Software Requirements Specification (SRS)

AI Receptionist MVP

---

1. Introduction

1.1 Purpose

The purpose of this document is to define the functional, non-functional, and technical requirements for the AI Receptionist MVP. This system will assist small businesses by automating customer interactions, answering common inquiries, scheduling appointments, and providing personalized responses.

1.2 Scope

The AI Receptionist MVP will:

- Fetch business details from a database.
- Schedule and manage appointments.
- Process customer inquiries using NLP.
- Provide personalized responses based on stored templates.
- Log all interactions for analytics.

This system will be implemented using **FastAPI, MongoDB, and NLP models (Hugging Face Transformers)**.

1.3 Definitions, Acronyms, and Abbreviations

- **MVP** - Minimum Viable Product
- **NLP** - Natural Language Processing
- **API** - Application Programming Interface
- **CRUD** - Create, Read, Update, Delete
- **DB** - Database (MongoDB in this project)

1.4 References

- MongoDB Documentation: https://www.mongodb.com/docs/
- FastAPI Documentation: https://fastapi.tiangolo.com/
- Hugging Face Transformers: https://huggingface.co/docs/transformers/

---

2. System Overview

2.1 System Architecture

The AI Receptionist MVP consists of the following components:

1. **Database (MongoDB)** - Stores business data, appointment schedules, custom responses, and customer interactions.
2. **FastAPI Backend** - Handles API requests, processes NLP queries, and manages appointments.
3. **AI-Powered Query Processor** - Uses NLP models to understand customer inquiries.
4. **Admin Interface** - Allows businesses to update response templates and manage appointments.

2.2 Users & Stakeholders

- **Business Owners/Admins**: Manage business information and response templates.
- **Customers**: Interact with the AI to get information and book appointments.
- **System Developers**: Maintain and update the AI Receptionist MVP.

---

3. Functional Requirements

3.1 Business Information Integration

- The system shall store business details (name, services, hours, contact info) in MongoDB.
- The system shall provide an API endpoint (/business_info) to fetch business details.

- The system shall allow businesses to update their information.

3.2 Appointment Scheduling

- The system shall store available slots and booked appointments.
- The system shall provide an API endpoint (/schedule) to book appointments.
- The system shall prevent double-booking and handle invalid time slots.
- The system shall return confirmation messages for successful bookings.

3.3 AI-Powered Customer Interaction

- The system shall use an NLP model to process customer queries.
- The system shall fetch relevant information from the database.
- The system shall store all customer queries in MongoDB for analytics.

3.4 Personalization & Custom Responses

- The system shall allow businesses to store custom response templates in MongoDB.
- The system shall provide an API endpoint (/custom_responses) to add, update, and retrieve custom responses.
- The system shall prioritize custom responses before using NLP.

---

4. Non-Functional Requirements

4.1 Performance Requirements

- The system shall respond to customer queries within **2 seconds**.
- The system shall handle at least **100 concurrent users**.

4.2 Security Requirements

- The system shall encrypt all stored customer data.
- The system shall implement authentication for admin endpoints.

4.3 Usability Requirements

- The system shall provide a user-friendly API for interaction.
- The system shall log all customer interactions for future improvements.

5. System Design & Implementation

5.1 Database Design (MongoDB)

Collections:

1. business_data - Stores business details.
2. calendar - Stores appointment slots.
3. customer_queries - Logs customer interactions.
4. custom_responses - Stores predefined responses.

5.2 API Endpoints (FastAPI)

Endpoint	Method	Description
/business_info	GET	Fetch business details
/schedule	POST	Book an appointment
/ask	POST	Process customer queries using AI
/custom_responses	POST	Add a custom response
/custom_responses/{query_type}	GET	Retrieve a custom response

/custom_responses/{query_type}	PUT	Update a custom response

5.3 AI Model (NLP Processing)

- **Model Used**: facebook/bart-large-mnli
- **Task**: Question answering based on business data.

---

6. Deployment & Maintenance

6.1 Deployment Environment

- Backend: **FastAPI**
- Database: **MongoDB** (Atlas or Local Instance)
- Hosting: **AWS/GCP/Heroku** (for future scalability)

6.2 Maintenance Plan

- **Monthly Updates**: Improve NLP accuracy and add new features.
- **Bug Fixes**: Regular monitoring for API failures.
- **Database Optimization**: Ensure efficient query execution.

---

7. Future Enhancements

- Integrate **voice-based AI assistant**.
- Add **multi-language support** for global accessibility.
- Implement a **chatbot UI** for seamless customer interactions.

---

8. Conclusion

This document outlines the AI Receptionist MVP’s requirements, ensuring a scalable, efficient, and intelligent customer support solution. The system will automate customer interactions, improve service efficiency, and enhance user experience.
