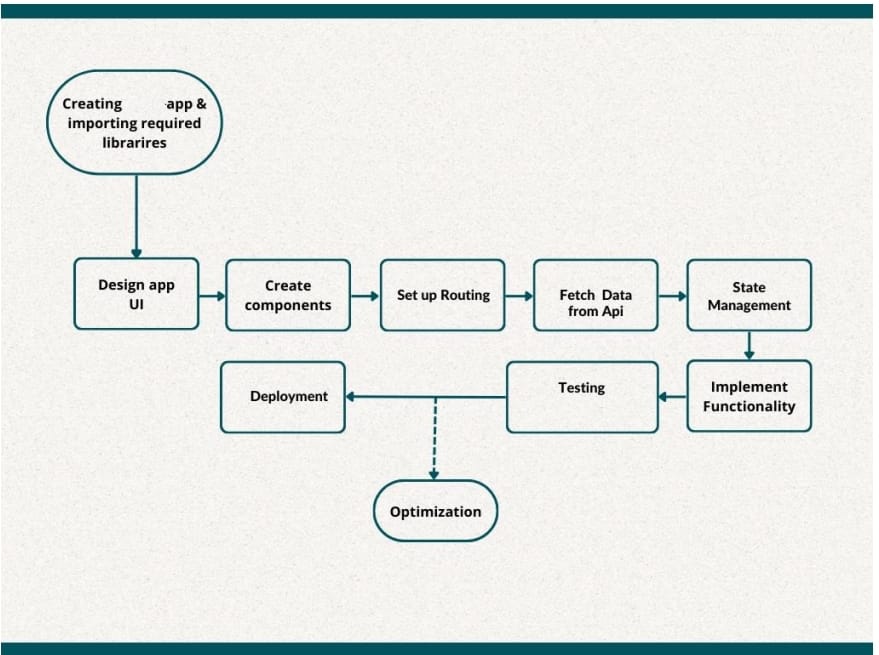
**Abstract:**

This project focuses on the development of a Machine Learning Natural Language Processing (NLP) Price Negotiation Chatbot, employing various technologies to enhance its functionality. The chatbot utilizes Dialogflow as an interface for seamless communication between the backend and frontend, providing a user-friendly experience. The Cassandra Database is employed to efficiently store login details, track product availability, and preserve negotiated prices and entire conversation histories.

The backend of the chatbot is powered by Python Flask, ensuring robust and scalable performance. To facilitate accessibility across devices, NGROK is employed to run the chatbot as an HTTPS server. This setup allows users to engage in price negotiations and product inquiries effortlessly, while also enabling the system to store and retrieve pertinent information for a streamlined user experience. The amalgamation of these technologies forms a comprehensive solution for an intelligent, efficient, and adaptable price negotiation Chatbot.

1. Introduction
   1. Why this Architecture Design Document?

The main objective of the Architecture design documentation is to provide the internal logic understanding of the Price Negotiation Chatbot code. The Architecture design documentation is designed in such a way that the programmer can directly code after reading each module description in the documentation.

1. Architecture

n

1. Architecture Design
   1. Data Description

The dataset has been created using Dialogflow.

[https://dialogflow.cloud.google.com/#/editAgent/limupa-chatbot-wmug/](https://dialogflow.cloud.google.com/%23/editAgent/limupa-chatbot-wmug/)

* 1. Data Description

After making api call you will get a JSON array containing information about various product details along with prices in an ecommerce store and also indicates the intent and entities of the product, Here’s a description of the data.

**Product Information:** Each object in the array represents a product available sale and their synonyms.

**Product details include:**

* **ID:** A unique Identifier for the product.
* **Name:** Name of the product (e.g.: IPhone 7, Samsung s9….)
* **Price:** Price of the product is specified in the Indian rupees
* **Image:** URL of the product image
  1. Components.
* The frontend is structured using a component-based architecture, with reusable UI components for various elements such as headers, product listing, cart, and checkout.
* Component are organized into a hierarchy with higher-level components composing lower-level ones to create complex UI layouts.
  1. State Management
* State management is handled using Flask API for managing global application state.
* Context providers are used to encapsulate state related to user authentication, cart items, billing process and negotiations and other application data.
* Dialogflow API gives faster response to the chatbot.
  1. Routing
* Flask routing is used for client-side routing, enables navigation between different pages and components within the application.
* Routers are defined to map URLs t corresponding components, allowing for a single-page application.
* By using routing techniques had connection between Dialogflow and flask application.
  1. Styling
* CSS modules or styled-components are used for styling components, providing styles and better maintainability.
* Responsive design principles are applied to ensure the application is accessible and usable across different devices and screen sizes.