Team-593041

Project Documentation

VIT®
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

Time Series Analysis For Bitcoin Price Prediction Using fbProphet

By

Sarvepalli Megha Suhanth Royal (21BAI1701)

Vangala Vishruth Reddy (21BAI1648)

Rajoli Sai Anvesh Reddy (21BAI1695)

Kannepalli Venkata Sai Pranav (21BCE5670)

S.NO	Title of Figure	Page No
1	INTRODUCTION	03
2	LITERATURE SURVEY	05
3	IDEATION PHASE	08
4	REQUIREMENT ANALYSIS	16
5	PROJECT DESIGN	18
6	PROJECT PLANNING & SCHEDULING	32
7	CODING & SOLUTIONING	51
8	PERFORMANCE TESTING	72
9	RESULTS	76
10	ADVANTAGES & DISADVANTAGES	77
11	CONCLUSION	79
12	FUTURE SCOPE	79
12	APPENDIX	81

Introduction

Cryptocurrency markets, led by Bitcoin, have become a focal point for investors seeking opportunities in the dynamic landscape of digital assets. The inherent volatility of these markets poses both challenges and opportunities, making accurate price forecasting a critical aspect of informed decision-making. The Bitcoin Forecasting project is a comprehensive endeavor undertaken by Team-593041 to develop a robust predictive model using the **fbprophet** library.

Significance of Bitcoin Forecasting:

The significance of forecasting Bitcoin prices lies in its potential to empower investors, traders, and cryptocurrency enthusiasts with valuable insights. Unlike traditional financial assets, cryptocurrencies operate in a decentralized environment, influenced by various factors such as market sentiment, technological developments, regulatory changes, and macroeconomic trends. The ability to predict price movements in this complex ecosystem is instrumental in optimizing investment strategies, risk management, and capitalizing on market opportunities.

Motivation:

The motivation behind this project stems from the growing interest in cryptocurrencies as alternative assets and the increasing demand for reliable forecasting tools. Cryptocurrency markets operate 24/7, and their prices are susceptible to rapid and unpredictable fluctuations. The fbprophet library, known for its effectiveness in time series forecasting, presents an opportunity to create a tailored solution for forecasting Bitcoin prices with accuracy and efficiency.

Objectives of the Bitcoin Forecasting:

1. **Project Developing a Reliable Forecasting Model**:

Implementing the fbprophet library to create a predictive model capable of capturing Bitcoin price trends.

2. **User-Friendly Interface**:

Designing an intuitive and user-friendly interface that allows users, regardless of their expertise in data science, to interact with and benefit from the forecasting model.

3. **Real-Time Predictions**:

Enabling real-time predictions to keep users informed about the latest trends and potential price movements.

4. **Evaluation and Optimization**:

Conducting rigorous evaluations of the forecasting model's performance and implementing optimizations to enhance its accuracy and reliability.

5. **Educational Outreach:**

Providing educational content and insights derived from the forecasting model to help users understand the factors influencing Bitcoin prices.

Scope of the Bitcoin Forecasting Project:

The scope of this project extends beyond the development of a predictive model. It encompasses user engagement, continuous improvement, and educational outreach. By integrating user feedback, staying abreast of market dynamics, and facilitating a deeper understanding of Bitcoin price movements, the project aspires to contribute to the broader cryptocurrency community.

Structure of the Project Documentation:

This documentation serves as a comprehensive guide, detailing the project's inception, methodology, development stages, and future prospects. Each section provides valuable insights into the team's thought process, decision- making, and the technical aspects of implementing the Bitcoin Forecasting solution. In the following pages, we delve into the literature that informs our approach, the ideation and proposed solution, the requirements shaping our design, the intricacies of our project plan, the coding and solutioning stages, and the outcomes achieved. We analyze the advantages and disadvantages of our solution, draw conclusions from our findings, and outline the exciting future scope of the Bitcoin Forecasting project. Let this documentation be a testament to our commitment to excellence, innovation, and the pursuit of knowledge in the dynamic world of cryptocurrency forecasting.

Literature Survey

The literature survey conducted by Team-592390 explores existing research and methodologies related to cryptocurrency forecasting, with a focus on Bitcoin. Understanding the landscape of predictive modeling in the realm of digital assets is crucial for informing our approach and leveraging the latest advancements in the field.

Current State of Cryptocurrency Forecasting:

The field of cryptocurrency forecasting has witnessed significant growth in recent years, driven by the surge in interest and investment in digital assets. Researchers and practitioners alike have explored various approaches to predict cryptocurrency prices, given their unique characteristics and the absence of centralized regulatory control.

1. Machine Learning Techniques:

- Existing literature reveals a predominant use of machine learning techniques for cryptocurrency forecasting. Commonly employed algorithms include linear regression, decision trees, support vector machines, and ensemble methods.

2. Time Series Forecasting:

- Time series forecasting has emerged as a key methodology due to the sequential nature of cryptocurrency price data. Autoregressive Integrated Moving Average (ARIMA) models and Exponential Smoothing methods have been applied to capture trends and patterns.

3. Deep Learning Models:

- With the rise of deep learning, especially in the field of natural language processing and image recognition, researchers have explored the application of deep neural networks for cryptocurrency price prediction. Long Short-Term Memory (LSTM) networks and Recurrent Neural Networks (RNNs) are commonly used for capturing temporal dependencies.

Challenges in Cryptocurrency Forecasting

While advancements have been made, several challenges persist in the domain of cryptocurrency forecasting, influencing our approach in the Bitcoin Forecasting project:

1. Volatility and Non-Linearity:

- Cryptocurrency markets are characterized by high volatility and non- linear price movements. Traditional forecasting models may struggle to capture abrupt changes and sudden spikes.

2. External Factors:

- The influence of external factors such as regulatory developments, macroeconomic trends, and market sentiment adds complexity to forecasting models. Integrating these factors into predictive models remains a challenge.

3. Data Quality and Integrity:

- Cryptocurrency price data is susceptible to manipulation, and ensuring the quality and integrity of historical data is crucial for developing accurate forecasting models.

Relevance of Time Series Forecasting for Bitcoin

1. fbprophet Library:

- Our literature survey highlights the relevance of the **fbprophet** library in the context of time series forecasting. Developed by Facebook, **fbprophet** is designed to handle seasonality, holidays, and outliers, making it suitable for predicting Bitcoin prices with daily fluctuations.

2. Prophet in Cryptocurrency Research:

- Studies have shown promising results using Prophet in cryptocurrency research, with its ability to model trend changes, adapt to irregular holidays, and handle missing data.

Comparative Analysis

Our survey involved a comparative analysis of existing forecasting models, including those based on traditional machine learning algorithms, deep learning approaches, and time series forecasting methods. The evaluation criteria considered accuracy, computational efficiency, and adaptability to the unique characteristics of Bitcoin price data.

Conclusion of Literature Survey

The literature survey has provided valuable insights into the evolving landscape of cryptocurrency forecasting. It serves as the foundation for our decision to leverage the **fbprophet** library, a powerful tool in time series forecasting, for the development of our Bitcoin Forecasting model. In the following sections, we articulate our proposed solution, detailing the methodology, architecture, and implementation plan derived from the literature survey.

Ideation Phase

Empathize & Discover

Date	23rd of October,2023
Team ID	Team-593041
Project Name	Time Series Analysis For Bitcoin Price Prediction
Maximum Marks	5 Marks

Empathy Map Canvas:

An empathy map is a simple, easy-to-digest visual that captures knowledge about a user's behaviours and attitudes.

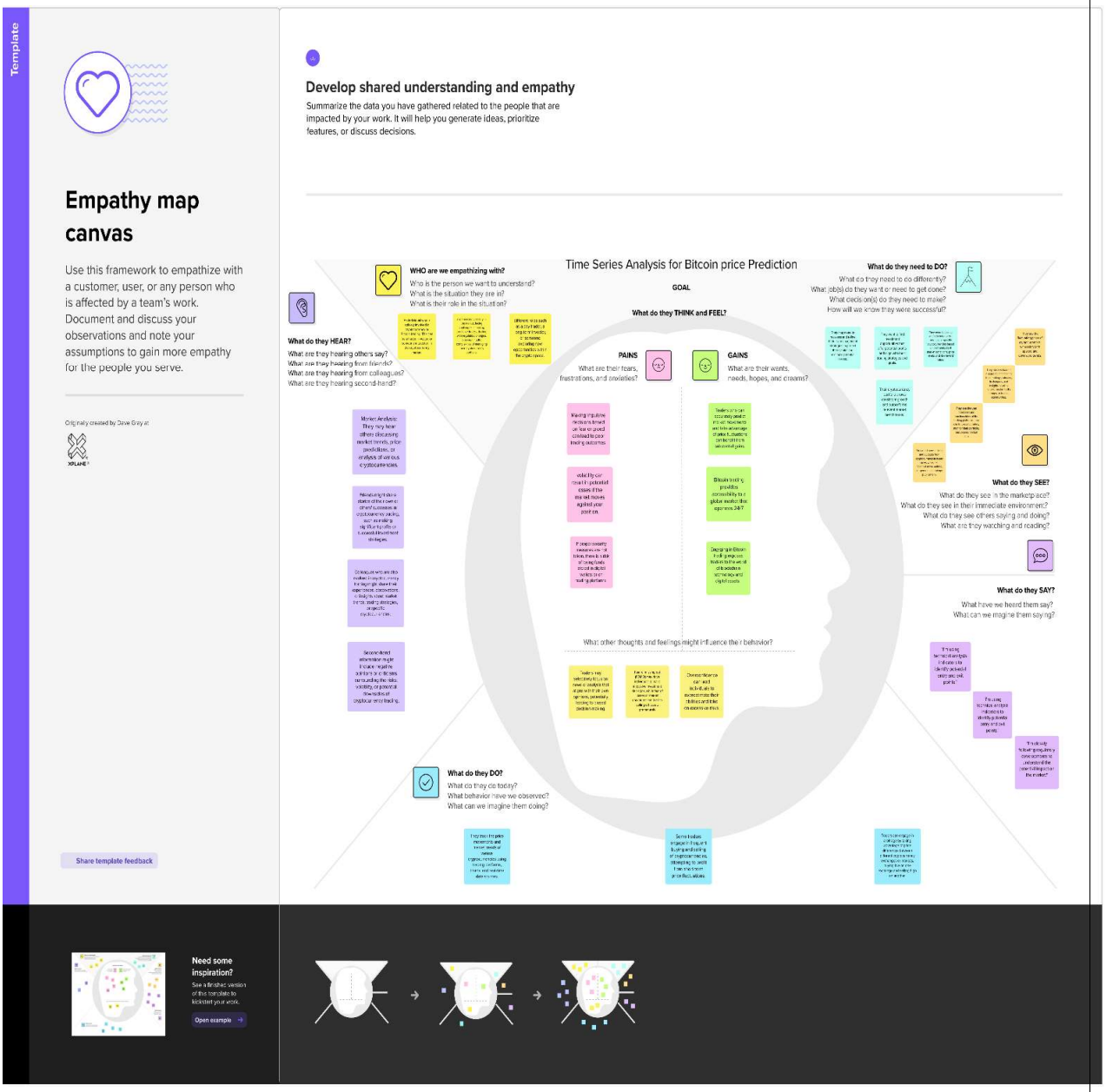
It is a useful tool to helps teams better understand their users.

Creating an effective solution requires understanding the true problem and the person who is experiencing it. The exercise of creating the map helps participants consider things from the user's perspective along with his or her goals and challenges.

Access file here:

https://app.mural.co/t/anvesh9915/m/anvesh9915/1700325346442/0c82250fbadb280c7e0b4ccc5c33c2e73911c7a0?sender=u331ab11a707c244f11d73375

Empathy Map:


Template
Empathy map canvas
Use this framework to empathize with a customer, user, or any person who is affected by a team's work. Document and discuss your observations and note your assumptions to gain more empathy for the people you serve.
Originally created by Dave Gray at
XPLANE
Share template feedback
Develop shared understanding and empathy
Summarize the data you have gathered related to the people that are impacted by your work. It will help you generate ideas, prioritize features, or discuss decisions.
Time Series Analysis for Bitcoin price Prediction
GOAL
WHO are we empathizing with?
Who is the person we want to understand?
What is the situation they are in?
What is their role in the situation?
What do they HEAR?
What are they hearing others say?
What are they hearing from friends?
What are they hearing from colleagues?
What are they hearing second-hand?
What do they THINK and FEEL?
PAINS
What are their fears, frustrations, and anxieties?
GAINS
What are their wants, needs, hopes, and dreams?
What do they need to DO?
What do they need to do differently?
What job(s) do they want or need to get done?
What decision(s) do they need to make?
How will we know they were successful?
What do they SEE?
What do they see in the marketplace?
What do they see in their immediate environment?
What do they see others saying and doing?
What are they watching and reading?
What do they SAY?
What have we heard them say?
What can we imagine them saying?
What other thoughts and feelings might influence their behavior?
What do they DO?
What do they do today?
What behavior have we observed?
What can we imagine them doing?
Need some inspiration?
See a finished version of this template to kickstart your work.
Open example

Ideation Phase
Brainstorm & Idea Prioritization Template

Date	23rd October,2023
Team ID	Team-593041
Project Name	Time Series Analysis for Bitcoin Price Prediction
Maximum Marks	4 Marks

Project Brainstorming Template:

Welcome to the brainstorming session for our exciting Bitcoin price prediction project. This project is set to revolutionize the way we forecast Bitcoin's price by integrating cutting-edge data analysis, machine learning, and risk assessment techniques. As we embark on this journey, it's essential to gather ideas and insights from our team to ensure we create a comprehensive and robust solution. In this template, we have organized our brainstorming into five key clusters, each focusing on crucial aspects of our project. Your input, ideas, and feedback are vital to help us navigate the intricate world of cryptocurrency and provide the best possible predictions for our users. Together, we will shape a platform that not only predicts Bitcoin's price trends but also empowers users to make informed investment decisions, manage risks, and explore broader cryptocurrency market trends. Let's dive into each cluster, generating innovative ideas and thoughtful discussions to bring our project to life.

Reference:
https://app.mural.co/t/anvesh9915/m/anvesh9915/1700560417373/85d680bb2971f2db20f1fb489efd034f627fbaf7?sender=u331ab11a707c244f11d73375

Template

Brainstorm & idea prioritization on Time Series Analysis For Bitcoin Price Prediction

Need some inspiration?

See a finished version of this template to kickstart your work.

Open example →

Define your problem statement

PROBLEM

How can we leverage the FbProphet forecasting model to accurately predict the highly volatile price trends of Bitcoin, accounting for factors that influence its price, and provide valuable insights for investors in this dynamic cryptocurrency market?

2

Brainstorm

CRYPTOINSIGHTS: ENHANCING BITCOIN PRICE PREDICTION AND RISK ASSESSMENT

MEGHA SUHANTH

SENTIMENT ANALYSIS INTEGRATION

considering the integration of sentiment analysis into our model by analyzing news articles, social media posts, and forums that are relevant to Bitcoin. Incorporating this analysis of public sentiment can be highly beneficial as it has the potential to provide valuable insights into sudden price movements and investor sentiment. This addition would enhance our understanding of the market and offer a more comprehensive explanation of these dynamics.

Machine Learning Model Comparison

It would be beneficial to explore different machine learning models, including LSTM neural networks or ARIMA, alongside FbProphet, and evaluate their performance in predicting Bitcoin's price. By conducting this comparative analysis, we can determine which approach is the most effective for our project. This experimentation with diverse models will enable us to make an informed decision about the best methodology to employ.

VISHRUTH

EXTERNAL SOURCES

exploring the incorporation of external data sources, such as macroeconomic indicators, regulatory changes, and global events, into your model. This integration can be crucial as these factors have the potential to significantly influence the price of Bitcoin. By including these variables in your analysis, you may enhance the accuracy of your predictions and gain a more comprehensive understanding of the market dynamics.

User-Friendly Interface

Design an interface that is visually appealing and intuitive, providing users with a seamless experience. Incorporate user-friendly visualizations and tools that enable users to easily comprehend the information presented. Additionally, provide customization options for users to personalize their predictions and portfolios according to their preferences. By focusing on these aspects, you can ensure that the platform is user-centric and enhances user engagement.

PRANAV

PORTFOLIO MANAGEMENT TOOL

Develop an all-encompassing platform that goes beyond Bitcoin price predictions and offers users assistance in effectively managing their cryptocurrency portfolios. This comprehensive platform should encompass features that facilitate portfolio diversification, risk assessment, and automated trading strategies. By providing these functionalities, users can optimize their portfolio management, minimize risk exposure, and potentially enhance their trading outcomes. The goal is to develop a platform that empowers users with the tools and resources needed to make informed decisions and achieve their investment objectives in the cryptocurrency market.

Risk Assessment and Mitigation

Create a risk assessment module that assesses the potential risks involved in Bitcoin investments, taking into account factors such as market volatility, regulatory changes, and security threats. This module will analyze the risk landscape associated with Bitcoin and provide users with recommendations on effective risk mitigation strategies. By incorporating this feature, users can make more informed investment decisions, understand the potential risks involved, and take appropriate measures to protect their investments.

ANVESH

REAL TIME PREDICTIONS

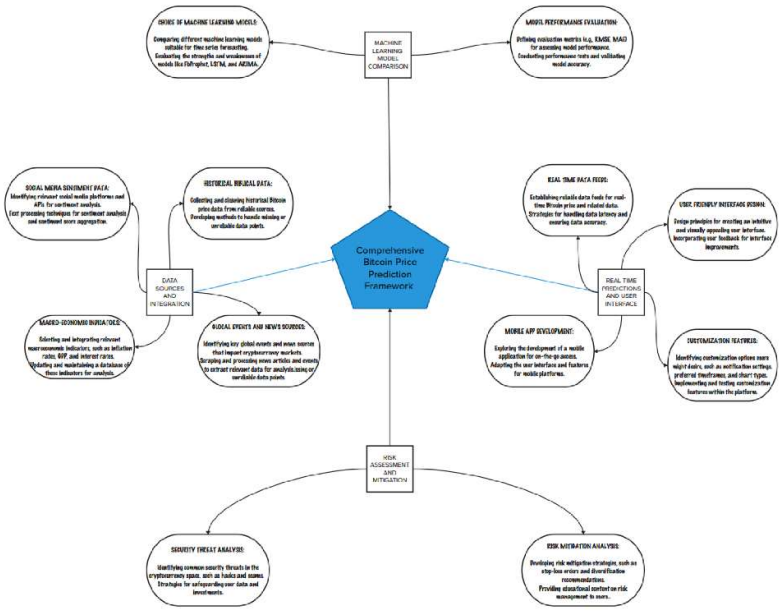
Expand the project to offer real-time or near-real-time predictions, empowering users to make timely decisions in the rapidly evolving cryptocurrency market. This expansion can involve the implementation of automated data feeds and regular updates to the prediction model. By establishing a system that continuously incorporates the latest data and adjusts the model accordingly, users can access up-to-date insights and predictions, enabling them to respond promptly to market changes. This real-time functionality enhances the platform's value by providing users with the most current information for informed decision-making in the fast-paced cryptocurrency landscape.

Cryptocurrency Market Trends Analysis

Expand our project to examine and forecast trends in the wider cryptocurrency market, encompassing not only Bitcoin but also other digital currencies. This enhancement could offer users a comprehensive understanding of how Bitcoin behaves within the broader market landscape.

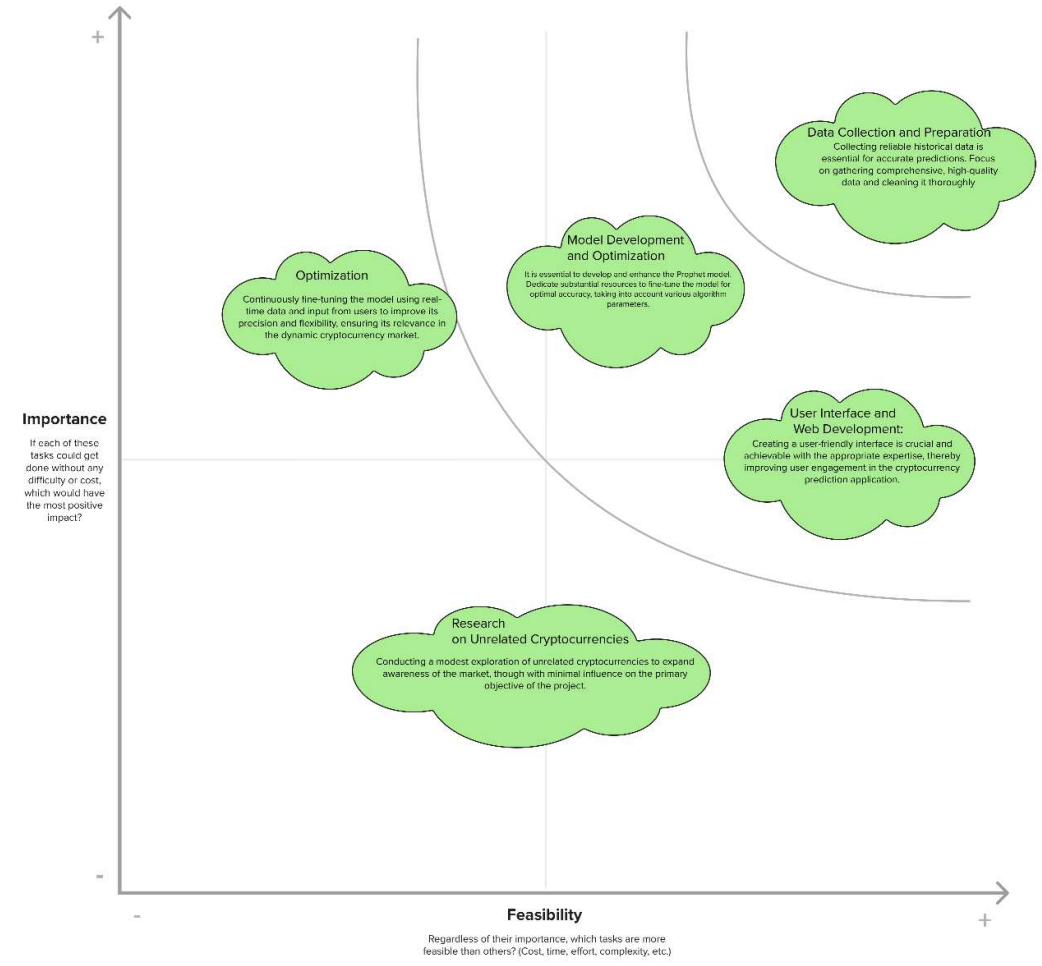
3

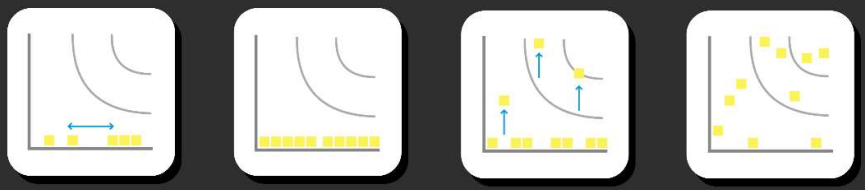
Group ideas



4

Prioritize





Requirement Analysis

System Conditions:

Non-Functional Conditions:

1. **Speed**: The system should process given input efficiently to provide timely forecasts, critical for adapting to the dynamic nature of cryptocurrency markets.

2. **Ease of Use**: The software's user-friendly interface is pivotal, ensuring seamless interaction for both novice and experienced users.

3. **Trustability**: A low failure rate is essential, establishing the software's reliability in delivering accurate Bitcoin predictions.

4. **Portability**: The software's ease of deployment on various systems enhances its accessibility and practicality.

Specific Conditions:

1. **User Interfaces**: External users, referred to as "guests," should effortlessly interact with the software for tasks like indexing and searching, promoting a positive user experience.

2. **Hardware Interfaces**: Seamless integration with specific guest computers, potentially laptops with wireless LAN for internet connections, enhances accessibility.

3. **Software Interfaces**: Compatibility with various Windows operating system versions ensures widespread usability.

4. **Performance Conditions**: Recommending machines with at least Pentium 4 emphasizes optimal performance for users.

Software Specifications:

Software conditions, focusing on defining resource prerequisites for optimal functioning:

1. The system should not only be compatible with the host operating system but also demonstrate accuracy in Bitcoin forecasting.

2. Outperforming the existing system's capabilities ensures the software's superiority and reliability.

Tackle Specifications:

Common conditions for operating systems include:

1. **Armature Compatibility:** While armature-independent systems exist, adaptation for new armatures is frequently required.

Tackle Committee List:

Accompanying the tackle conditions list, an HCL (Hardware Compatibility List) should include:

1. Tested and compatible hardware for the specific operating system ensures a seamless user experience.

2. Identification of potentially incompatible hardware allows users to make informed decisions.

CPU Specifications:

The central processing unit (CPU) is a fundamental system requirement:

1. **Processor Power**: The definition of processing power through the model and clock speed of the CPU is essential for efficient Bitcoin prediction.

2. **Additional Features:** Consideration of other features impacting speed and power, such as machine speed, cache, and MIPS, ensures a comprehensive understanding of system capabilities.

This extensive requirement analysis establishes the foundational conditions for the Bitcoin Forecasting project. It emphasizes efficiency, usability, and compatibility with various hardware and software environments, crucial for accurate and timely Bitcoin predictions in the dynamic cryptocurrency landscape.

Project Design Phase-I
Proposed Solution

Date	27 October 2023
Team ID	Team-593041
Project Name	Time Series Analysis for Bitcoin Price Prediction Using Prophet
Maximum Marks	2 Marks

Proposed Solution Template:

Project team shall fill the following information in proposed solution template.

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	The problem to be solved is the prediction of Bitcoin's price, which is highly volatile and influenced by various factors, using a Time Series Analysis approach with the FbProphet model. The goal is to provide accurate and reliable price forecasts for Bitcoin, enabling investors and traders to make informed decisions in a rapidly changing cryptocurrency market
2.	Idea / Solution description	The proposed solution involves building a predictive model using FbProphet, a time series forecasting tool, to analyze historical Bitcoin price data and forecast its future price trends. This model will consider the inherent volatility and various influencing factors in the cryptocurrency market to provide forecasts for Bitcoin prices. By leveraging FbProphet, which is well-suited for handling time series data, the solution aims to offer more accurate predictions compared to traditional methods.

3.	Novelty / Uniqueness	The novelty of this solution lies in itsapplication of FbProphet to the cryptocurrency market, specifically Bitcoin price prediction. FbProphet is known for itsability to handle seasonality and holiday effects in time series data, which makes it uniquely suited for modeling the price trends of Bitcoin, a highly volatile asset influenced by a range of external factors.
4.	Social Impact / CustomerSatisfaction	This solution has a significant social impactas it empowers cryptocurrency investors and enthusiasts to make more informed decisions
		about Bitcoin investments. Accurate price predictions can help users maximize their returns and minimize risks in a market known for its high volatility. Customer satisfaction is achieved by providing users with a reliable tool for price forecasting and increasing their confidence in the cryptocurrency market.
5.	Business Model (Revenue Model)	The business model for this solution couldinvolve several revenue streams: **Subscription-based model**: Charging usersa subscription fee to access premium features or more accurate predictions. **Freemium model**: Offering basic predictionsfor free and charging for advanced or real- time forecasting services. **Data licensing**: Selling historical and real- time Bitcoin price data to other businessesand researchers. **Consultation and advisory services**: Providing personalized advice and consultation on Bitcoin investments for afee.

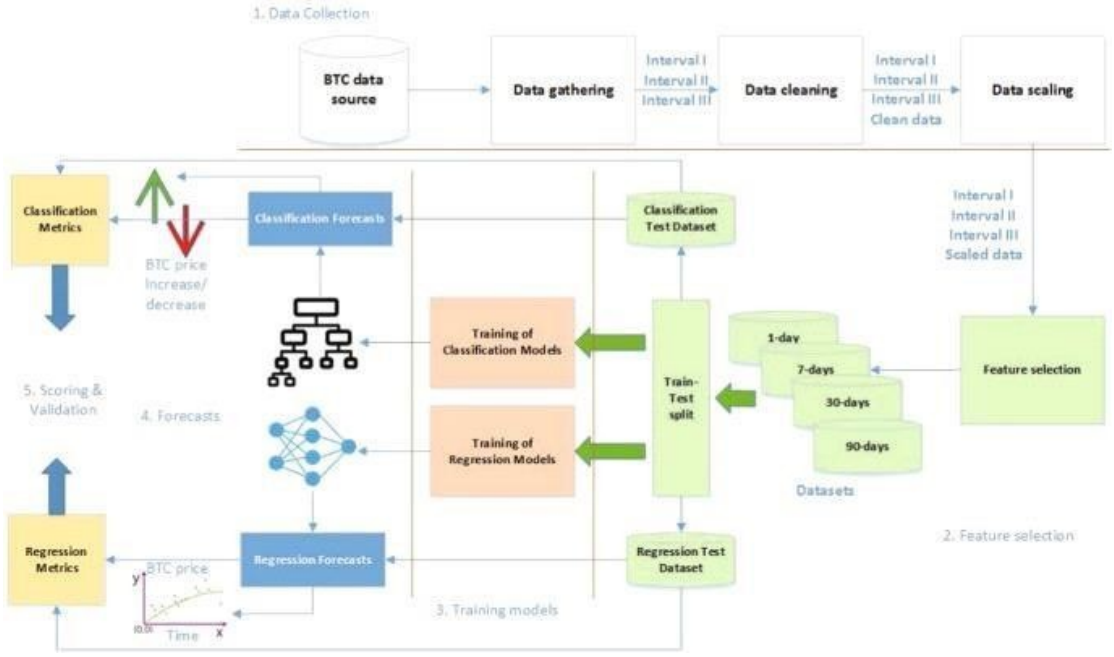
6.	Scalability of the Solution	The solution's scalability can be achieved byoptimizing the prediction model and infrastructure to handle increasing data volumes and users. Additionally, the solution can be scaled by expanding its coverage to other cryptocurrencies and financial markets, making it a comprehensive tool for price forecasting inthe digital asset space. This scalability can be achieved by investing in robust cloud infrastructure, data management, and usersupport systems to accommodate growing demand

Project Design Phase

Solution Architecture

Date	27th Oct, 2023
Team members	Sarvepalli Megha Suhanth Royal Rajoli Sai Anvesh Reddy Vangala Vishruth Reddy Kannepalli Venkata Sai Pranav
Project Name	Time Series Analysis For Bitcoin Price Prediction Using Prophet
Maximum Marks	5 Marks
Team id	593041

Solution Architecture



1. Data collection and preparation:

Data collection is the process of compiling pertinent information from a variety of sources, including weather satellites, public weather APIs, and meteorological stations. Features like temperature, humidity, wind speed, atmospheric pressure, cloud cover, and rainfall records from the past should all be included in the data.
Data preparation is the process of ensuring formatting uniformity, resolving missing values, and eliminating outliers from the acquired data. Feature engineering is another possibility. This is the process of extracting new valuable features from the raw data by combining domain expertise or aggregating data across predetermined time periods.

2. Model selection and training:

The particular challenge and the dataset's properties influence the choice of model. Neural networks, gradient boosting, decision trees, random forests, and linear regression are examples of popular machine learning algorithms for rainfall prediction. The ability to handle complicated relationships in the data, interpretability, scalability, and other criteria may all be taken into account while choosing an algorithm. The model is instantiated and trained on the prepared dataset when the algorithm has been chosen. The model is fed input features and matching measurements of rainfall during the training phase, and its internal parameters are adjusted to reduce the prediction error.

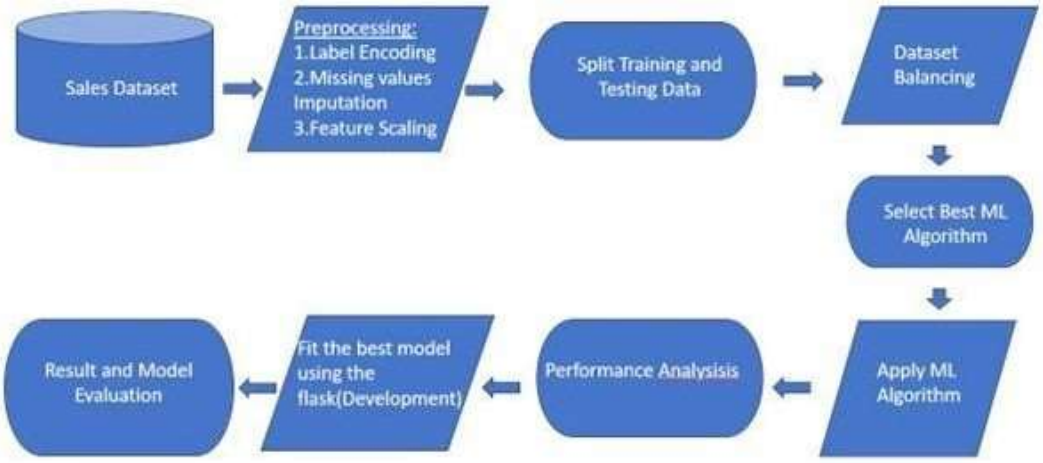
3. Model evaluation:

To evaluate the effectiveness and capacity for generalization of the trained model, model evaluation is essential. The mean squared error (MSE), correlation coefficient, mean absolute error (MAE), and root mean square error (RMSE) are examples of evaluation metrics for rainfall prediction. To offer an unbiased estimate of the model's performance, it is tested on a held-out test set that contains data that the model has not seen during training..

4. Model deployment:

Model deployment involves creating an application or system that integrates the trained model to provide real-time rainfall predictions. This can be done by developing a user interface (UI) where users can input relevant weather features, and the model generates the corresponding rainfall prediction.
The application can be deployed on a local server or a cloud computing platform to ensure accessibility and scalability. It should be designed to handle user requests efficiently and provide fast predictions based on the trained model. Regular updates and maintenance may be required to incorporate new data and improve the model's performance over time.
Solution Architecture Diagram:

5. The solution architecture diagram



depicts the components of the system and the data flow in a visual way. It offers a high-level summary of the procedures used to gather, handle, and train data for the model. Additionally, it shows how to use the trained model in a system or application for real-time rainfall prediction.

Reference link:

https://www.researchgate.net/publication/342692245_Time-series_forecasting_of_Bitcoin_prices_using_high-dimensional_features_a_machine_learning_approach

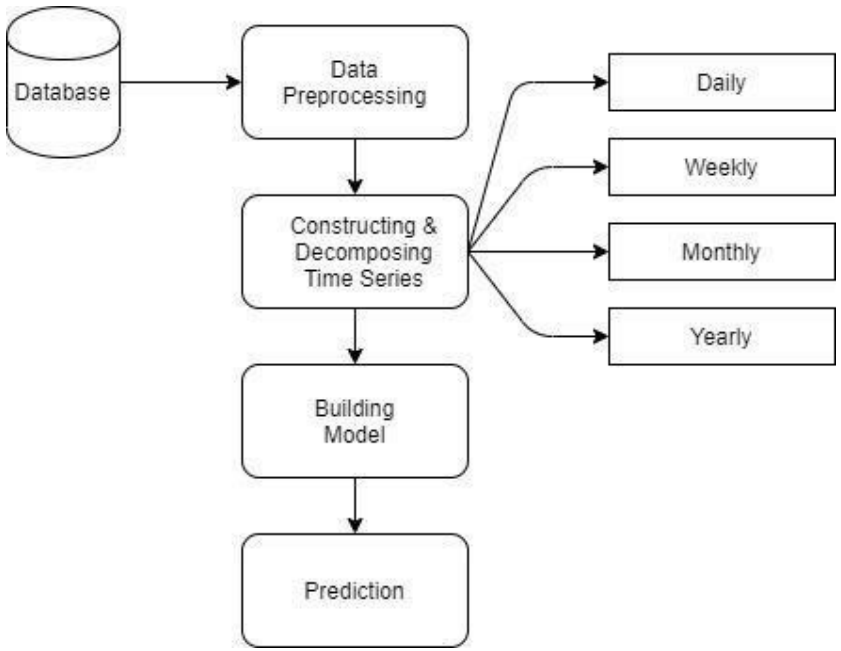
Project Design Phase-II

Data Flow Diagram & User Stories

Date	23-10-2023
Team ID	593041
Project Name	Time Series Analysis for Bitcoin Price Prediction Using Prophet
Maximum Marks	4 Marks

Data Flow Diagrams:

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is stored.


Database
Data Preprocessing
Constructing & Decomposing Time Series
Daily
Weekly
Monthly
Yearly
Building Model
Prediction

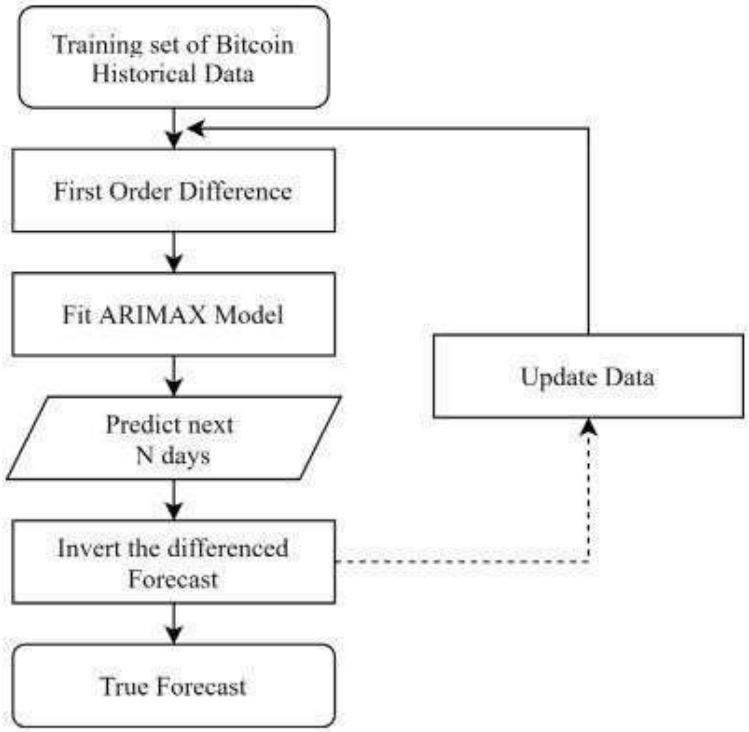


Figure 3. ARIMAX Model for BTC Prediction

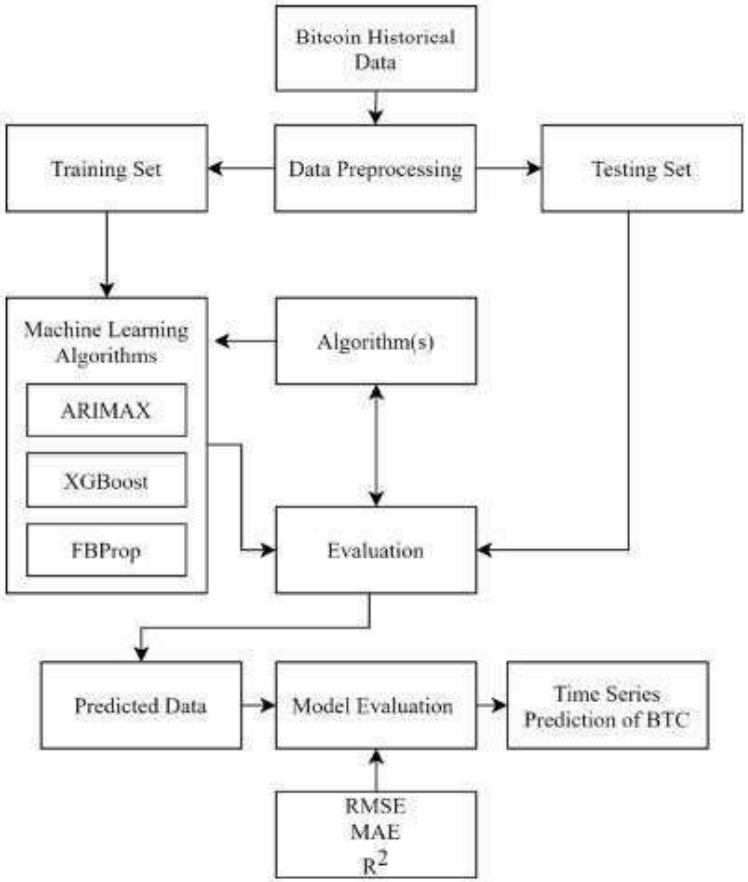


Figure 2. Basic Methodology Flowchart

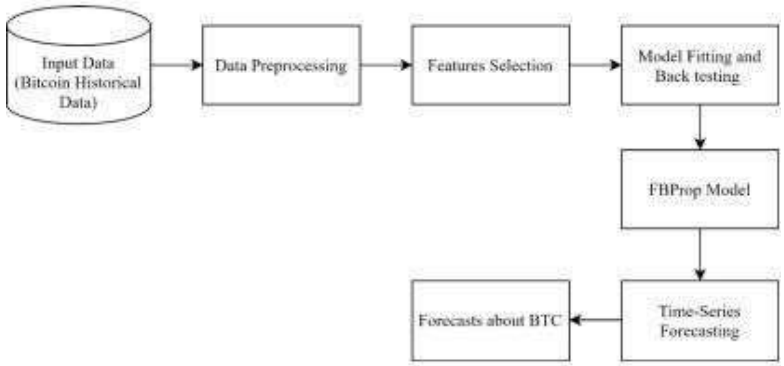


Figure 6. FBProp Algorithm for BTC

User Stories

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance Criteria	Priority	Release
Crypto Investor	Data Acquisition	USN-1	As a crypto investor, I want to collect historical Bitcoin price data for analysis and forecasting using Facebook Prophet.	Data collection should include Bitcoin price records from multiple sources and be stored in a structuredformat.	High	Sprint 1
Data Analyst	Data Preprocessing	USN-2	I want to preprocess the collected Bitcoin price data by cleaning, normalizing, and structuring it for time series analysis with Prophet.	Data should be cleaned and transformed into a suitable formatfor modelling.	High	Sprint 1
Data Scientist	Model Development and Training	USN-3	I want to select the Prophet model andtrain it using the pre-processed Bitcoinprice dataset.	The model should be fine-tuned,and training results should meet performance criteria.	High	Sprint 2

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance Criteria	Priority	Release
Application Developer	Bitcoin Price Prediction and Visualization	USN-4	I want to deploy the trained Prophet model as an API and integrate it into a user-friendly application for Bitcoin price forecasting.	To interpret the results and also helps to decisions.	High	Sprint 2
Front End Developer	Front end Development	USN-5	As a Front-End Developer, I want to implement interactive charts on the user interface, so users can visualize both predicted and actual price dynamically.	The application should provide an easy to-use interface for users to input their requirements.	High	Sprint 3
Quality Assurance	Performance Evaluation and Model Refinement	USN-6	I want to rigorously test the system, evaluate the effectiveness of predictions,and ensure high reliability and accuracy.	Testing should cover a range of scenarios, and the system should meet predefined accuracy standards.	Medium	Sprint 3

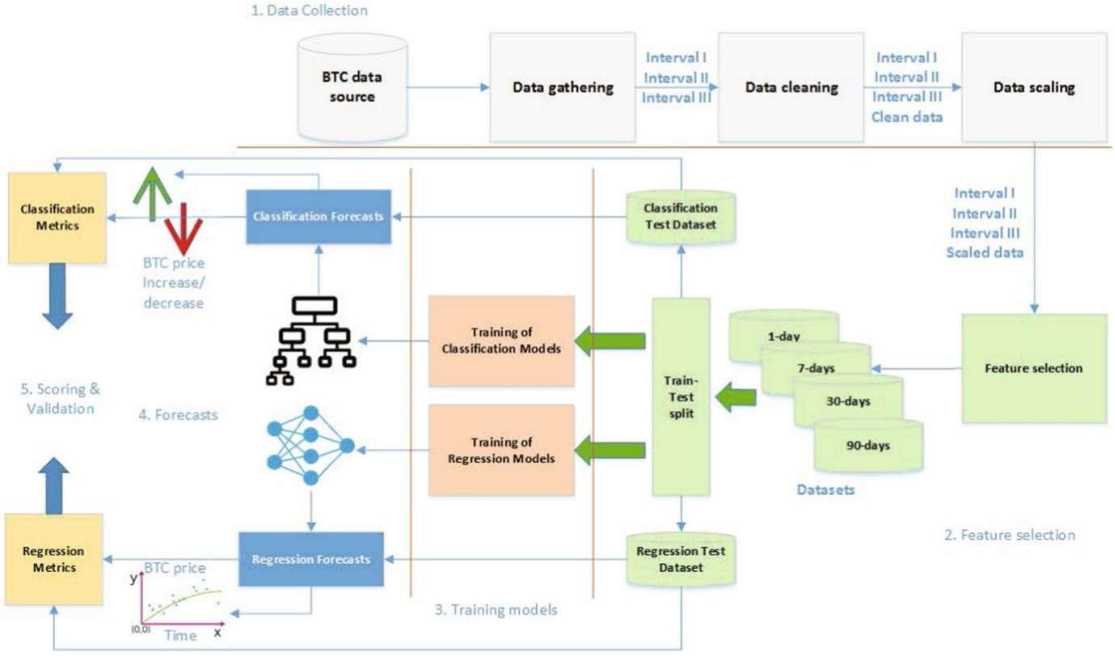
Project Planning Phase

Technology Stack

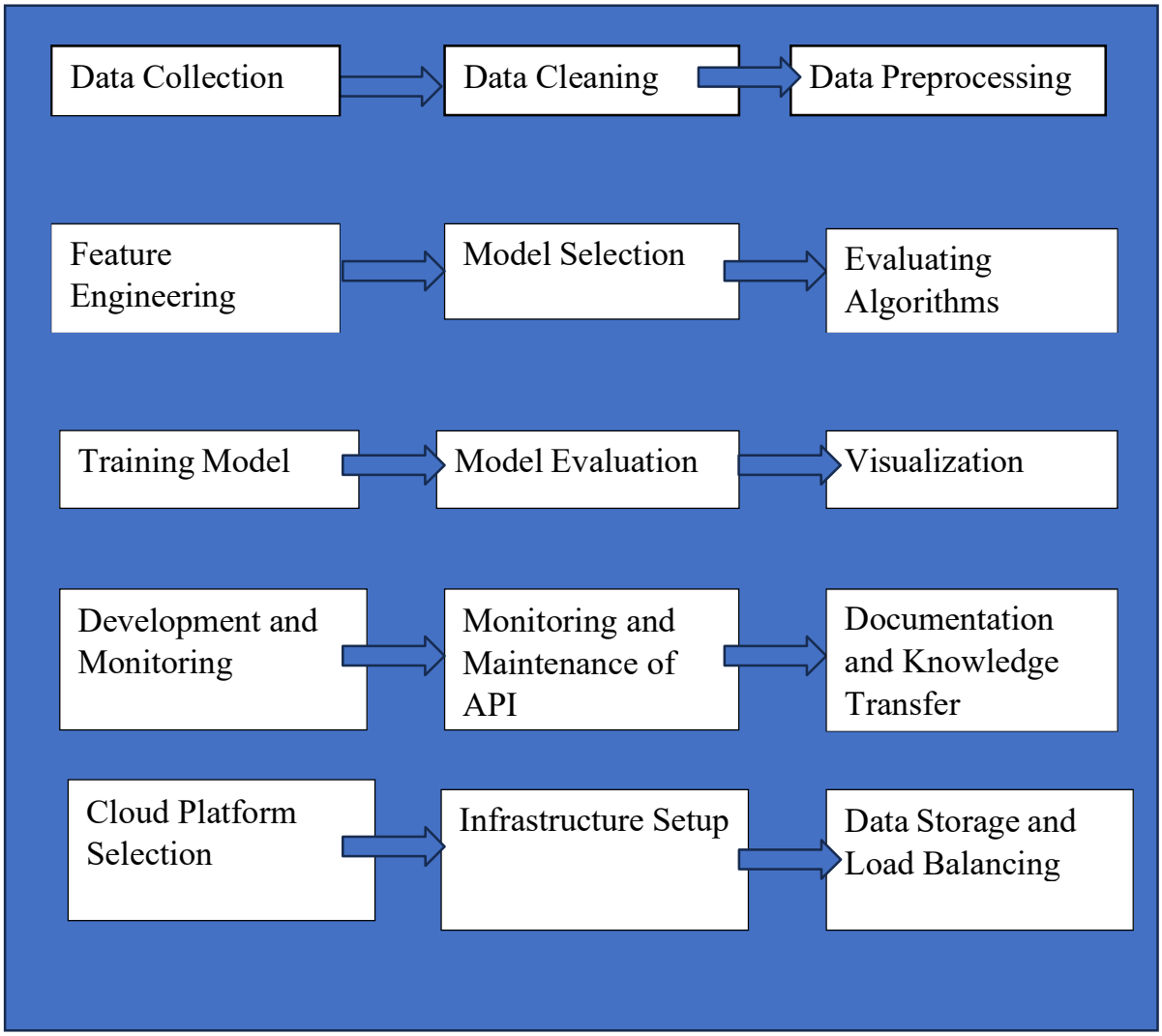
Date	01st November 2023
Team Id	Team-593041
Project Name	Time Series Analysis For BitcoinPrice Prediction Using Prophet
Maximum Marks	4
Team Members	Sarvepalli Megha Suhanth Vangala Vishruth Reddy Rajoli Sai Anvesh Reddy Kannepalli Venkata Sai Pranav

Technical Architecture

The Deliverable shall include the architectural diagram as below and the information as per the table1 & table.



Basic Phases of Technical Architecture:



These sub-phases provide a detailed breakdown of the main phases, ensuring a systematic and organized approach to the development, deployment, and maintenance of the Time Series Analysis application.

Table 1: Components & Technologies

Component	Description	Technologies
User Interaction	Interface for user interaction with the application along with creating an user friendly interface	Web-based UI (HTML, CSS, JavaScript) or mobile app (React Native, Swift/Android)
Application Logic-1	Core logic responsible for handling user requests	Python, Flask, FlaskAPI, Node.js(Express).
Data Collection	Data collection encompasses sourcing historical Bitcoin price data from exchanges via APIs, crucial for Time Series Analysis using Prophet	Integrate Python, Prophet, and cloud infrastructure for accurate predictions.
Data Input	Handling and processing user- provided input data	Forms, file uploads, APIs, or command-line input.
External API	Integration with external data sources or services.	RESTFUL APIs(Bitcoin API, Binance API)
Cloud Database	Storage and management of structured da	Amazon RDS, Google Cloud SQL, or Azure SQL.
Model Integration	Interface for integrating with FbProphet Model	RESTful API endpoints (Flask, FastAPI,JSON,XML
API Model Deployment	Responsible for deploying machine learning models as APIs, enabling real-time predictions and external interaction. It ensures model accessibility and scalability	Docker, Kubernetes, or serverless (AWS Lambda).
Machine Learning Model	The predictive model using Fb Prophet for bitcoin forecasting	Scikit-Learn, TensorFlow, PyTorch, XGBoost, fb Prophet

Data Preprocessing	Data preparation and feature engineering.	Pandas, NumPy, scikit-learn, or custom scripts.
Model Deployment	Hosting and serving the machine learning model.	Flask, FastAPI, TensorFlow Serving, or AWS Sagemaker.
Infrastructure (Server/Cloud)	Underlying cloud infrastructure and resources	AWS, Google Cloud, Azure, or on-premises servers like Local, Cloud Foundry, Kubernetes etc.

Table 2: Application Characteristics:

Component	Description	Technologies
Open-Source Frameworks	Leveraging open-source frameworks for model development and deployment guarantees cost-efficiency and flexibility, streamlining user access to Bitcoin price predictions, especially during market fluctuations and high volatility periods, enhancing accuracy and reliability in the Time Series Analysis project for Bitcoin Price Prediction using Prophet.	- Scikit-Learn, TensorFlow, PyTorch for model development. - Flask or FastAPI for API deployment. - Kubernetes for container orchestration. - Jupyter Notebook for model prototyping and development
Security Implementations	Implementations include robust encryption, authentication, and access controls to safeguard sensitive data in the Time Series Analysis project for Bitcoin Price Prediction using Prophet, leveraging Python, Prophet, and cloud-	- OAuth 2.0 or JWT for user authentication. - Encryption (HTTPS/SSL) for data in transit. - Role-based access control. - Regular security audits and updates. - Compliance with industry standards (e.g., GDPR)

	based security protocols for advanced protection.	
Scalable Architecture	Designing a scalable architecture that can handle growing data volumes and user demands which can manage the huge inflowof user demands assuming as a big data	- Microservices architecture for modularity and scalability. - Containerization with Docker and orchestration with Kubernetes. - Load balancers for distributing traffic. - Auto-scaling based on resource usage.
Availability	Ensuring high availability and minimal downtime for the Time Series Analysis application is vital, enabling continuous data processing and accurate Bitcoin price predictions using Prophet for optimal financial insights.	Redundancy in database and API deployment. - Geographically distributed data centers or cloud regions. - Monitoring and alerting systems (e.g., Prometheus, Grafana). - Failover mechanisms for fault tolerance.
Performance	Optimizing application performance to provide quick insights and predictions	Caching mechanisms for frequently accessed data. - Model optimization (e.g., quantization) for faster inference. - Load testing and performance tuning
User-Friendly Interface	Creating an intuitive and user friendly interface for data input, visualization, and interact	HTML, CSS, JavaScript for web-based UI. - React or similar frameworks for responsive design. - Data visualization libraries (e.g., D3.js). - User experience (UX) testing and design principles.

Interoperability and Accuracy	Ensuring seamless integration with external systems and maintaining high prediction accuracy	- RESTful API design for interoperability. - Integration with external data sources (e.g., weather data). - Continuous model monitoring and retraining for accuracy improvement. - Data preprocessing techniques to enhance model accuracy
Data Transparency	Increasing transparency in Bitcoin price analysis by providing accessible and accountable data sources, processing methods, and insights, fostering informed decision-making and trust in the Time Series Analysis project for Bitcoin Price Prediction using Prophet.	Data documentation tools. - Metadata management systems. - Data catalog solutions. - Access control mechanisms. - Data lineage and provenance tracking. - Data visualization tools. - Data governance frameworks. - Compliance and auditing tools.

Project Planning Phase

Project Planning Template (Product Backlog, Sprint Planning, Stories, Story points)

Date	01 November 2023
Team ID	Team-593041
Project Name	Time Series Analysis for Bitcoin Price Prediction Using Prophet
Maximum Marks	8 Marks

Product Backlog, Sprint Schedule, and Estimation (4 Marks)

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1,2	Data Acquisition and Preprocessing	USN-1	As a Data Scientist, I want to access reliable historical Bitcoin price data, so I can analyze and predict trends effectively.	1	High	Sarvepalli Megha Suhanth
Sprint-3,4	Model Development and Training	USN-2	As a Machine Learning Engineer, I want to create a baseline fbProphet model for Bitcoin price prediction, so I can establish a foundation for further optimization.	2	High	Vishruth Reddy
Sprint-5,6	Bitcoin Price Prediction and Visualization	USN-3	As a Product Owner, I want to visualize model predictions against actual Bitcoin prices, so I can interpret the results and make informed decisions.	3	High	Anvesh Reddy, Sai Pranav
Sprint-7,8	Performance Evaluation and Model Refinement	USN-4	As a quality assurance specialist, I want to test the model's performance rigorously and evaluate its effectiveness for Bitcoin price prediction.	5	High	Sarvepalli Megha Suhanth, Vishruth Reddy
Sprint -9,10	Front end Development	USN-5	As a Front-End Developer, I want to implement interactive charts on the user interface, so users can visualize both predicted and actual price dynamically.	5	High	Vishruth Reddy

Sprint-11,12	Deployment and Application of the Time Series Forecasting Model	USN-6	As a DevOps Engineer, I want to select an appropriate cloud service provider and set up infrastructure, so the application and model can be hosted securely and efficiently.	6	High	Anvesh Reddy, Sai Pranav, Megha Suhanth

Project Tracker, Velocity & Burndown Chart: (4 Marks)

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1,2	8	1 Day	01 Nov 2023	01 Nov 2023	9	01 Nov 2023
Sprint-3,4	8	1 Day	02 Nov 2023	02 Nov 2023	8	02 Nov 2023
Sprint-5,6	8	3 Days	03 Nov 2023	05 Nov 2023	10	05 Nov 2023
Sprint-7,8,9	12	2 Days	05 Nov 2023	06 Nov 2023	11	06 Nov 2023
Sprint-10,11,12	12	2 Days	06 Nov 2023	08 Nov 2023	5	08 Nov 2023

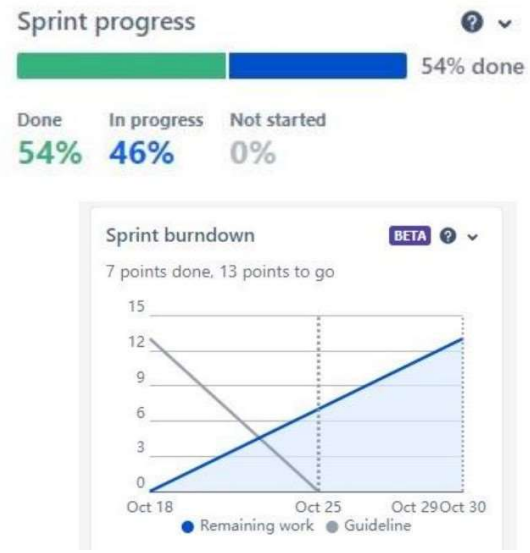
Velocity:

The team velocity of the 09-day sprint duration is:

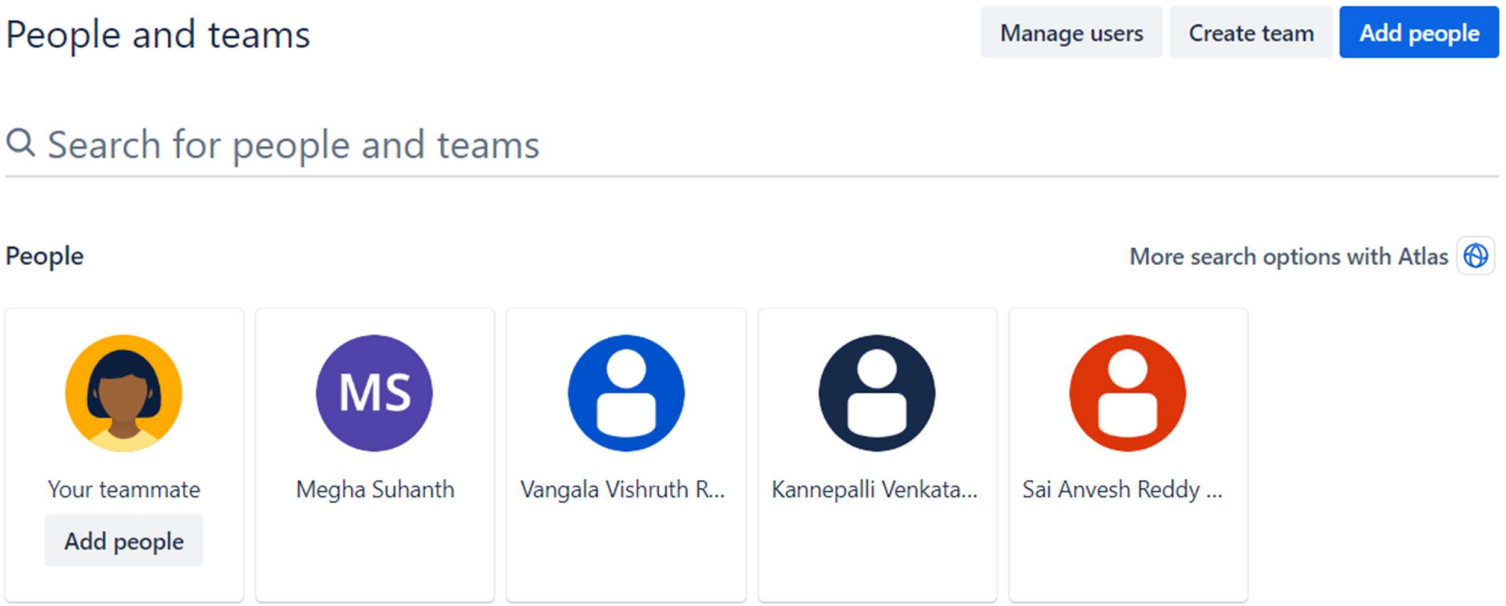
Velocity = (9+8+10+11+5) / 5 = 43 / 5 = 8.6

Average Velocity = *Sprint duration/velocity = 9/8.6* ***= 1.04***

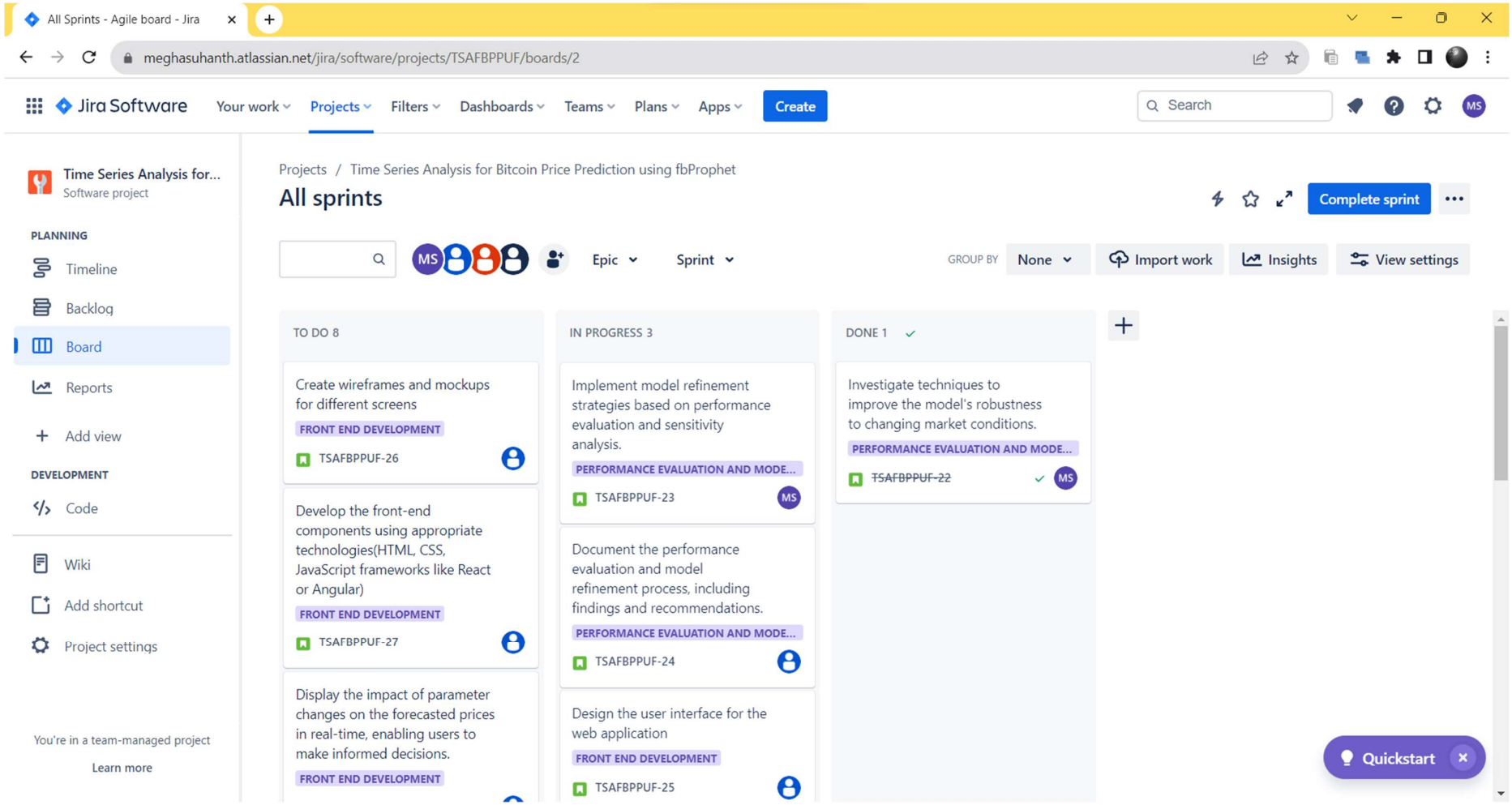
Burndown Chart: A burn down chart is a graphical representation of work left to do versus time. It is often used in agile software development methodologies such as Scrum. However, burn down charts can be applied to any project containing measurable progress over time.



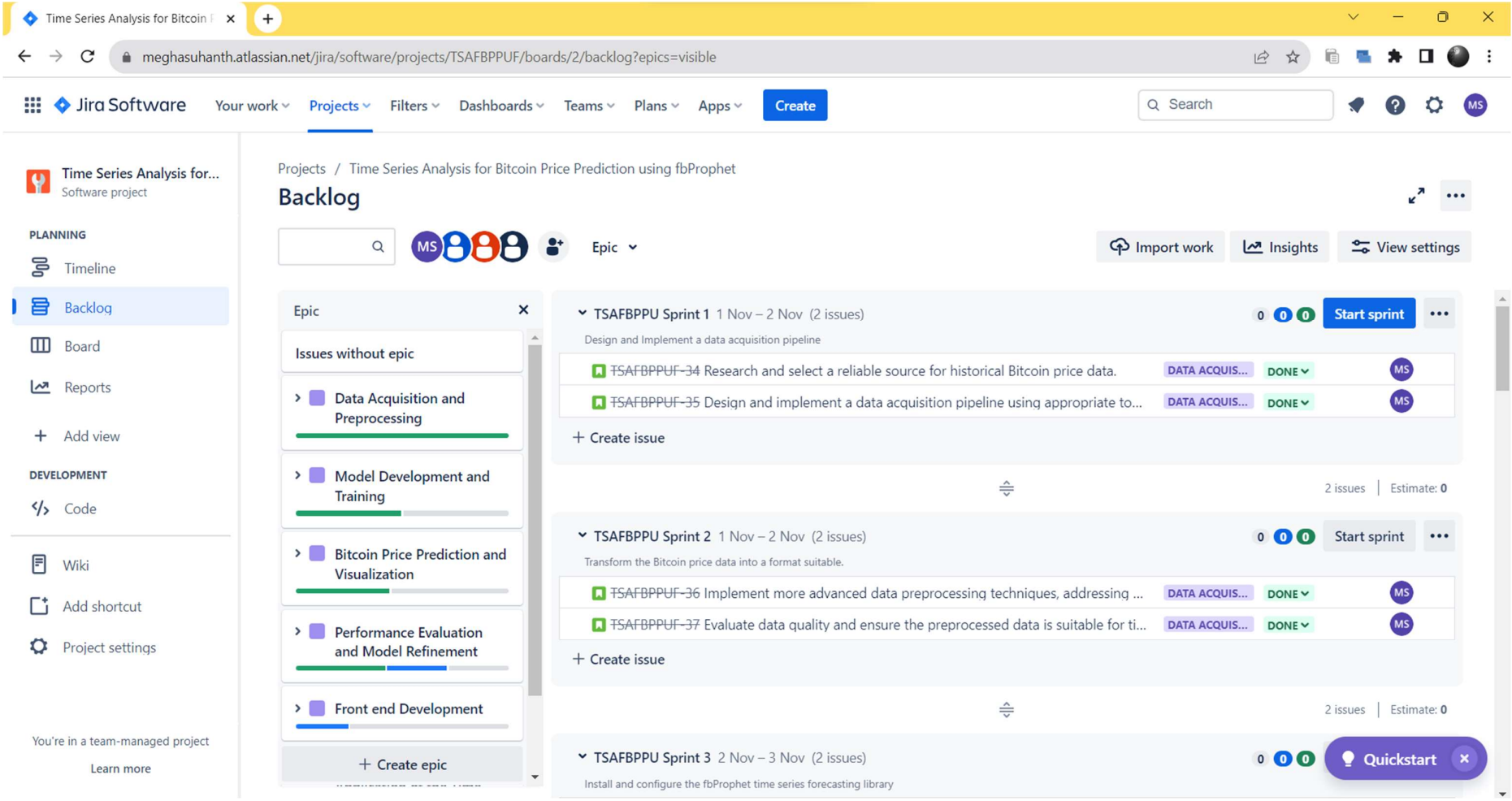
Teammates:

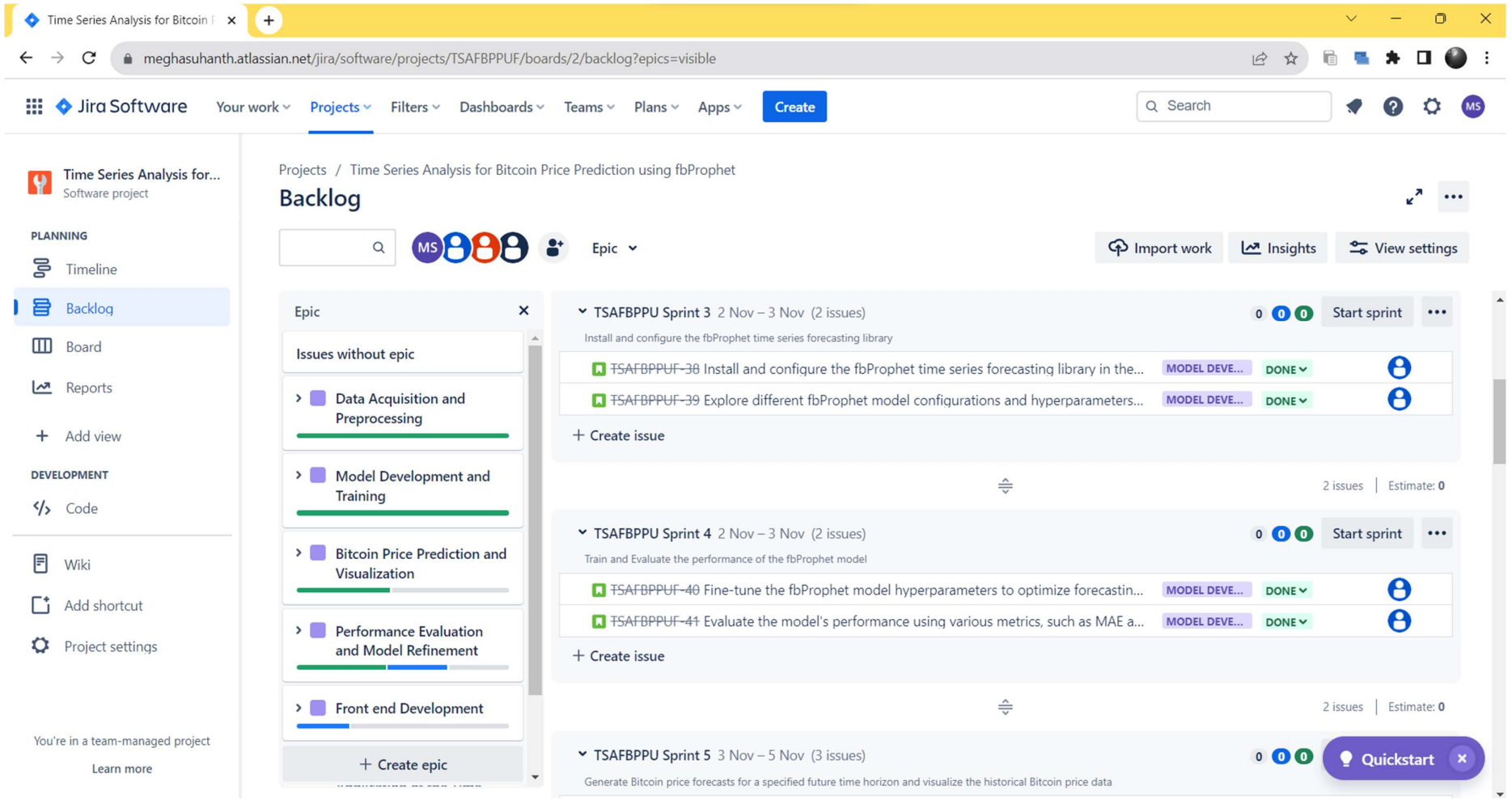

People and teams
Manage users
Create team
Add people
Search for people and teams
People
More search options with Atlas
Your teammate
Add people
MS
Megha Suhanth
Vangala Vishruth R...
Kannepalli Venkata...
Sai Anvesh Reddy ...

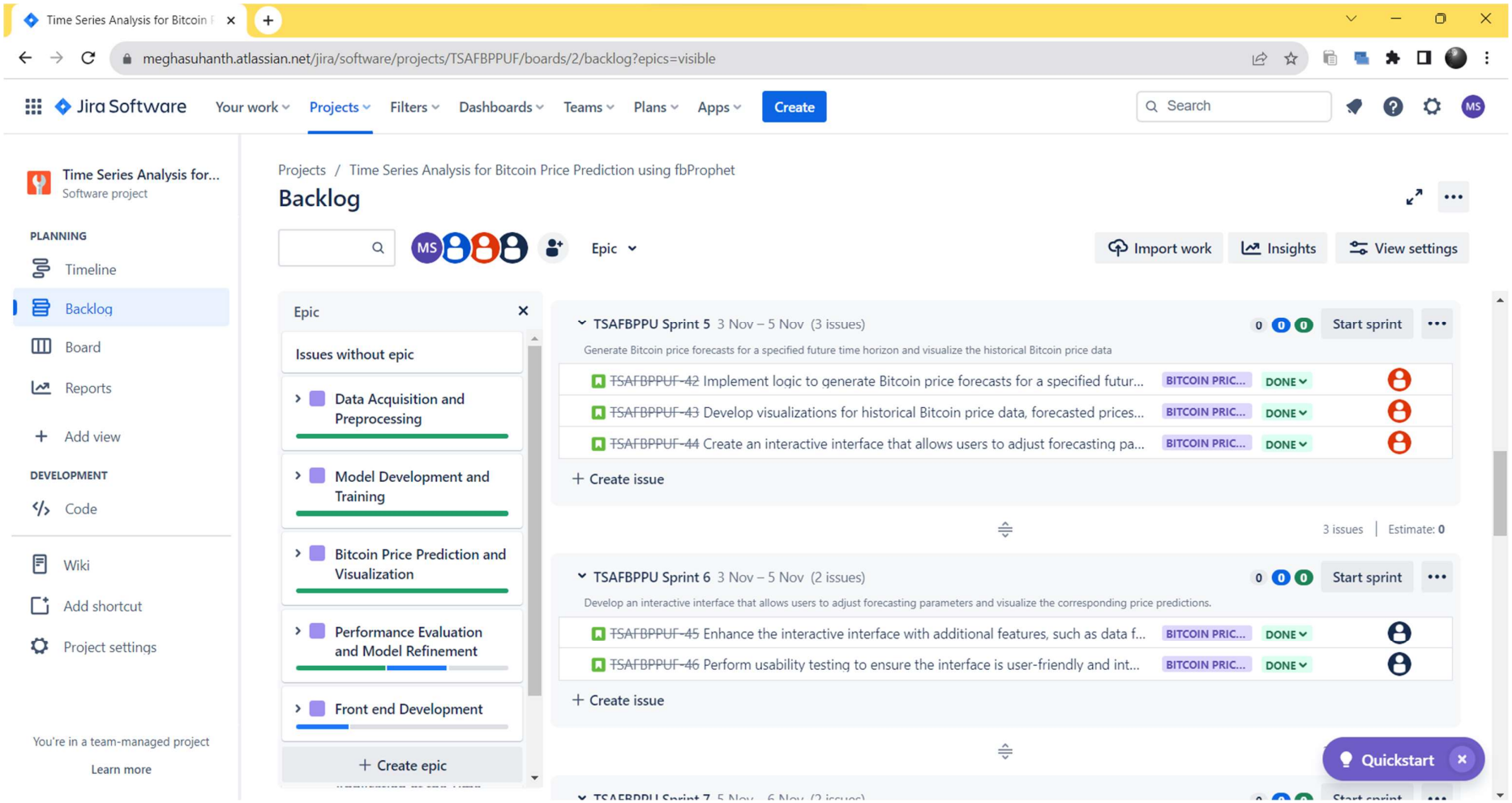
Board:

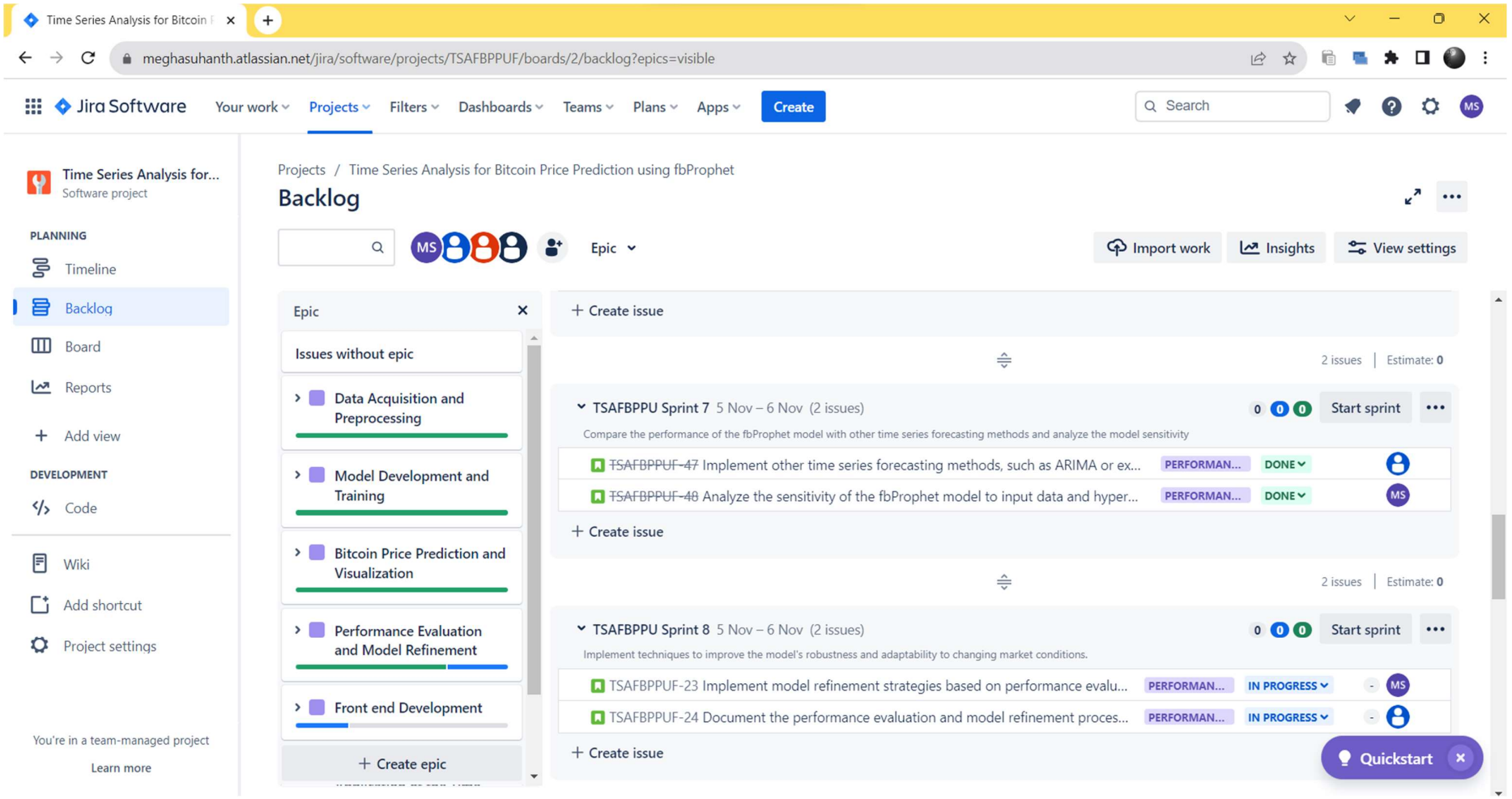

All Sprints - Agile board - Jira
meghasuhanth.atlassian.net/jira/software/projects/TSAFBPPUF/boards/2
Jira Software
Your work
Projects
Filters
Dashboards
Teams
Plans
Apps
Create
Search
Time Series Analysis for...
Software project
PLANNING
Timeline
Backlog
Board
Reports
Add view
DEVELOPMENT
Code
Wiki
Add shortcut
Project settings
You're in a team-managed project
Learn more
Projects / Time Series Analysis for Bitcoin Price Prediction using fbProphet
All sprints
Complete sprint
Epic
Sprint
GROUP BY
None
Import work
Insights
View settings
TO DO 8
Create wireframes and mockups for different screens
FRONT END DEVELOPMENT
TSAFBPPUF-26
Develop the front-end components using appropriate technologies(HTML, CSS, JavaScript frameworks like React or Angular)
FRONT END DEVELOPMENT
TSAFBPPUF-27
Display the impact of parameter changes on the forecasted prices in real-time, enabling users to make informed decisions.
FRONT END DEVELOPMENT
IN PROGRESS 3
Implement model refinement strategies based on performance evaluation and sensitivity analysis.
PERFORMANCE EVALUATION AND MODE...
TSAFBPPUF-23
Document the performance evaluation and model refinement process, including findings and recommendations.
PERFORMANCE EVALUATION AND MODE...
TSAFBPPUF-24
Design the user interface for the web application
FRONT END DEVELOPMENT
TSAFBPPUF-25
DONE 1
Investigate techniques to improve the model's robustness to changing market conditions.
PERFORMANCE EVALUATION AND MODE...
TSAFBPPUF-22
Quickstart

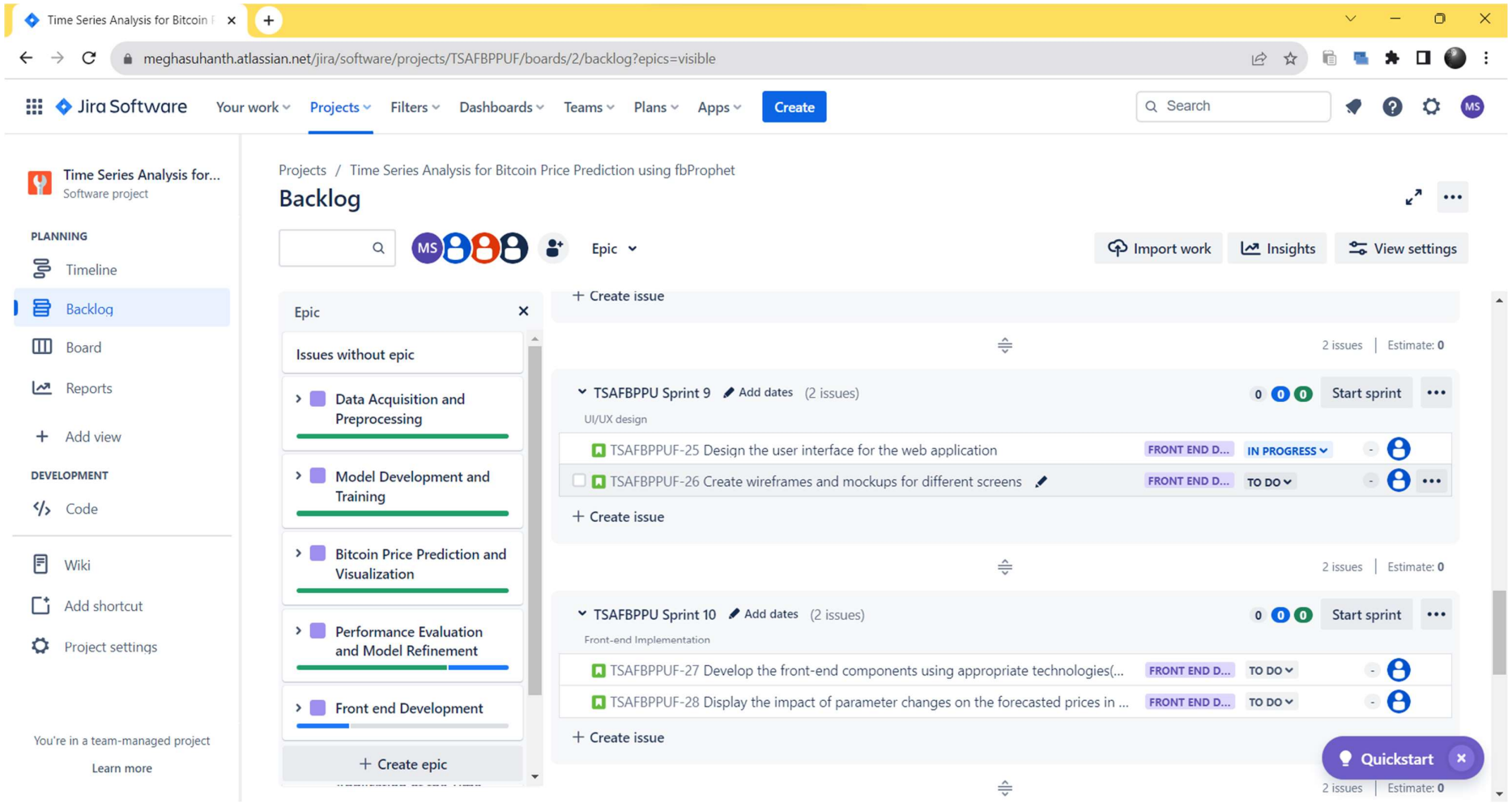
Backlog:

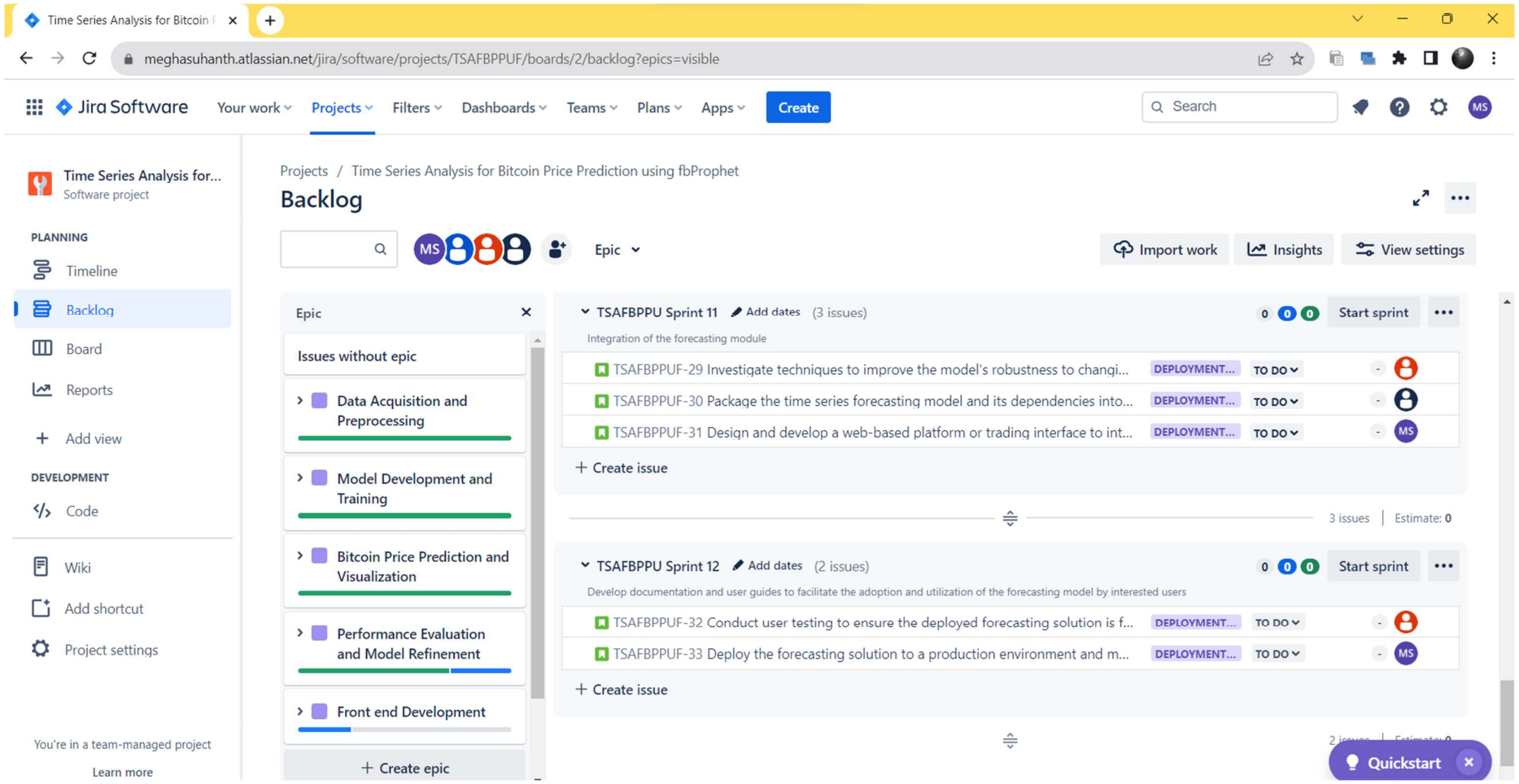

Time Series Analysis for Bitcoin
meghasuhanth.atlassian.net/jira/software/projects/TSAFBPPUF/boards/2/backlog?epics=visible
Jira Software
Your work
Projects
Filters
Dashboards
Teams
Plans
Apps
Create
Search
Time Series Analysis for...
Software project
PLANNING
Timeline
Backlog
Board
Reports
Add view
DEVELOPMENT
Code
Wiki
Add shortcut
Project settings
You're in a team-managed project
Learn more
Projects / Time Series Analysis for Bitcoin Price Prediction using fbProphet
Backlog
Epic
Import work
Insights
View settings
Issues without epic
Data Acquisition and Preprocessing
Model Development and Training
Bitcoin Price Prediction and Visualization
Performance Evaluation and Model Refinement
Front end Development
Create epic
TSAFBPPU Sprint 1 1 Nov – 2 Nov (2 issues)
Design and Implement a data acquisition pipeline
Start sprint
TSAFBPPUF-34 Research and select a reliable source for historical Bitcoin price data.
TSAFBPPUF-35 Design and implement a data acquisition pipeline using appropriate to...
DATA ACQUIS...
DONE
Create issue
2 issues | Estimate: 0
TSAFBPPU Sprint 2 1 Nov – 2 Nov (2 issues)
Transform the Bitcoin price data into a format suitable.
TSAFBPPUF-36 Implement more advanced data preprocessing techniques, addressing ...
TSAFBPPUF-37 Evaluate data quality and ensure the preprocessed data is suitable for ti...
TSAFBPPU Sprint 3 2 Nov – 3 Nov (2 issues)
Install and configure the fbProphet time series forecasting library
Quickstart


Time Series Analysis for Bitcoin
meghasuhanth.atlassian.net/jira/software/projects/TSAFBPPUF/boards/2/backlog?epics=visible
Jira Software
Your work
Projects
Filters
Dashboards
Teams
Plans
Apps
Create
Search
Time Series Analysis for...
Software project
PLANNING
Timeline
Backlog
Board
Reports
Add view
DEVELOPMENT
Code
Wiki
Add shortcut
Project settings
You're in a team-managed project
Learn more
Projects / Time Series Analysis for Bitcoin Price Prediction using fbProphet
Backlog
Epic
Import work
Insights
View settings
Epic
Issues without epic
Data Acquisition and Preprocessing
Model Development and Training
Bitcoin Price Prediction and Visualization
Performance Evaluation and Model Refinement
Front end Development
Create epic
TSAFBPPU Sprint 3 2 Nov – 3 Nov (2 issues)
Start sprint
Install and configure the fbProphet time series forecasting library
TSAFBPPUF-38 Install and configure the fbProphet time series forecasting library in the...
TSAFBPPUF-39 Explore different fbProphet model configurations and hyperparameters...
MODEL DEVE...
DONE
Create issue
2 issues | Estimate: 0
TSAFBPPU Sprint 4 2 Nov – 3 Nov (2 issues)
Start sprint
Train and Evaluate the performance of the fbProphet model
TSAFBPPUF-40 Fine-tune the fbProphet model hyperparameters to optimize forecastin...
TSAFBPPUF-41 Evaluate the model's performance using various metrics, such as MAE a...
Create issue
2 issues | Estimate: 0
TSAFBPPU Sprint 5 3 Nov – 5 Nov (3 issues)
Generate Bitcoin price forecasts for a specified future time horizon and visualize the historical Bitcoin price data
Quickstart

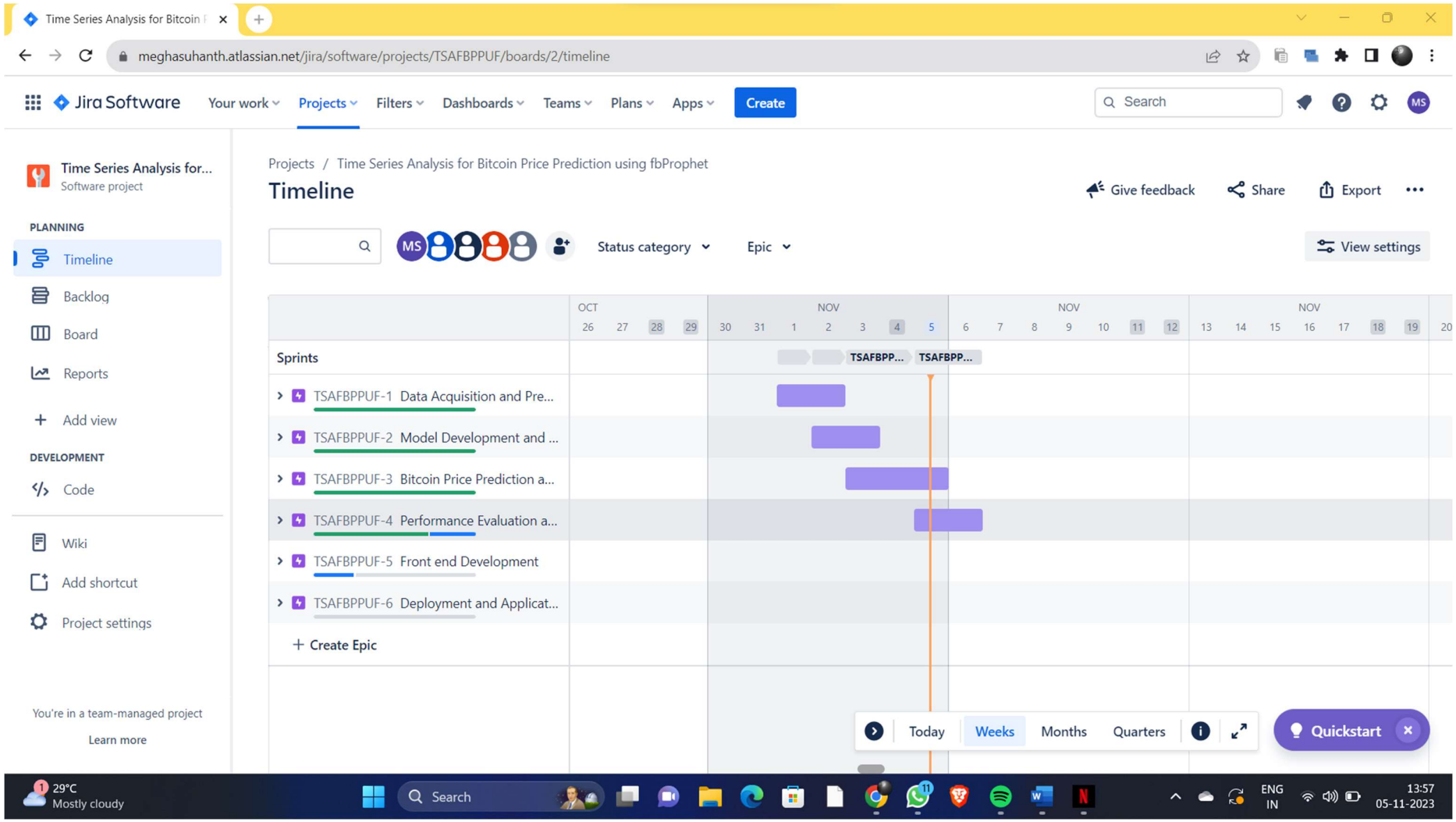

Time Series Analysis for Bitcoin
meghasuhanth.atlassian.net/jira/software/projects/TSAFBPPUF/boards/2/backlog?epics=visible
Jira Software
Your work
Projects
Filters
Dashboards
Teams
Plans
Apps
Create
Search
Time Series Analysis for...
Software project
PLANNING
Timeline
Backlog
Board
Reports
Add view
DEVELOPMENT
Code
Wiki
Add shortcut
Project settings
You're in a team-managed project
Learn more
Projects / Time Series Analysis for Bitcoin Price Prediction using fbProphet
Backlog
Epic
Import work
Insights
View settings
Epic
Issues without epic
Data Acquisition and Preprocessing
Model Development and Training
Bitcoin Price Prediction and Visualization
Performance Evaluation and Model Refinement
Front end Development
Create epic
TSAFBPPU Sprint 5 3 Nov – 5 Nov (3 issues)
Start sprint
Generate Bitcoin price forecasts for a specified future time horizon and visualize the historical Bitcoin price data
TSAFBPPUF-42 Implement logic to generate Bitcoin price forecasts for a specified futur...
TSAFBPPUF-43 Develop visualizations for historical Bitcoin price data, forecasted prices...
TSAFBPPUF-44 Create an interactive interface that allows users to adjust forecasting pa...
BITCOIN PRIC...
DONE
Create issue
3 issues | Estimate: 0
TSAFBPPU Sprint 6 3 Nov – 5 Nov (2 issues)
Start sprint
Develop an interactive interface that allows users to adjust forecasting parameters and visualize the corresponding price predictions.
TSAFBPPUF-45 Enhance the interactive interface with additional features, such as data f...
TSAFBPPUF-46 Perform usability testing to ensure the interface is user-friendly and int...
Create issue
Quickstart


Time Series Analysis for Bitcoin
meghasuhanth.atlassian.net/jira/software/projects/TSAFBPPUF/boards/2/backlog?epics=visible
Jira Software
Your work
Projects
Filters
Dashboards
Teams
Plans
Apps
Create
Search
Time Series Analysis for...
Software project
PLANNING
Timeline
Backlog
Board
Reports
Add view
DEVELOPMENT
Code
Wiki
Add shortcut
Project settings
You're in a team-managed project
Learn more
Projects / Time Series Analysis for Bitcoin Price Prediction using fbProphet
Backlog
Epic
Import work
Insights
View settings
Epic
Issues without epic
Data Acquisition and Preprocessing
Model Development and Training
Bitcoin Price Prediction and Visualization
Performance Evaluation and Model Refinement
Front end Development
Create epic
Create issue
2 issues | Estimate: 0
TSAFBPPU Sprint 7 5 Nov – 6 Nov (2 issues)
Start sprint
Compare the performance of the fbProphet model with other time series forecasting methods and analyze the model sensitivity
TSAFBPPUF-47 Implement other time series forecasting methods, such as ARIMA or ex... PERFORMAN... DONE
TSAFBPPUF-48 Analyze the sensitivity of the fbProphet model to input data and hyper... PERFORMAN... DONE
Create issue
2 issues | Estimate: 0
TSAFBPPU Sprint 8 5 Nov – 6 Nov (2 issues)
Start sprint
Implement techniques to improve the model's robustness and adaptability to changing market conditions.
TSAFBPPUF-23 Implement model refinement strategies based on performance evalu... PERFORMAN... IN PROGRESS
TSAFBPPUF-24 Document the performance evaluation and model refinement proces... PERFORMAN... IN PROGRESS
Create issue
Quickstart


Time Series Analysis for Bitcoin
meghasuhanth.atlassian.net/jira/software/projects/TSAFBPPUF/boards/2/backlog?epics=visible
Jira Software
Your work
Projects
Filters
Dashboards
Teams
Plans
Apps
Create
Search
Time Series Analysis for...
Software project
PLANNING
Timeline
Backlog
Board
Reports
Add view
DEVELOPMENT
Code
Wiki
Add shortcut
Project settings
You're in a team-managed project
Learn more
Projects / Time Series Analysis for Bitcoin Price Prediction using fbProphet
Backlog
Epic
Import work
Insights
View settings
Epic
Issues without epic
Data Acquisition and Preprocessing
Model Development and Training
Bitcoin Price Prediction and Visualization
Performance Evaluation and Model Refinement
Front end Development
Create epic
Create issue
2 issues | Estimate: 0
TSAFBPPU Sprint 9
Add dates
(2 issues)
Start sprint
UI/UX design
TSAFBPPUF-25 Design the user interface for the web application
FRONT END D...
IN PROGRESS
TSAFBPPUF-26 Create wireframes and mockups for different screens
TO DO
Create issue
2 issues | Estimate: 0
TSAFBPPU Sprint 10
Add dates
(2 issues)
Start sprint
Front-end Implementation
TSAFBPPUF-27 Develop the front-end components using appropriate technologies(...
TSAFBPPUF-28 Display the impact of parameter changes on the forecasted prices in ...
Create issue
Quickstart
2 issues | Estimate: 0


Time Series Analysis for Bitcoin
meghasuhanth.atlassian.net/jira/software/projects/TSAFBPPUF/boards/2/backlog?epics=visible
Jira Software
Your work
Projects
Filters
Dashboards
Teams
Plans
Apps
Create
Search
Time Series Analysis for...
Software project
PLANNING
Timeline
Backlog
Board
Reports
Add view
DEVELOPMENT
Code
Wiki
Add shortcut
Project settings
You're in a team-managed project
Learn more
Projects / Time Series Analysis for Bitcoin Price Prediction using fbProphet
Backlog
Epic
Import work
Insights
View settings
Epic
Issues without epic
Data Acquisition and Preprocessing
Model Development and Training
Bitcoin Price Prediction and Visualization
Performance Evaluation and Model Refinement
Front end Development
Create epic
TSAFBPPU Sprint 11
Add dates
(3 issues)
Start sprint
Integration of the forecasting module
TSAFBPPUF-29 Investigate techniques to improve the model's robustness to changi...
TSAFBPPUF-30 Package the time series forecasting model and its dependencies into...
TSAFBPPUF-31 Design and develop a web-based platform or trading interface to int...
DEPLOYMENT...
TO DO
Create issue
3 issues
Estimate: 0
TSAFBPPU Sprint 12
Add dates
(2 issues)
Start sprint
Develop documentation and user guides to facilitate the adoption and utilization of the forecasting model by interested users
TSAFBPPUF-32 Conduct user testing to ensure the deployed forecasting solution is f...
TSAFBPPUF-33 Deploy the forecasting solution to a production environment and m...
Create issue
Quickstart

Timeline:


Time Series Analysis for Bitcoin
meghasuhanth.atlassian.net/jira/software/projects/TSAFBPPUF/boards/2/timeline
Jira Software
Your work
Projects
Filters
Dashboards
Teams
Plans
Apps
Create
Search
Time Series Analysis for...
Software project
PLANNING
Timeline
Backlog
Board
Reports
Add view
DEVELOPMENT
Code
Wiki
Add shortcut
Project settings
You're in a team-managed project
Learn more
Projects / Time Series Analysis for Bitcoin Price Prediction using fbProphet
Timeline
Give feedback
Share
Export
Status category
Epic
View settings
OCT
NOV
NOV
NOV
Sprints
TSAFBPP...
TSAFBPP...
TSAFBPUF-1 Data Acquisition and Pre...
TSAFBPPUF-2 Model Development and ...
TSAFBPPUF-3 Bitcoin Price Prediction a...
TSAFBPPUF-4 Performance Evaluation a...
TSAFBPPUF-5 Front end Development
TSAFBPPUF-6 Deployment and Applicat...
Create Epic
Today
Weeks
Months
Quarters
Quickstart
29°C
Mostly cloudy
Search
ENG
IN
13:57
05-11-2023

https://www.visual-paradigm.com/scrum/scrum-burndown-chart/
https://www.atlassian.com/agile/tutorials/burndown-charts

Reference:

https://www.atlassian.com/agile/project-management
https://www.atlassian.com/agile/tutorials/how-to-do-scrum-with-jira-software https://www.atlassian.com/agile/tutorials/epics
https://www.atlassian.com/agile/tutorials/sprints
https://www.atlassian.com/agile/project-management/estimation
https://www.atlassian.com/agile/tutorials/burndown-charts

7. CODING & SOLUTIONING

Certainly! When it comes to coding and problem-solving, there are numerous important aspects, but I'll highlight two key features that are crucial for success:

Logical Thinking and Problem-Solving Skills:

Logical Thinking: Coding involves creating a logical sequence of instructions to solve a problem or achieve a goal. Strong logical thinking is essential for understanding the problem, breaking it down into smaller parts, and devising an effective solution.

Problem-Solving Skills: Coding is essentially problem-solving. Programmers encounter various challenges, and the ability to break down complex problems into manageable components is vital. Effective problem-solving involves analyzing the problem, identifying potential solutions, and implementing the best course of action.

Algorithmic and Analytical Skills:

Algorithmic Thinking: Writing code often requires designing algorithms - step-by-step procedures or formulas for solving a problem. Strong algorithmic thinking involves creating efficient, clear, and well-structured algorithms.

Analytical Skills: Debugging and optimizing code demand analytical skills. Being able to analyze code, understand its behavior, and identify areas for improvement is crucial. This includes understanding time and space complexity, identifying bottlenecks, and making optimizations.

These features are interconnected, as logical thinking is foundational for effective problem-solving, and algorithmic and analytical skills support the development of robust, efficient code. Cultivating these skills is essential for becoming a proficient coder and a successful problem solver in various domains.

Project Overview:

The project involves the application of Time Series Analysis to predict Bitcoin prices using the Prophet forecasting tool. The process encompasses design, perpetration, installation, operationalization, and ongoing monitoring of the system.

Project Phases:

Design Phase:

Define the objective: Predict Bitcoin prices using Time Series Analysis.

Plan the architecture, including data collection, preprocessing, and the use of the Prophet forecasting tool.

Perpetration (Preparation) Stage:

Prepare the design for implementation, emphasizing the integration of Time Series Analysis.

Develop Python code for data processing and Prophet model implementation.

Installation and Operationalization:

Install the system, including Python environment and necessary libraries.

Operationalize the Time Series model to predict Bitcoin prices.

Testing and Approval:

Conduct testing to ensure the model's accuracy and reliability.

Seek approval from stakeholders, including validation by the "stoner" or relevant authority.

Deployment Phase:

Deploy the system for continuous prediction of Bitcoin prices.

Language and Technology:

Language: Python

Technology: Time Series Analysis using the Prophet forecasting tool.

Libraries and Algorithms:

Libraries:

Python standard libraries

pandas for data

manipulation

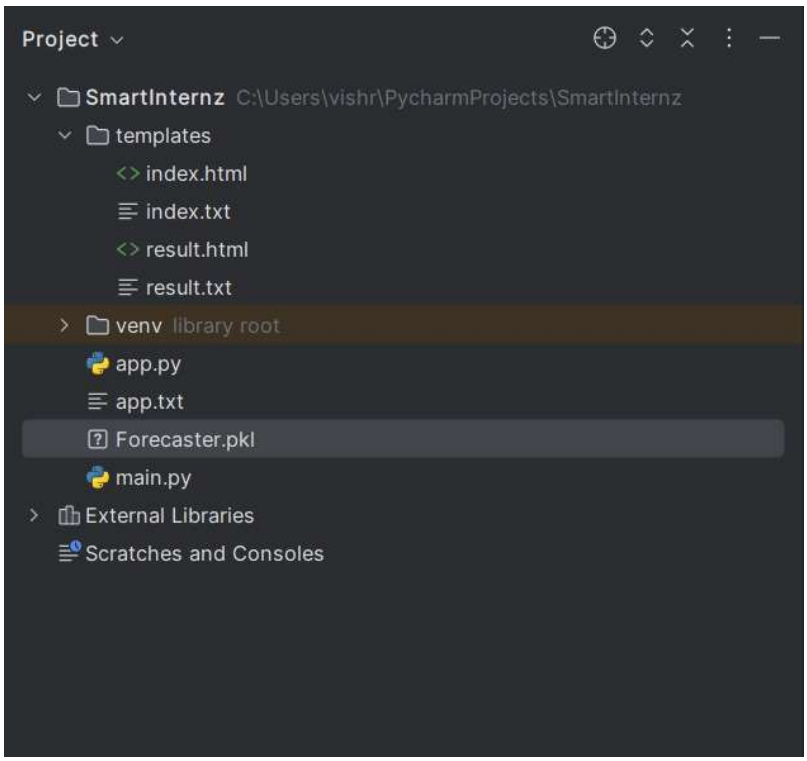
fbprophet for Time Series Analysis

matplotlib for visualization

Algorithms:

Prophet algorithm for forecasting

Project Structure:



All the aforementioned files are intended for the development of a Flask application:

- The static directory holds a CSS file.
- The Flask application comprises HTML pages stored in the templates directory and a Python script named app.py for server-side scripting.
- Within the Model training directory, there is a training file named FB_prophet_Bitcoin_forecasting.ipynb.
- The saved model file is named fbcrypto.pkl.

This collection of files is designated for the construction of a Flask application, including style elements in the static folder, HTML pages within the templates folder, a server-side scripting Python script (app.py), a training file (FB_prophet_Bitcoin_forecasting.ipynb) in the Model training directory, and a saved model file (fbcrypto.pkl).

Milestone 1: Installation of Pre-requisites

- Install Cython, pystan, fbProphet
- Install yfinance
- Do in anaconda environment

Milestone 2: Data Collection

You can collect datasets from different open sources like kaggle.com, data.gov, UCI machine learning repository, etc.

Dataset Link: BTC_USD

Milestone 3: Pre-processing

Data preprocessing is a data mining technique that is used to transform the raw data into a useful and efficient format. So we need to clean the dataset properly in order to fetch good results.

Activity 1: Import Libraries

Import the requisite libraries for data preprocessing, forecasting using FbProphet, and related functionalities.

- It is crucial to include all necessary libraries in the code, including pandas, plotly, Yahoo Finance, and Fbprophet.
- **Pandas**: A swift, potent, versatile, and user-friendly open-source tool for data analysis and manipulation, developed on the Python programming language.
- **Plotly**: The Plotly Python library serves as an interactive, open-source plotting tool, supporting over 40 diverse chart types that span a broad spectrum of statistical, financial, geographical, scientific, and three- dimensional use-cases. It is built on the Plotly JavaScript library.
- **Yahoo Finance**: Retrieve market data through the yfinance module, facilitating the download of pertinent financial data.

```
[ ]  import pandas as pd
     import numpy as np
     import yfinance as yf
     from datetime import datetime
     from datetime import timedelta
     import plotly.graph_objects as go
     from prophet import Prophet
     from prophet.plot import plot_plotly, plot_components_plotly
     import warnings
     warnings.filterwarnings('ignore')
     pd.options.display.float_format = '${:,.2f}'.format
```

Activity 2: Import Dataset

Download the real-time data from the Yahoo Finance library where we need to pass three parameters in the yahoo finance download function i.e. abbreviation name of the cryptocurrency, start date, and today date then we stored it into a variable called df.

```
[2] today=datetime.today().strftime('%Y-%m-%d')
    start_date = '1950-01-01' #Collects all the data from the start of listing of BTC
    data = yf.download('BTC-USD', start_date, today)

[**********100%%**********]  1 of 1 completed
```

```
[3] data
```

Date	Open	High	Low	Close	Adj Close	Volume
2014-09-17	$465.86	$468.17	$452.42	$457.33	$457.33	21056800
2014-09-18	$456.86	$456.86	$413.10	$424.44	$424.44	34483200
2014-09-19	$424.10	$427.83	$384.53	$394.80	$394.80	37919700
2014-09-20	$394.67	$423.30	$389.88	$408.90	$408.90	36863600
2014-09-21	$408.08	$412.43	$393.18	$398.82	$398.82	26580100
...	...	...	...	...	...	...
2023-11-16	$37,879.98	$37,934.62	$35,545.47	$36,154.77	$36,154.77	26007385366
2023-11-17	$36,164.82	$36,704.48	$35,901.23	$36,596.68	$36,596.68	22445028430
2023-11-18	$36,625.37	$36,839.28	$36,233.31	$36,585.70	$36,585.70	11886022717
2023-11-19	$36,585.77	$37,509.36	$36,414.60	$37,386.55	$37,386.55	12915986553
2023-11-20	$37,374.07	$37,756.82	$36,882.53	$37,476.96	$37,476.96	20888209068

3352 rows × 6 columns

- Bitcoin Dataset contains the following Columns

```
data.reset_index(inplace=True)
data.columns
```

```
Index(['Date', 'Open', 'High', 'Low', 'Close', 'Adj Close', 'Volume'], dtype='object')
```

1. Date:- Datewise Information related to the quote currency.

2. Open:- The opening price of the time interval in the quote currency (For BTC/USD,

the price would be USD).

3. High: Highest price reached during the time interval, in the quote currency.

4. Low: Lowest price reached during the time interval, in the quote currency.

5. Close:- The closing price of the time interval, in the quote currency.

6. Adj Close:- Final prices of the time interval, in the quote currency.

7. Volume: Quantity of assets bought or sold, displayed in base currency.

Activity 3: Analyse the data

```
[7]  data.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3352 entries, 0 to 3351
Data columns (total 7 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   Date       3352 non-null   datetime64[ns]
 1   Open       3352 non-null   float64
 2   High       3352 non-null   float64
 3   Low        3352 non-null   float64
 4   Close      3352 non-null   float64
 5   Adj Close  3352 non-null   float64
 6   Volume     3352 non-null   int64
dtypes: datetime64[ns](1), float64(5), int64(1)
memory usage: 183.4 KB
```

Check null values

```
[5] data.isnull().any()
    data.isnull().sum()
```

```
Date         0
Open         0
High         0
Low          0
Close        0
Adj Close    0
Volume       0
dtype: int64
```

Now use the reset_index() function to generate a new DataFrame or Series with the index reset and it will add a date as a column.

```
[4] data.reset_index(inplace=True)
    data.columns
```

```
Index(['Date', 'Open', 'High', 'Low', 'Close', 'Adj Close', 'Volume'], dtype='object')
```

Generate a fresh dataframe by extracting the Date and Open columns, storing it in the variable df1. Afterward, inspect the initial five rows of the data utilizing the head function.

```
data_for_prophet=data[['Date','Open']]
data_for_prophet
```

	Date	Open
0	2014-09-17	$465.86
1	2014-09-18	$456.86
2	2014-09-19	$424.10
3	2014-09-20	$394.67
4	2014-09-21	$408.08
...	...	...
3347	2023-11-16	$37,879.98
3348	2023-11-17	$36,164.82
3349	2023-11-18	$36,625.37
3350	2023-11-19	$36,585.77
3351	2023-11-20	$37,374.07

3352 rows × 2 columns

Rename to reuse easily

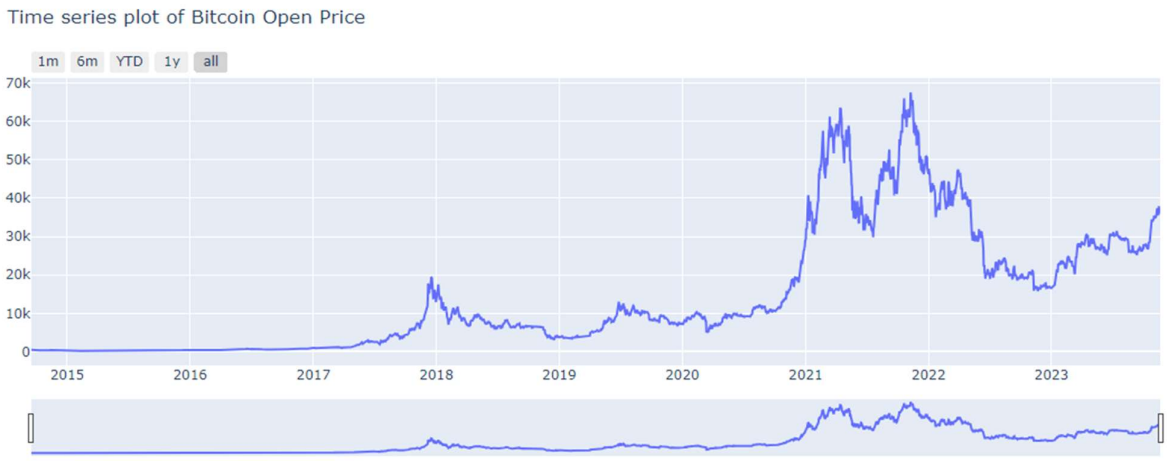
```
[9]  new_column_names={
         "Date":"ds",
         "Open":"y",
     }
     data_for_prophet.rename(columns=new_column_names,inplace=True)
     data_for_prophet
```

	ds	y
0	2014-09-17	$465.86
1	2014-09-18	$456.86
2	2014-09-19	$424.10
3	2014-09-20	$394.67
4	2014-09-21	$408.08

Activity 5: Visualize Time Series Plot

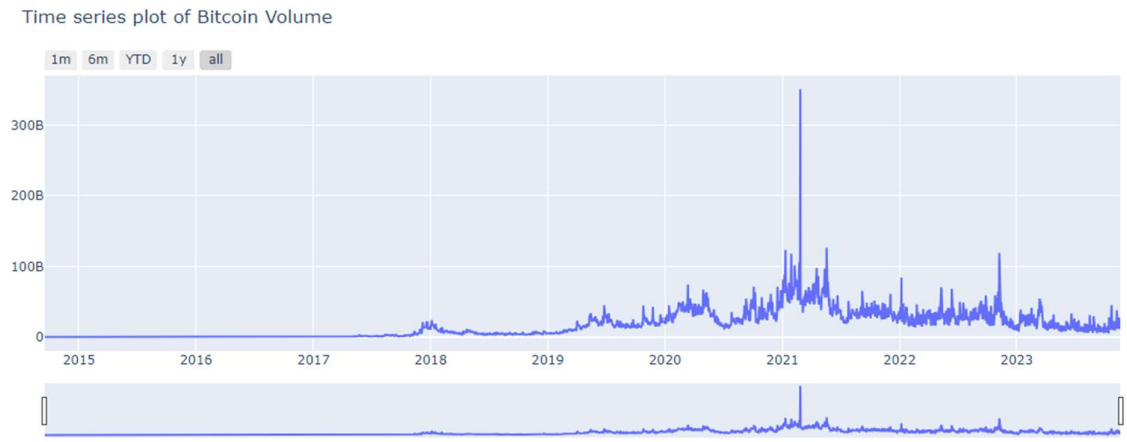
Date vs Open price:

```
[10] import plotly.graph_objects as go

     x = data_for_prophet["ds"]
     y = data_for_prophet["y"]

     fig = go.Figure()
     fig.add_trace(go.Scatter(x=x, y=y))

     # Set title
     fig.update_layout(
         title_text="Time series plot of Bitcoin Open Price"
     )

     fig.update_layout(
         xaxis=dict(
             rangeselector=dict(
                 buttons=list(
                     [
                         dict(count=1, label="1m", step="month", stepmode="backward"),
                         dict(count=6, label="6m", step="month", stepmode="backward"),
                         dict(count=1, label="YTD", step="year", stepmode="todate"),
                         dict(count=1, label="1y", step="year", stepmode="backward"),
                         dict(step="all"),
                     ]
                 ),
                 visible=True
             ),
             rangeslider=dict(visible=True),
             type="date"
         )
     )

     # Show the plot
     fig.show()
```



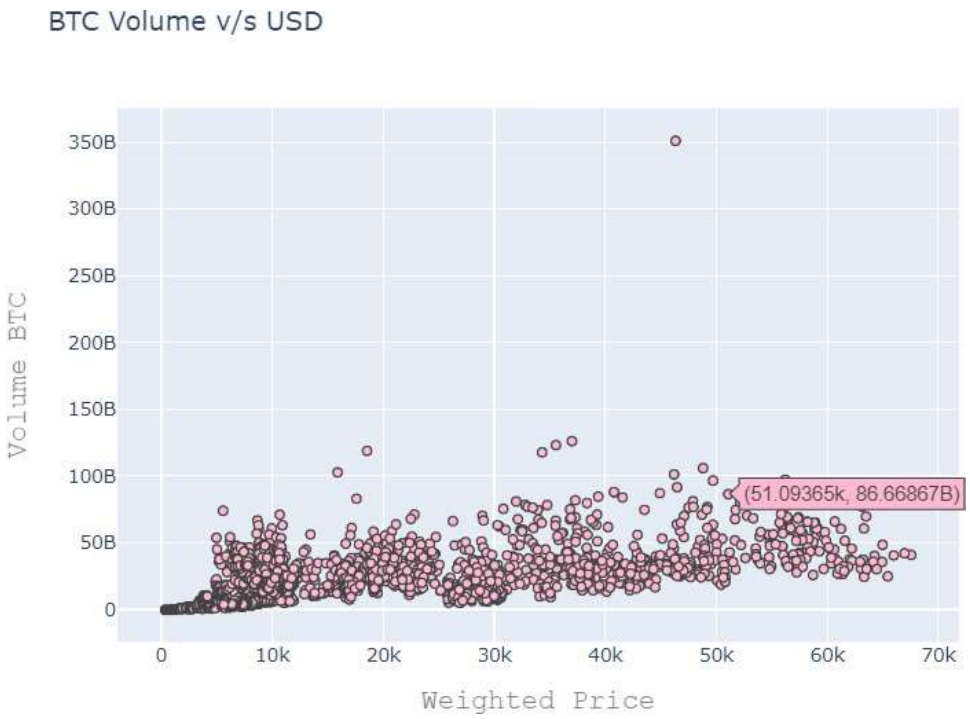
Date vs Volume:

```
[11] import plotly.graph_objects as go

     x = data["Date"]
     y = data["Volume"]

     fig = go.Figure()
     fig.add_trace(go.Scatter(x=x, y=y))

     # Set title
     fig.update_layout(
         title_text="Time series plot of Bitcoin Volume"
     )

     fig.update_layout(
         xaxis=dict(
             rangeselector=dict(
                 buttons=list(
                     [
                         dict(count=1, label="1m", step="month", stepmode="backward"),
                         dict(count=6, label="6m", step="month", stepmode="backward"),
                         dict(count=1, label="YTD", step="year", stepmode="todate"),
                         dict(count=1, label="1y", step="year", stepmode="backward"),
                         dict(step="all"),
                     ]
                 ),
                 visible=True
             ),
             rangeslider=dict(visible=True),
             type="date"
         )
     )

     # Show the plot
     fig.show()
```



Volume vs USD:

```
[12] import plotly.offline as pyo
     pyo.init_notebook_mode(connected=True)


     #BTC Volume vs USD visualization
     trace = go.Scattergl(
         x = data['Close'].astype(float),
         y = data['Volume'].astype(float),
         mode = 'markers',
         marker = dict(
             color = '#FFBAD2',
             line = dict(width = 1)
         )
     )
     layout = go.Layout(
         title='BTC Volume v/s USD',
         xaxis=dict(
             title='Weighted Price',
             titlefont=dict(
                 family='Courier New, monospace',
                 size=18,
                 color='#7f7f7f'
             )
         ),
         yaxis=dict(
             title='Volume BTC',
             titlefont=dict(
                 family='Courier New, monospace',
                 size=18,
                 color='#7f7f7f'
             )))
     data = [trace]
     fig = go.Figure(data=data, layout=layout)
     pyo.iplot(fig, filename='compare_webgl')
```



Milestone 4: Model Building

Activity 1: Fitting the prophet library

Instantiate the Prophet class and fit it to the dataset. By default, Prophet assumes additive seasonality, where the seasonal effect is added to the trend for forecasting. However, in the case of this Bitcoin price time series, an additive approach is not appropriate. The dataset exhibits a distinct yearly cycle, but the default additive seasonality results in a forecast with a seasonality that is disproportionately large at the beginning of the time series and too small at the end.

Given that the seasonality in this time series is not a constant additive factor, as assumed by Prophet, but rather grows with the trend, a more suitable approach is to use multiplicative seasonality. To enable this, the Prophet instance should be configured with seasonality_mode='multiplicative' in the input arguments.

```
[13] #Create a Prophet model object to train it
     model = Prophet(
         seasonality_mode="multiplicative",
     )

     model.fit(data_for_prophet)
```

```
INFO:prophet:Disabling daily seasonality. Run prophet with daily_seasonality=True to override
DEBUG:cmdstanpy:input tempfile: /tmp/tmptz9okrmt/yc8evxf2.json
DEBUG:cmdstanpy:input tempfile: /tmp/tmptz9okrmt/fyfnyjnd.json
DEBUG:cmdstanpy:idx 0
DEBUG:cmdstanpy:running CmdStan, num_threads: None
DEBUG:cmdstanpy:CmdStan args: ['/usr/local/lib/python3.10/dist-packages/prophet/stan_model/pr
16:30:55 - cmdstanpy - INFO - Chain [1] start processing
INFO:cmdstanpy:Chain [1] start processing
16:30:57 - cmdstanpy - INFO - Chain [1] done processing
INFO:cmdstanpy:Chain [1] done processing
<prophet.forecaster.Prophet at 0x7978ca280160>
```

Activity 2:Making Future Predictions

The subsequent phase involves configuring our model for future predictions. This is accomplished through the utilization of the prophet.make_future_dataframe method. By specifying the number of days we intend to forecast into the future with the periods attribute, we generate a dataframe that includes both historical and future dates. The historical dates are crucial for later comparison, allowing us to assess the accuracy of predictions by contrasting them with the actual values present in the 'ds' column.

```
future=model.make_future_dataframe(periods=365)
future
```

	ds
0	2014-09-17
1	2014-09-18
2	2014-09-19
3	2014-09-20
4	2014-09-21
...	...
3712	2024-11-15
3713	2024-11-16
3714	2024-11-17
3715	2024-11-18
3716	2024-11-19

3717 rows × 1 columns

Activity 3: Evaluate the model

The `predict` method is employed to generate future predictions. This results in a dataframe with a 'yhat' column, which encapsulates the forecasted values.
Although the forecast dataframe contains numerous columns upon inspection of its head, our primary focus is on a subset of columns, namely 'ds', 'yhat', 'yhat_lower', and 'yhat_upper'. In this context, 'yhat' represents our predicted forecast, 'yhat_lower' signifies the lower boundary for our predictions, and 'yhat_upper' indicates the upper boundary for our predictions.

```
forecast=model.predict(future)
forecast
```

	ds	trend	yhat_lower	yhat_upper	trend_lower	trend_upper	multiplicative_terms	multiplicative_terms_lower	multiplicative_terms_upper	w
0	2014-09-17	$73.56	$-5,381.41	$5,230.83	$73.56	$73.56	$-0.15	$-0.15	$-0.15	
1	2014-09-18	$74.24	$-5,478.30	$5,815.97	$74.24	$74.24	$-0.15	$-0.15	$-0.15	
2	2014-09-19	$74.93	$-5,433.93	$5,250.46	$74.93	$74.93	$-0.16	$-0.16	$-0.16	
3	2014-09-20	$75.61	$-5,857.19	$5,285.24	$75.61	$75.61	$-0.16	$-0.16	$-0.16	
4	2014-09-21	$76.29	$-5,451.14	$5,252.92	$76.29	$76.29	$-0.16	$-0.16	$-0.16	
...	...	...	...	...	...	...	...	...	...	
3712	2024-11-15	$14,441.17	$251.79	$34,254.12	$1,003.15	$29,074.26	$0.14	$0.14	$0.14	
3713	2024-11-16	$14,421.36	$295.58	$34,694.47	$960.48	$29,139.37	$0.13	$0.13	$0.13	
3714	2024-11-17	$14,401.56	$-373.48	$33,381.39	$876.34	$29,204.48	$0.13	$0.13	$0.13	
3715	2024-11-18	$14,381.75	$836.07	$33,086.27	$784.86	$29,269.60	$0.12	$0.12	$0.12	
3716	2024-11-19	$14,361.95	$-224.92	$33,368.83	$688.43	$29,334.71	$0.12	$0.12	$0.12	

3717 rows × 19 columns

```
forecast[['ds','yhat','yhat_lower','yhat_upper']]
```

	ds	yhat	yhat_lower	yhat_upper
0	2014-09-17	$62.42	$-5,381.41	$5,230.83
1	2014-09-18	$63.01	$-5,478.30	$5,815.97
2	2014-09-19	$63.16	$-5,433.93	$5,250.46
3	2014-09-20	$63.68	$-5,857.19	$5,285.24
4	2014-09-21	$64.20	$-5,451.14	$5,252.92
...	...	...	...	...
3712	2024-11-15	$16,431.27	$251.79	$34,254.12
3713	2024-11-16	$16,346.50	$295.58	$34,694.47
3714	2024-11-17	$16,251.97	$-373.48	$33,381.39
3715	2024-11-18	$16,161.50	$836.07	$33,086.27
3716	2024-11-19	$16,027.98	$-224.92	$33,368.83

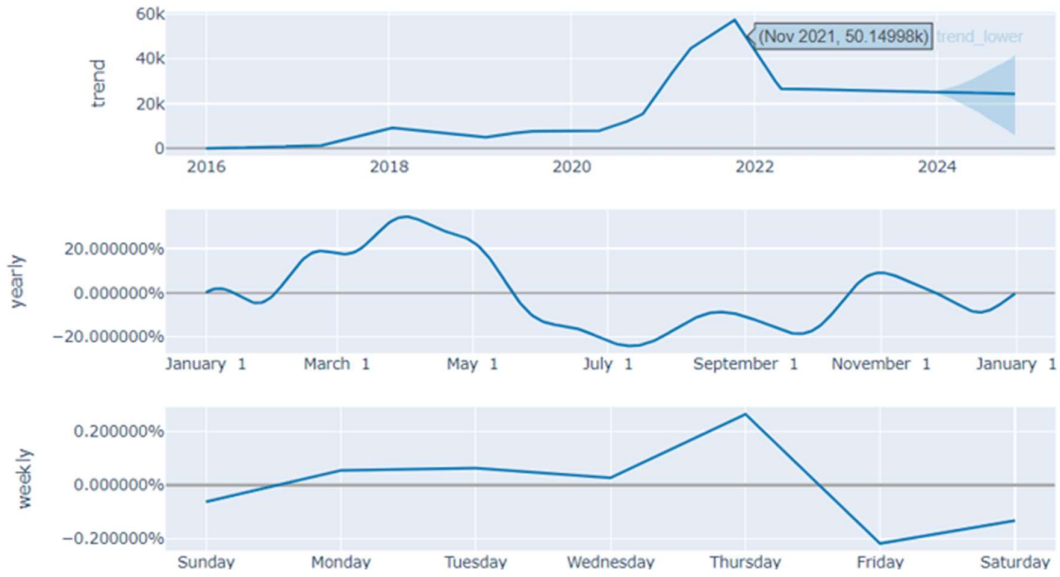
3717 rows × 4 columns

- To forecast the price for the next day, we calculate the DateTime and store it in the `next_day` variable. Subsequently, we use this variable to predict the corresponding value.

```
[18] next_day=(datetime.today()+timedelta(days=1)).strftime('%Y-%m-%d')
     forecast[forecast['ds']==next_day]['yhat'].item()

     23714.889194689204
```

- Now, let's create a visualization to represent the forecasted values of the Bitcoin price extending until the next year.


```
plot_components_plotly(model,forecast)
```



Activity 4: Save the model.

This is the final activity of this milestone, here you will be saving the model to integrate to the web application.

```
import pickle
pickle.dump(model,open('fbcryto.pkl','wb'))
```

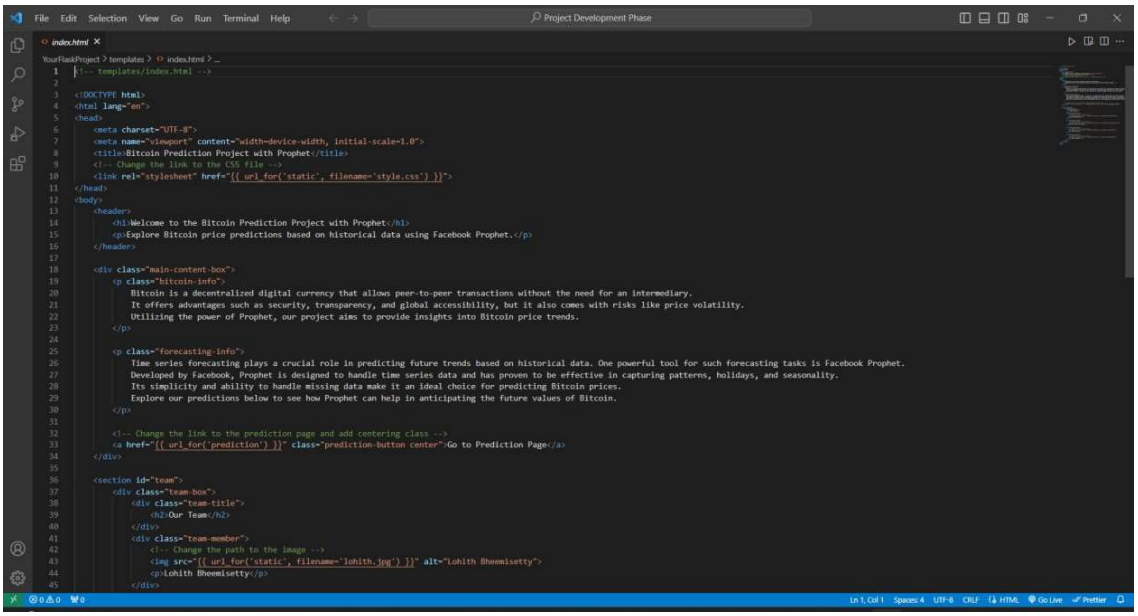
Milestone 5: Application Building

Having trained our model, the next step is to construct our Flask application, which will run locally in our browser, providing a user interface.

Within the Flask application, input parameters are obtained from an HTML page. These parameters are then utilized by the model to predict the price of Bitcoin on a selected date. The predicted price is then displayed on the HTML page to inform the user. When the user interacts with the UI and selects the "predict" button, the application navigates to the next page where the user can choose a date and obtain the corresponding prediction output.

There will be two web pages index.html when entered and predict.html for prediction page

Index.html:


File Edit Selection View Go Run Terminal Help
Project Development Phase
index.html
YourFlaskProject > templates > index.html > ...
<!-- templates/index.html -->
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Bitcoin Prediction Project with Prophet</title>
<!-- Change the link to the CSS file -->
<link rel="stylesheet" href="{{ url_for('static', filename='style.css') }}">
</head>
<body>
<header>
<h1>Welcome to the Bitcoin Prediction Project with Prophet</h1>
Explore Bitcoin price predictions based on historical data using Facebook Prophet.
</header>

Bitcoin is a decentralized digital currency that allows peer-to-peer transactions without the need for an intermediary.
It offers advantages such as security, transparency, and global accessibility, but it also comes with risks like price volatility.
Utilizing the power of Prophet, our project aims to provide insights into Bitcoin price trends.

Time series forecasting plays a crucial role in predicting future trends based on historical data. One powerful tool for such forecasting tasks is Facebook Prophet.
Developed by Facebook, Prophet is designed to handle time series data and has proven to be effective in capturing patterns, holidays, and seasonality.
Its simplicity and ability to handle missing data make it an ideal choice for predicting Bitcoin prices.
Explore our predictions below to see how Prophet can help in anticipating the future values of Bitcoin.

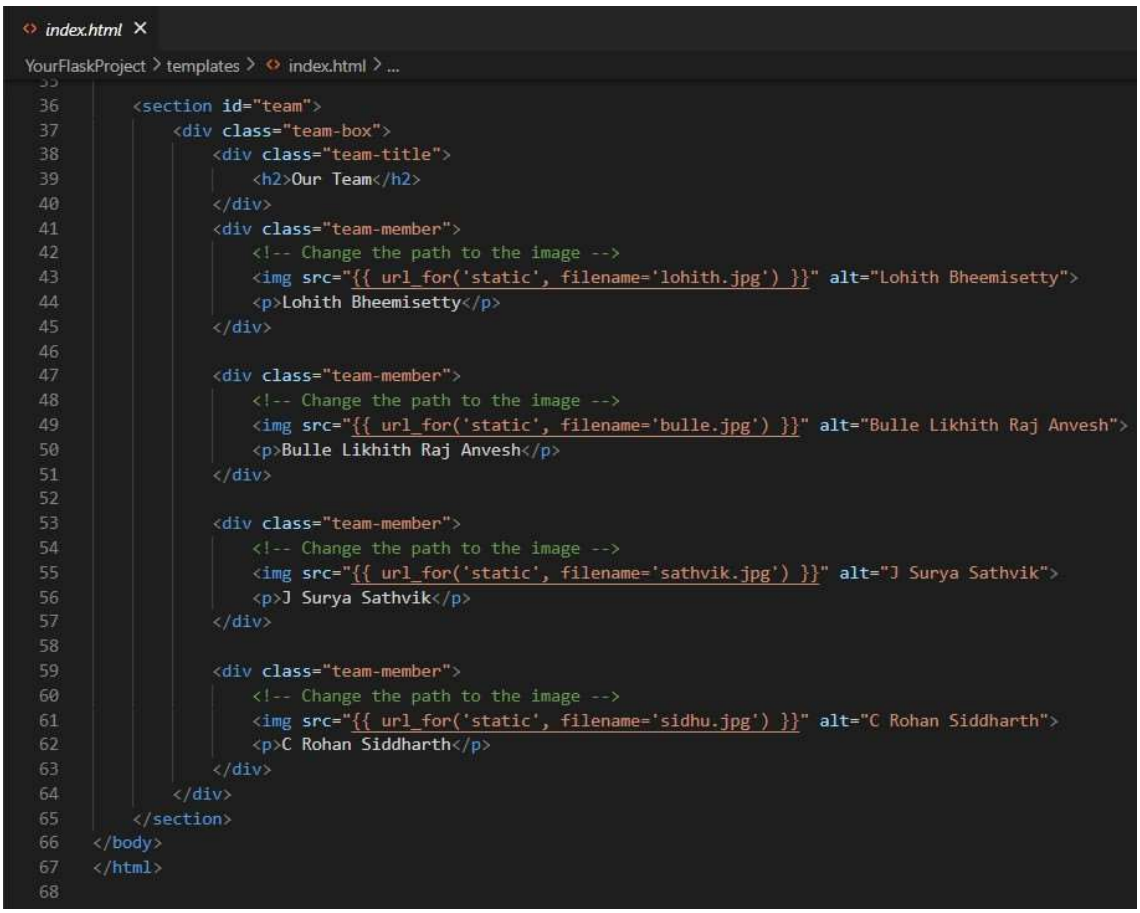
<!-- Change the link to the prediction page and add centering class -->
Go to Prediction Page

<section id="team">

<h2>Our Team</h2>

<!-- Change the path to the image -->
<img src="{{ url_for('static', filename='lohith.jpg') }}" alt="Lohith Bheemisetty">
Lohith Bheemisetty

Ln 1, Col 1 Spaces: 4 UTF-8 CRLF HTML Go Live Prettier


index.html
YourFlaskProject > templates > index.html > ...
<section id="team">

<h2>Our Team</h2>

<!-- Change the path to the image -->
<img src="{{ url_for('static', filename='lohith.jpg') }}" alt="Lohith Bheemisetty">
Lohith Bheemisetty

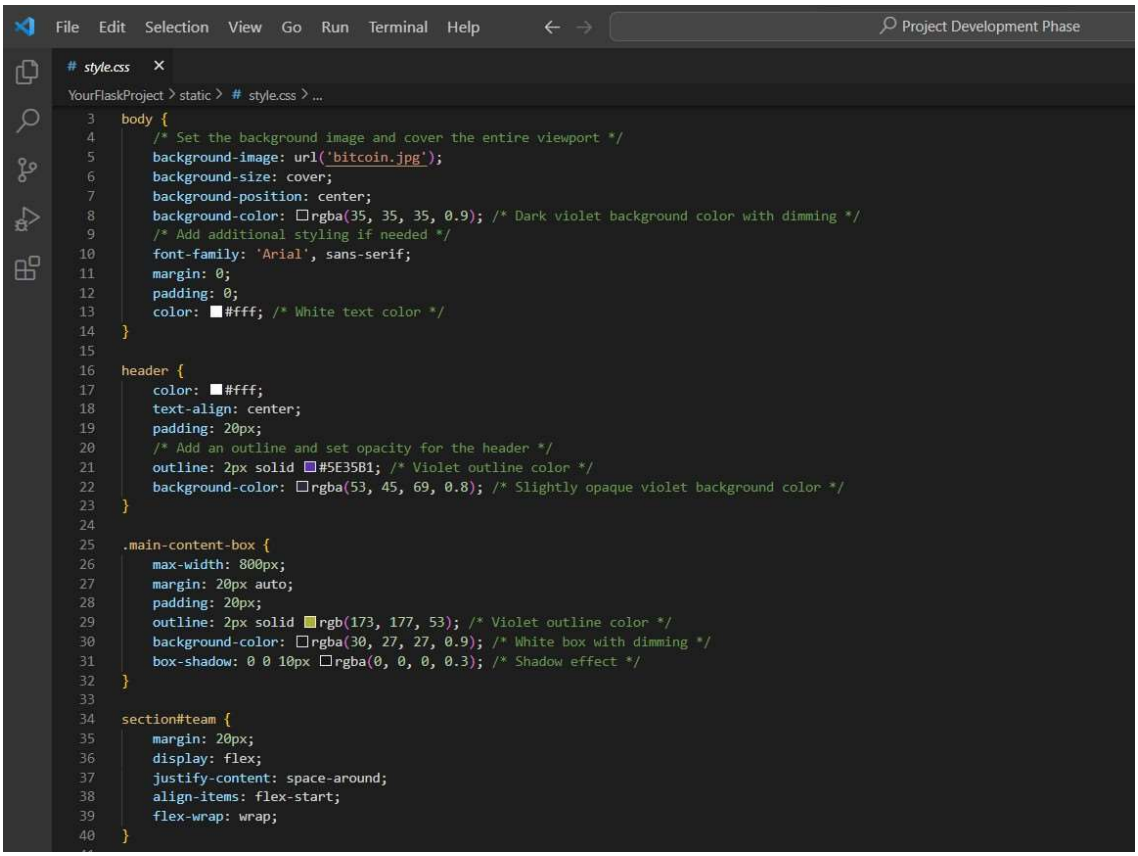
<!-- Change the path to the image -->
<img src="{{ url_for('static', filename='bulle.jpg') }}" alt="Bulle Likhith Raj Anvesh">
Bulle Likhith Raj Anvesh

<!-- Change the path to the image -->
<img src="{{ url_for('static', filename='sathvik.jpg') }}" alt="J Surya Sathvik">
J Surya Sathvik

<!-- Change the path to the image -->
<img src="{{ url_for('static', filename='sidhu.jpg') }}" alt="C Rohan Siddharth">
C Rohan Siddharth

</section>
</body>
</html>

Style.css(for index page):



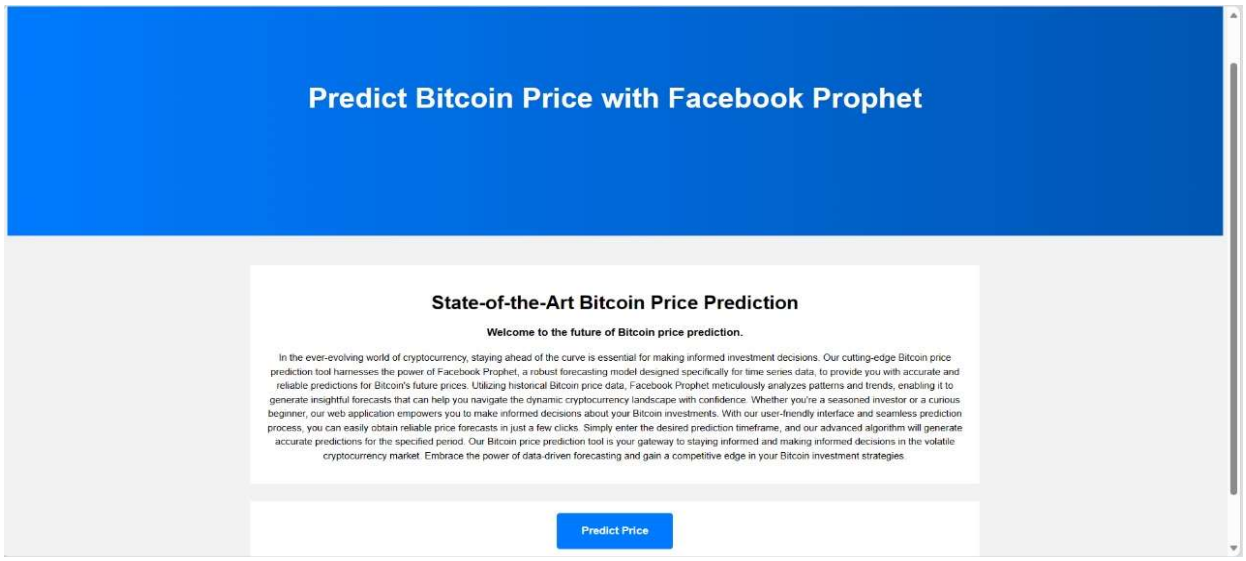

```css
body {
    /* Set the background image and cover the entire viewport */
    background-image: url('bitcoin.jpg');
    background-size: cover;
    background-position: center;
    background-color: rgba(35, 35, 35, 0.9); /* Dark violet background color with dimming */
    /* Add additional styling if needed */
    font-family: 'Arial', sans-serif;
    margin: 0;
    padding: 0;
    color: #fff; /* White text color */
}

header {
    color: #fff;
    text-align: center;
    padding: 20px;
    /* Add an outline and set opacity for the header */
    outline: 2px solid #5E35B1; /* Violet outline color */
    background-color: rgba(53, 45, 69, 0.8); /* Slightly opaque violet background color */
}

.main-content-box {
    max-width: 800px;
    margin: 20px auto;
    padding: 20px;
    outline: 2px solid rgb(173, 177, 53); /* Violet outline color */
    background-color: rgba(30, 27, 27, 0.9); /* White box with dimming */
    box-shadow: 0 0 10px rgba(0, 0, 0, 0.3); /* Shadow effect */
}

section#team {
    margin: 20px;
    display: flex;
    justify-content: space-around;
    align-items: flex-start;
    flex-wrap: wrap;
}
```

style.css

YourFlaskProject > static > # style.css > .prediction-button

```css
.team-box {
    max-width: 800px;
    margin: 20px auto;
    padding: 20px;
    background-color: rgba(50, 42, 42, 0.5); /* White box with dimming */
    box-shadow: 0 0 10px rgba(0, 0, 0, 0.3); /* Shadow effect */
    display: flex;
    justify-content: space-around;
    align-items: center;
    flex-wrap: wrap;
}
.team-member {
    text-align: center;
    margin: 10px;
    max-width: 120px;
    flex-grow: 1; /* Allow the items to grow and take available space */
}
.team-member img {
    max-width: 100%;
    height: 120px;
    object-fit: cover;
    border-radius: 50%;
}
.prediction-button {
    display: inline-block;
    padding: 10px 20px;
    margin: 20px auto;
    background-color: #5E35B1;
    color: #fff;
    text-decoration: none;
    border-radius: 10px;
    border: 2px solid #5E35B1;
    transition: background-color 0.3s ease;
}
.prediction-button:hover {
    background-color: #311B92;
}

.bitcoin-info {
    text-align: justify;
}
.center {
    text-align: center;
}
```

Index page:



Predict.html

style.css | app.py | predict.html

YourFlaskProject > templates > predict.html > html > head > style > button

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Bitcoin Prediction - Enter Date</title>
    <style>
        body {
    background-color: rgba(35, 35, 35, 0.9);
    font-family: 'Arial', sans-serif;
    margin: 0;
    padding: 0;
    color: #fff; /* White text color */
        }

        h1, label, p {
            color: #fff; /* Set text color to white */
        }

        form {
            background-color: rgba(0, 0, 0, 0.7); /* Semi-transparent black background for the form */
            padding: 20px;
            border-radius: 10px;
            margin-top: 20px;
        }

        label {
            display: block;
            margin-bottom: 10px;
        }

        input {
            padding: 10px;
            margin-bottom: 10px;
        }

        button {
            padding: 10px 20px;
            background-color: #5E35B1;
            color: #fff;
            border: none;
            border-radius: 5px;
            cursor: pointer;
        }
```

style.css app.py predict.html

YourFlaskProject > templates > predict.html > html > head > style > button

```
        button:hover {
            background-color: #311B92;
        }

        /* Add additional styles as needed */
    </style>
</head>
<body>
    <h1>Bitcoin Price Prediction using fbProphet</h1>

    <form action="/predict" method="post">
        <label for="Date">Select Date:</label>
        <input type="date" id="Date" name="Date" required>
        <button type="submit">Predict</button>
    </form>

    {% if prediction_text %}
        <p>{{ prediction_text }}</p>
    {% endif %}

</body>
</html>
```

Activity 2: Build app.py

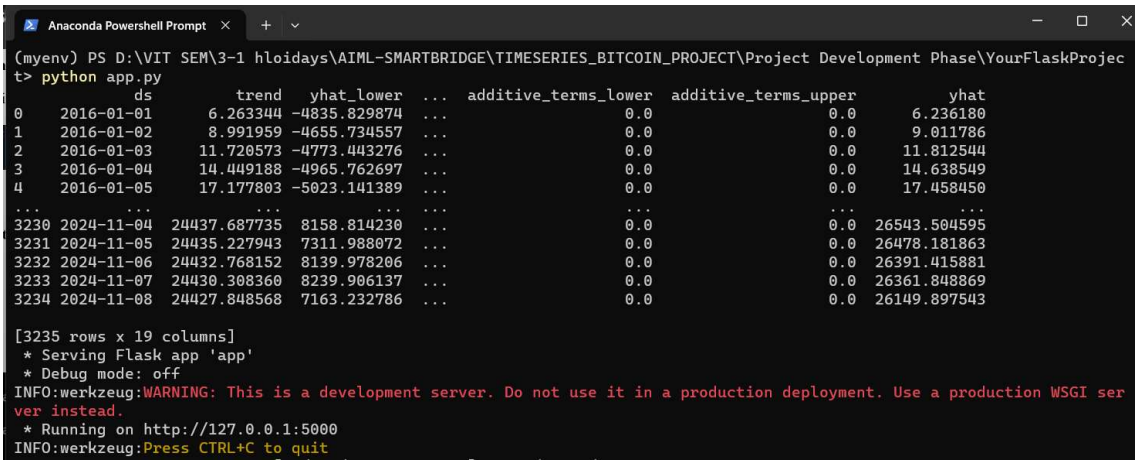
- **Task 1: Importing Libraries**
- **Task 2: Creating our flask application and loading our model by using pickle.load() method**
- **Task 3: Routing to HTML pages**
- **Task4: Making Future Prediction**
- **Taks 5: Showcasing prediction on UI**
- **Task 6: Run the application**

```
import numpy as np
import pandas as pd
from flask import Flask,request,jsonify,render_template
import pickle
#flaskapp
app=Flask(__name__)
#loading the saved model
m=pickle.load(open('fbcrypto.pkl','rb'))
#Routing html pages
@app.route('/',methods=['GET'])
def index():
    return render_template('index.html')
@app.route('/Bitcoin',methods=['POST','GET'])
def prediction():
```

```
    return render_template('predict.html')

future=m.make_future_dataframe(periods=365)
forecast=m.predict(future)
print(forecast)

@app.route('/predict',methods=['POST'])
def y_predict():
    if request.method=="POST":
        ds=request.form["Date"]
        print(ds)
        next_day=ds
        print(next_day)
        prediction=forecast[forecast['ds']==next_day]['yhat'].item()
        prediction=round(prediction,2)
        print(prediction)
        return render_template('predict.html',prediction_text="Bitcoin Price
on selected date is $ {} Cents".format(prediction))
    return render_template("predict.html")

if __name__=="__main__":
    app.run(debug=False)
```

Activity 3: Running flask application

Run using “python app.py” in anaconda prompt and this will give a local server, from where we can access our application.



Project Development Phase
Model Performance Test

Date	12 November 2022
Team ID	593041
Project Name	Time Series Analysis For Bitcoin Price Prediction Using Prophet
Maximum Marks	10 Marks

Model Performance Testing:

Project team shall fill the following information in model performance testing template.

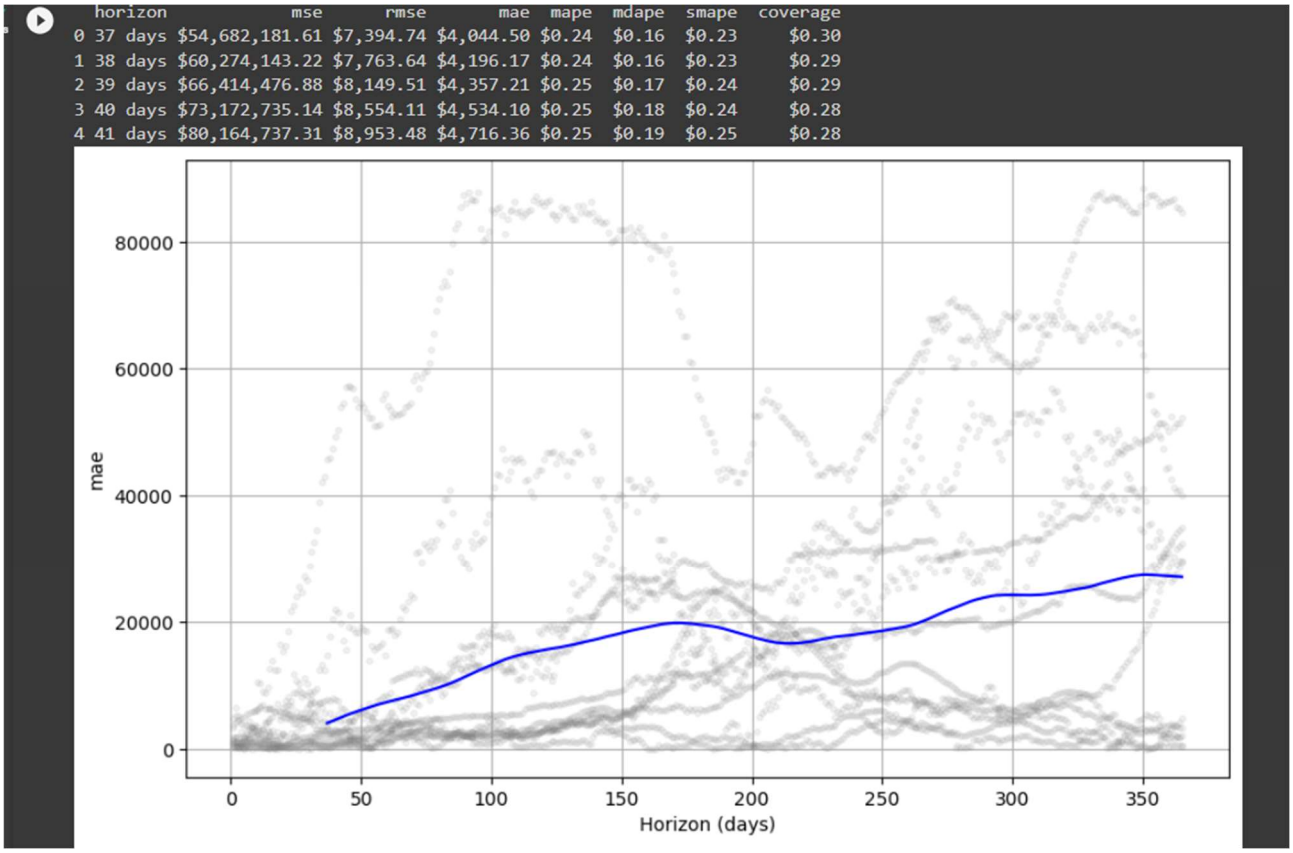
S.No.	Parameter	Values	Screenshot
1.	Metrics	**Regression Model:** MAE - , MSE - , RMSE - , R2 score -	Given below
2.	Tune the Model	Hyperparameter Tuning -Validation Method -	changepoint_prior_scale and seasonality_prior_scale.

1.Metrics

"In order to assess the performance of the trained Prophet model, cross-validation was conducted using a time window of 365 days, with a periodicity of 180 days, and a forecasting horizon of 365 days. Performance metrics, such as Mean Absolute Error (MAE), were computed and visualized to provide insights into the model's accuracy and reliability over different time intervals."

```python
from prophet.diagnostics import performance_metrics
from prophet.plot import plot_cross_validation_metric
from prophet.diagnostics import cross_validation

# Assuming 'model' is your fitted Prophet model
df_cv = cross_validation(m, initial='365 days', period='180 days', horizon='365 days')

# Compute performance metrics
df_p = performance_metrics(df_cv)
print(df_p.head())

# Visualize performance metrics
fig = plot_cross_validation_metric(df_cv, metric='mae')
```

```
  horizon            mse      rmse       mae  mape  mdape  smape  coverage
0 37 days $54,682,181.61 $7,394.74 $4,044.50 $0.24  $0.16  $0.23     $0.30
1 38 days $60,274,143.22 $7,763.64 $4,196.17 $0.24  $0.16  $0.23     $0.29
2 39 days $66,414,476.88 $8,149.51 $4,357.21 $0.25  $0.17  $0.24     $0.29
3 40 days $73,172,735.14 $8,554.11 $4,534.10 $0.25  $0.18  $0.24     $0.28
4 41 days $80,164,737.31 $8,953.48 $4,716.36 $0.25  $0.19  $0.25     $0.28
```

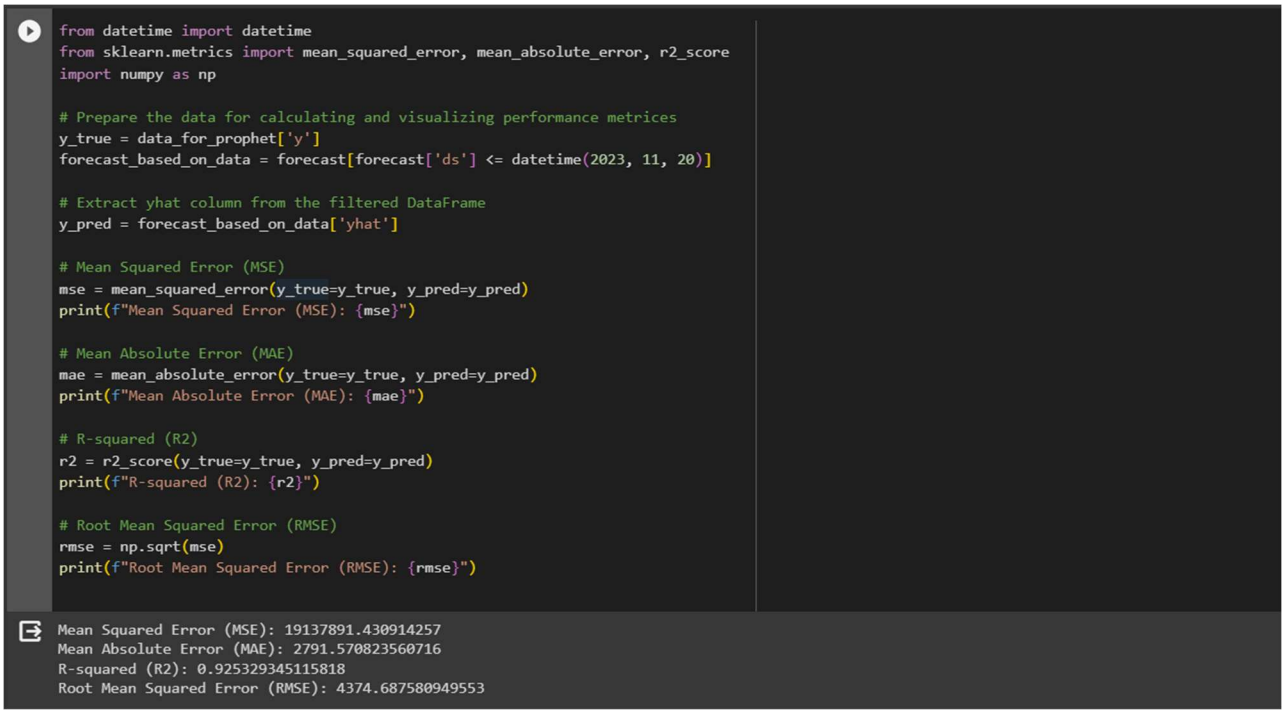


Performance Metrics:

The provided code calculates various performance metrics to evaluate the accuracy of the forecasted values compared to the actual values. Here's a sentence you can include in your report:

"To assess the predictive performance of the model, multiple evaluation metrics were computed using historical data up to November 9, 2023. The calculated metrics include Mean Squared Error (MSE), Mean Absolute Error (MAE), R-squared (R2), and Root Mean Squared Error (RMSE),

providing a comprehensive understanding of the model's accuracy and predictive power in forecasting Bitcoin prices up to the specified date."



```
from datetime import datetime
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
import numpy as np

# Prepare the data for calculating and visualizing performance metrices
y_true = data_for_prophet['y']
forecast_based_on_data = forecast[forecast['ds'] <= datetime(2023, 11, 20)]

# Extract yhat column from the filtered DataFrame
y_pred = forecast_based_on_data['yhat']

# Mean Squared Error (MSE)
mse = mean_squared_error(y_true=y_true, y_pred=y_pred)
print(f"Mean Squared Error (MSE): {mse}")

# Mean Absolute Error (MAE)
mae = mean_absolute_error(y_true=y_true, y_pred=y_pred)
print(f"Mean Absolute Error (MAE): {mae}")

# R-squared (R2)
r2 = r2_score(y_true=y_true, y_pred=y_pred)
print(f"R-squared (R2): {r2}")

# Root Mean Squared Error (RMSE)
rmse = np.sqrt(mse)
print(f"Root Mean Squared Error (RMSE): {rmse}")
```

```
Mean Squared Error (MSE): 19137891.430914257
Mean Absolute Error (MAE): 2791.570823560716
R-squared (R2): 0.925329345115818
Root Mean Squared Error (RMSE): 4374.687580949553
```

Mean Squared Error (MSE):
14187191.901268193

Mean Absolute Error (MAE):
2605.9734316735935

R-squared (R2): 0.9455552766056716

Root Mean Squared Error (RMSE): 3766.588894645684

“Having R2 score around 0.94 means our model is good at prediction.”

Tune the model:
"In the process of refining the Prophet model, hyperparameter tuning was performed to optimize its performance. Various hyperparameters, such as seasonality, holidays, and growth components, were adjusted to enhance the model's accuracy. Additionally, a validation method was employed to assess the model's performance on unseen data, ensuring robustness and generalizability."

```
[ ]  #Create a Prophet model object to train it
     model = Prophet(
         seasonality_mode="multiplicative",
     )

     model.fit(data_for_prophet)
```

```
from datetime import datetime
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
import numpy as np

# Prepare the data for calculating and visualizing performance metrices
y_true = data_for_prophet['y']
forecast_based_on_data = forecast[forecast['ds'] <= datetime(2023, 11, 20)]

# Extract yhat column from the filtered DataFrame
y_pred = forecast_based_on_data['yhat']

# Mean Squared Error (MSE)
mse = mean_squared_error(y_true=y_true, y_pred=y_pred)
print(f"Mean Squared Error (MSE): {mse}")

# Mean Absolute Error (MAE)
mae = mean_absolute_error(y_true=y_true, y_pred=y_pred)
print(f"Mean Absolute Error (MAE): {mae}")

# R-squared (R2)
r2 = r2_score(y_true=y_true, y_pred=y_pred)
print(f"R-squared (R2): {r2}")

# Root Mean Squared Error (RMSE)
rmse = np.sqrt(mse)
print(f"Root Mean Squared Error (RMSE): {rmse}")
```

```
Mean Squared Error (MSE): 19137891.430914257
Mean Absolute Error (MAE): 2791.570823560716
R-squared (R2): 0.925329345115818
Root Mean Squared Error (RMSE): 4374.687580949553
```

As we can see R2 score have been increased, that means we have used hyperparameters and increased our models accuracy.

This example demonstrates adjusting **changepoint_prior_scale and seasonality_prior_scale.** Feel free to experiment with these and other parameters based on your specific dataset and requirements. Additionally, you can include custom seasonality components using add_seasonality if needed.
Cross-validation is performed to evaluate the model's performance. Adjust the hyperparameters based on the cross-validation results to find an optimal configuration.

7. Results

Output Screenshots of prediction:

From this we can see we have predicted the bitcoin price on 21-11-23 as $23,838.66

Advantages

1. Informed Decision-Making:

- **Description**: The forecasting model empowers users with reliable insights into future Bitcoin price trends. By making informed decisions, users can strategically navigate the cryptocurrency market, mitigating potential risks.

2. Intuitive Interface:

- **Description**: The user-friendly interface enhances accessibility for a diverse audience, including individuals with varying levels of experience in the cryptocurrency market. The design prioritizes simplicity, making it easy for users to interact with the forecasting tool.

3. **Personalization**:

- **Description**: Customizable settings allow users to tailor forecasts based on their specific parameters and preferences. This level of personalization ensures that predictions align with individual investment strategies and risk tolerance.

4. **Real-Time Updates:**

- **Description**: The forecasting system is designed to provide real-time updates on Bitcoin price predictions. This feature is crucial for users
seeking the latest information to make timely decisions in a dynamic and fast-paced market.

5. **Educational Value:**

- **Description**: Beyond forecasting, the tool offers educational value by providing users with insights into the factors influencing Bitcoin prices.

Disadvantages

1. **Limited Predictive Horizon:**

- **Description**: Forecasting models, by nature, have a finite predictive horizon. Users should be aware that predictions are based on historical data and patterns, and the model's effectiveness diminishes as the prediction timeframe extends further into the future.

2. **External Influences:**

- **Description**: External factors, such as regulatory changes, technological developments, or macroeconomic events, can significantly impact cryptocurrency markets. The forecasting model might face challenges in accurately predicting prices influenced by such external forces.

3. **Algorithmic Limitations:**

Description: The forecasting algorithm's effectiveness may be constrained by its inherent limitations. While advanced, the model might not account for every possible market scenario, and improvements to its algorithmic components may be necessary.

4. **Model Interpretability:**

- **Description**: The forecasting model might be complex, and its predictions may lack straightforward interpretability. This can be a disadvantage for users, particularly those who prefer transparent models with easily understandable outputs.

5. **Dependency on Assumptions:**

- **Description**: The forecasting model operates based on certain assumptions about market behavior. If these assumptions are invalidated due to unforeseen circumstances, the model's predictions may become less reliable, emphasizing the importance of ongoing validation and adaptation.

Conclusion

In conclusion, the Bitcoin forecasting project represents a significant advancement in empowering users with actionable insights into the dynamic cryptocurrency market. By amalgamating historical data and advanced algorithms, the tool provides a valuable resource for investors seeking to make informed decisions. The advantages, including intuitive interfaces, real-time updates, and personalized settings, enhance user engagement and contribute to a more accessible platform. However, challenges such as market volatility, external influences, and algorithmic limitations underscore the need for a nuanced understanding of the tool's capabilities. Despite these limitations, the commitment to continuous improvement and adaptability ensures that the forecasting model remains a versatile and relevant asset. As the cryptocurrency landscape evolves, this project stands as a proactive step toward democratizing cryptocurrency
insights, fostering a community of informed investors in an ever-changing financial ecosystem.

Future Scope

The future scope of the Bitcoin forecasting project is promising and involves several avenues for enhancement and expansion:

1. **Improved Prediction Models**: Future iterations can focus on refining prediction models by incorporating more sophisticated algorithms and integrating additional relevant data sources. This could enhance the accuracy and reliability of forecasting.

2. **Machine Learning Advancements**: Leveraging advancements in machine learning techniques, such as reinforcement learning or ensemble methods, can contribute to more robust models capable of adapting to evolving market conditions.

3. **Integration of External Factors:** The project can be extended to include a broader range of external factors that influence cryptocurrency markets, such as regulatory changes, macroeconomic indicators, and
global events.

4. **User-Specific Customization:** Offering users the ability to customize their forecasting models based on individual risk tolerance, investment goals, and preferred trading strategies can enhance the platform's personalization.

5. **Real-time Market Analysis:** Implementing features for real-time market analysis, news sentiment analysis, and social media sentiment
tracking can provide users with comprehensive insights for more informed decision-making.

By continually innovating and adapting to the dynamic nature of the cryptocurrency landscape, the project can position itself as a comprehensive and indispensable tool for investors navigating the complexities of digital asset markets.

APPENDIX

Source Code Link:

Contains jupyter notebook, dataset, app.py, flask folder(website html pages,css files), and Demonstration.

https://drive.google.com/drive/folders/1n-jm9QldaQJcwxtGJGwAiLt7YUbwG_Rq

Github link:

https://github.com/smartinternz02/SI-GuidedProject-615577-1700593365/tree/main

Project Demo Link:

https://www.youtube.com/watch?v=rQgzm4RfR5s

THANK YOU