Software Requirements Specification (SRS)

Question Answering System using BERT Fine-Tuning Model

1. Introduction

Purpose: Define functional and non-functional requirements for a Question Answering System built using fine-tuned BERT and deployed using Flask on Render.

Scope: The system accepts context and question input, processes using BERT, and returns extractive answers.

2. Overall Description

Product Perspective: Web application with Flask backend, BERT model (HuggingFace Transformers), deployed on Render.

User Classes: General User, Admin.

Operating Environment: Python 3.10+, Flask, PyTorch, Transformers, Render.

3. Functional Requirements

FR-1: Accept context paragraph.

FR-2: Accept user question.

FR-3: Validate input fields.

FR-4: Tokenize input using BERT tokenizer.

FR-5: Load fine-tuned BERT model.

FR-6: Predict answer span.

FR-7: Extract answer text.

FR-8: Display answer clearly.

FR-9: Show confidence score (optional).

FR-10: Handle no-answer cases.

FR-11: Store logs.

FR-12: Log system errors.

4. Non-Functional Requirements

Performance: Response time <= 3 seconds.

Scalability: Render auto-scaling support.

Security: HTTPS, input validation.

Usability: Responsive UI.

Reliability: 99% uptime.

5. Architecture

User -> Flask App -> BERT Model -> Answer Output

Logs stored in database (optional).

6. API Endpoints

GET /

POST /predict

GET /health

7. Deployment

Use GitHub repository, requirements.txt, gunicorn, and deploy on Render.

8. Constraints

BERT token limit (512 tokens).

Cold start in free tier.

9. Future Enhancements

Multi-document QA, vector database, conversational interface.

10. Acceptance Criteria

System loads model successfully.

User can submit question and receive answer.

Deployment accessible via public URL.