3. SRS

1. Introduction

1.1 Purpose

This document specifies the software requirements for a Bitcoin price prediction system using an LSTM (Long Short-Term Memory) model. The system is designed to provide accurate predictions of Bitcoin prices to help users make informed trading decisions. This SRS aims to clearly outline functionality, constraints, and the interaction of the system with users to ensure alignment between stakeholders and the development team.

1.2 Document Conventions

This document follows the IEEE standard for SRS documents. Key terms and required software tools are listed and explained in the Glossary and References sections.

1.3 Intended Audience and Reading Suggestions

This document is intended for the project team, including developers, project managers, and testers, as well as external stakeholders such as investors and end-users (cryptocurrency traders). It is suggested that readers familiarize themselves with the project background before proceeding to the specific requirements.

1.4 Project Scope

The project will develop a predictive software system that utilizes an LSTM model to forecast Bitcoin prices based on historical data. The system will provide predictive insights through a user-friendly interface that allows traders to view predicted prices. Features include data ingestion, model training, result prediction, and a graphical display of historical and predicted prices.

1.5 References

- https://github.com/uttej2001/Bitcoin-Price-Prediction/tree/main

---

2. Overall Description

2.1 Product Perspective

The system is a standalone application but could potentially integrate with existing trading platforms through APIs in future versions. It is reliant on continuous data feeds from cryptocurrency exchanges and uses historical pricing data to train the LSTM model.

2.2 Product Features

- Data collection and preprocessing
- LSTM model training
- Price prediction generation
- User interface for data visualization
- Option to adjust model parameters (for advanced users)

2.3 User Classes and Characteristics

- **End-Users (Traders)**: Require accurate and timely predictions with an easy-to-use interface.
- **Data Scientists**: Need the ability to tweak algorithms and access detailed model performance data.
- **System Administrators**: Manage system operations and ensure data integrity and security.

2.4 Operating Environment

The system will be developed in Python, utilizing libraries such as TensorFlow, Keras, and Pandas. It will run on Linux and Windows operating systems and will require a GPU for efficient model training.

2.5 Design and Implementation Constraints

- The system requires a continuous and reliable data source.
- High-performance computing resources are needed for real-time data processing and model training.
- The system must comply with financial data protection regulations.

2.6 Assumptions and Dependencies

- Accurate predictions depend on the continuous availability of recent and historical Bitcoin price data.
- System performance is dependent on the advancements in LSTM and other machine learning technologies.

---

3. System Features

3.1 Data Collection and Management

Description: The system will automatically collect data from predefined cryptocurrency exchange APIs. The data will be cleaned, preprocessed, and stored in a structured format suitable for LSTM processing.
Requirements:

- R1.1: The system shall update the dataset at least every hour.
- R1.2: The system shall handle anomalies and outliers in data before processing.

3.2 LSTM Model Training and Prediction

Description: The LSTM model will be trained on historical data and used to make price predictions.
Requirements:

- R2.1: The system shall retrain the model on a daily basis or upon significant market events.
- R2.2: The system shall provide predictive outputs with an accuracy metric.

3.3 User Interface

Description: A graphical user interface that displays both historical data and predictions clearly.
Requirements:

- R3.1: The interface shall display predictive results graphically.
- R3.2: The interface shall allow users to view details on demand.

---

4. External Interface Requirements

4.1 User Interfaces

- GUI developed using a NextJS React Framework.

4.2 Hardware Interfaces

- Requires GPU support for model training.
- Standard PC hardware for running the software.

4.3 Software Interfaces

- Python 3.8 or higher.
- TensorFlow 2.x, Keras for LSTM model development.

4.4 Communications Interfaces

- HTTPS for secure data transfer from APIs.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

- The system shall process new data and update predictions within 1 minute of data receipt.

5.2 Security Requirements

- The system shall implement standard cybersecurity measures to protect data integrity and privacy.

5.3 Software Quality Attributes

- **Reliability**: The system should perform consistently under varying conditions.
- **Usability**: The interface should be intuitive and require minimal training for users.
- **Maintainability**: Code should be well-documented and maintainable.